Assignment: Predicting Customer Satisfaction using Logistic Regression

Objective

You are tasked with building a **binary logistic regression model** to predict **customer satisfaction** from airline service data. Your goal is to classify whether a customer is satisfied (`Satisfaction = 1`) or not (`Satisfaction = 0`) based on various service and demographic features.

Dataset Information

Each row in the dataset represents feedback from a single passenger. The original dataset contains the following columns:

- `Gender, Customer Type, Age, Type of Travel, Class, Flight Distance, Inflight wifi service, Departure/Arrival time convenient, Ease of Online booking, Gate location, Food and drink, Online boarding, Seat comfort, Inflight entertainment, On-board service, Leg room service, Baggage handling, Checkin service, Inflight service, Cleanliness, Departure Delay in Minutes, Arrival Delay in Minutes, Satisfaction`

Target column: `Satisfaction (Values: Satisfied/dissatisfied)`
(Will be converted to binary: `0 = neutral or dissatisfied`, `1 = satisfied`)

Instructions

1. Data Preparation

- Read the dataset using pandas.
- Encode categorical features manually (e.g., using one-hot encoding or label encoding).
- Drop rows with missing values if necessary.
- Convert the target column (`Satisfaction`) into binary numeric form:
    - `neutral or dissatisfied` → 0, `satisfied` → 1
- Move the target column to the last position in the dataframe.

2. Model Implementation

- Implement **logistic regression from scratch** (no use of libraries like `scikit-learn` or `statsmodels` for training).
- Use **gradient descent** to optimize the weights.
- Train the model using different learning rates: `0.1`, `0.01`, `0.001`, `0.0001`.

3. Analysis

- Plot the **total cost (loss)** vs **iterations** for each learning rate.
- Create a **summary table** that includes:
    - Learning rate
    - Final training accuracy
    - Final test accuracy

4. Evaluation

- Split the dataset into **80% training** and **20% testing**.
- Use **accuracy** as the evaluation metric.

Here is a sample pipeline of logistic regression

Dataset Preparation:

1. Select proper columns.
2. Normalize the dataset.
3. Randomly split the dataset into TRAINING(80%) and TEST(20%) sets.

Train (update Θ):

1. for each sample, $X = [x_1, x_2, ..., x_n]$ in TRAINING set:

   Concatenate 1 and turn it into $X' = [x_1, x_2, ..., x_n, 1]$
2. randomly initialize $\Theta = [\Theta_1, \Theta_2, ..., \Theta_{n+1}]$ within 0 to 1 // $\Theta_1, \Theta_2, ..., \Theta_n$:weights, Θ_{n+1}:bias
3. max_iter = 500, lr = 0.01
4. history = list()
5. for itr in [1, max_iter]:

   $J = 0$ //total cost

   for each sample, X' in TRAINING set:

   $z = X'.\Theta$ //use np.dot function

   $h = sigmoid(z)$ //sigmoid available in Python

   $L = -ylog(h) - (1 - y)log(h)$ //h = prediction label, y = true label

   $J += L$

   $dw = X'.(h - y)$ //$dim(dw) = n + 1$

   $\Theta = \Theta - dw * lr$ //$dim(\Theta) = n + 1$

   J /= N_train //N_train = # of training samples

   append J into history

Validation:

1. correct = 0
2. For each sample in X' in the TEST set:

   $z = X'.\Theta$

   $h = sigmoid(z)$

   if h >= 0.5: h=1

   else: h=0

   if h == y: correct += 1
3. *test_acc = correct*100/N_test* //N_test = # of testing samples

Deliverables

Submit the following:

1. A Python Jupyter notebook (`.ipynb`) that contains:
   - Code for data preprocessing
   - Your implementation of logistic regression
   - Visualizations and evaluation metrics
   - Comments and explanations
2. A short report (`.pdf`) including:

- Your methodology
- Summary of results and interpretations
- Table of learning rate vs accuracy

Do NOT use `scikit-learn`, `statsmodels`, or similar libraries to train the logistic regression model.