Assignment-1: Contextual Bandit-Based News Article Recommendation System

Objective

The goal of this assignment is to build a **news recommendation system** using a **contextual bandit approach**. The system will train an **RL model** to recommend articles based on user categories. Given a new user, the system will predict the most suitable news category and sample an article accordingly.

Dataset Description

You will be provided with two datasets:

1. **News Articles Dataset (`news_articles.csv`)**
   - Each row represents a **news article** with various **features**.
   - A **label** column specifies the **category** of the news article (arms of the bandit).

2. **User Data (`train_users.csv, test_users.csv`)**
   - Each row represents a **user** with various **features**.
   - A **label** column classifies the user into **User1, User2 and User3** (contexts for the bandit).

Reward Distributions Guide

You have been provided with a `.whl` file named **"sampler-1.0-py3-none-any.whl"**, which contains a class called **`sampler`**. This class is used to sample rewards from **unknown probability distributions**.

Installation Instructions:

1. Place the `.whl` file in the **same directory** as your assignment code.
2. Install the package using `pip`:
   ```
   pip install /path/to/sampler-1.0-py3-none-any.whl
   ```
   (Replace `/path/to/` with the actual file path.)

Usage Instructions:

After installation, you can use the `sampler` class as follows:

```
from my_package.sampler import sampler  # Import the module

reward_sampler = sampler(i)  # Create an instance of the class, passing your roll number (i)
reward = reward_sampler.sample(j)  # Call the sample() function to get a reward from arm j
```

Important Note:

- The **roll number `i`** should be the last digits of your **ID number**.
  Example Mapping:

ID Number	**i (Roll Number)**
U202200078	78
U202200115	115
U202200001	1

Understanding the Bandit Problem

This setup represents a **4-bandit problem** with **3 different user contexts**.

- There are **4 news categories** (bandits).
- Each user category has **4 arms**, leading to a total of **12 arms**.

Mapping of `j` values to arms:

j Values	**Arm Mapping (News Category, User Category)**
{0,1,2,3}	(Entertainment, User1), (Education, User1), (Tech, User1), (Crime, User1)
{4,5,6,7}	(Entertainment, User2), (Education, User2), (Tech, User2), (Crime, User2)
{8,9,10,11}	(Entertainment, User3), (Education, User3), (Tech, User3), (Crime, User3)

To obtain rewards for a specific arm, call the `sample(j)` function with the corresponding **`j` value**.

Tasks & Deliverables

Step 1: Data Pre-processing (10 Marks)

- Load the user dataset
- Perform **basic data cleaning**, handling missing values, and **feature encoding** if needed.

Step 2: Implementing Contextual Bandit Models (45 Marks)

2.1 Training Epsilon-Greedy Model (15 Marks)

- Treat **User Class (User1/User2/User3) as context** and **News Categories as arms**.

- Implement an **epsilon-greedy strategy**.
- Train a separate bandit model for **all 3 types of users**.
- **Compute Expected Reward Distribution** for each news category and for all 3 contexts.
- **Hyper tune epsilon by trying out different epsilon values, and compare expected payoffs for each epsilon.**

2.2 Training UCB model (15 Marks)

- Treat **User Class (User1/User2/User3) as context** and **News Categories as arms**.
- Implement a **UCB strategy**.
- Train a separate bandit model for **all 3 types of users**.
- **Compute Expected Reward Distribution** for each news category and for all 3 contexts.
- **Hyper tune C by trying out different C values, and compare expected payoffs for each C.**

2.3Training SoftMax model [take temperature parameter = 1] (15 Marks)

- Treat **User Class (User1/User2/User3) as context** and **News Categories as arms**.
- Implement an **SoftMax strategy**.
- Train a separate bandit model for **all 3 types of users**.
- **Compute Expected Reward Distribution** for each news category and for all 3 contexts.
-

Step 3: User Classification (10 Marks)

- Given a **new user's feature data**, classify them as **User1/User2/User3** using a **classification model** (e.g., Decision Tree, Logistic Regression, or a simple heuristic).
- Ensure the model has been trained on the given user dataset.

Step 4: Making Recommendations (20 Marks)

- Based on the classified user category, use all the three **trained contextual bandit policies** to **recommend a news category**.
- Sample a **random article** from the recommended category and **return its details**.

- ## Step 5: Evaluation & Analysis (20 Marks)

Using "test_users.csv" test your **Overall accuracy of classification** into User1/User2/User3

Train your RL models for a time horizon of 10000 steps and plot:-

- **Average reward collected over time** for each context
- Average reward comparison for different hyperparameters (different values of epsilon (in epsilon greedy) and C values in (UCB), Try at least 3 different values of these hyperparameters.

Prepare a comprehensive report analysing all three models, including detailed observations and comparisons. Ensure that all relevant plots are included in the report as part of your final submission.

Write your code such that :
Input = Features of new user
Output = what category of articles should be selected and a sampled article from the same category.

Datasets:

news.csv	#dataset for news articles
train_users.csv	#User dataset for training
test_users.csv	#User dataset for testing

Expected Learning Outcomes

By completing this assignment, you will: ✅ Understand **contextual bandits** and how they differ from standard bandits.
✅ Learn how to **train epsilon-greedy, UCB & SoftMax bandit models** for different contexts.
✅ Build an **adaptive recommender system** for users based on their characteristics.

Submission Guidelines

Please clone the GitHub repository provided for this assignment. Your submission should be pushed to the repository as a separate branch (do not push to the `main` branch).

Repository link: We will share the GitHub repo link for the submission on DLE announcements

Submission guidelines:

1. Ensure that you push all your code along with the results generated during the assignment (e.g., PDFs and images). Include any relevant files needed to fully understand your work.

2. Create a well-structured README file that serves as a report of your findings. This should include:

- A summary of your results and your discussion on the results.
- Detailed instructions on how to run and replicate your code. You may refer to online resources on creating effective README files (https://www.makeareadme.com/).
- The README file should essentially act as the report of your assignments.

3. Name your branch using the format \`firstname_rollno\` (e.g., \`john_12345\`).

The GitHub repository has been reformatted to include all assignments in one branch. Kindly maintain the same branch, you'll be using it throughout the course.

Due Date:- 11th Feb 2025
