STAT 4830 Group Project

Optimization in Preference Learning:

A Utility-based Probability Prediction Method in Hotel Recommendation System

Shuhan Zhang Xinyu Zhang Lexuan Chen

1. Problem Statement

1.1 Overview

- When using platforms like **Expedia** to select hotels, users are often presented with a recommendation list tailored to their preferences.
- A key challenge is determining how to provide the **optimal set of hotel recommendations** from an enormous product pool for each consumer.

🎯 Objective

- Model **consumer hotel selection behavior** using a **utility-based approach**.
- Assuming that consumers make choices based on perceived utility, we can estimate their preferences through a **Multinomial Logit Model (MNL)**, where the selection probabilities are driven by utility values and determined via a **SoftMax function**.

🔍 Simplified Scenario

In the initial simplified scenario, we assume:

- Consumers share **identical preferences**.
- They choose from a **randomized offer set** of 2 to 4 hotels.

This foundational model allows us to establish the basic framework before expanding to more complex cases involving **heterogeneous consumer preferences** and **dynamic choice conditions**.

🚀 Goal

Minimize the error between the **estimated selection probabilities** and the **actual observed probabilities** of a hotel being chosen.

1.2 Background

- While individual choices may vary, consumers tend to exhibit a **consistent probability distribution** when presented with the same set of options.
- Understanding this decision-making process allows platforms like **Expedia** to **optimize recommendation algorithms** and ultimately **enhance booking conversion rates**.

1.3 Measurement Methods

- **Root Mean Square Error (RMSE)**
- **Accuracy**
- **Negative Log-Likelihood (NLL)**

These metrics are used to evaluate the performance of our model in estimating consumer choice probabilities.

1.4 Data & Utility

Despite the potential benefits, several challenges arise:

- **Limited visibility** into the exact number of hotels viewed by each user.
- **Variability** in the offer sets presented across different sessions.
- The impracticality of defining a **true probability distribution** for every possible combination of hotel sets.

- To address these issues, we focus on **inferring the utility of each hotel** rather than attempting to model the true probability distribution directly.
- Our analysis relies on **Expedia user booking data** and **hotel attributes**.
- By framing the problem around **utility estimation** instead of direct probability prediction, we aim to develop a model that is both **robust** and **adaptable** to varying conditions, effectively capturing consumer decision-making patterns.

2. Technical Approach

2.1 Mathematical Formulation

We model consumer hotel selection using a **Multinomial Logit (MNL) Model** , where the probability of choosing hotel from a set is:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

with *i* representing the latent utility of the hotel. This framework builds on the intuition introduced by Batsell and Polking (1985), who proposed a market share model based on choice probabilities and competitive interactions.

Our objective is to minimize prediction error using **MSE** and **Negative Log- Likelihood (NLL)** as the loss function:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(P_i - y_i)^2$$

$$NLL = -\sum_{n}\sum_{i \in S_n} y_{n,i} \log P_{n,i}$$

2.2 PyTorch Implementation & Validation Methods

- **Solver**
  We employ Adam as the optimizer due to its efficiency in handling sparse data.

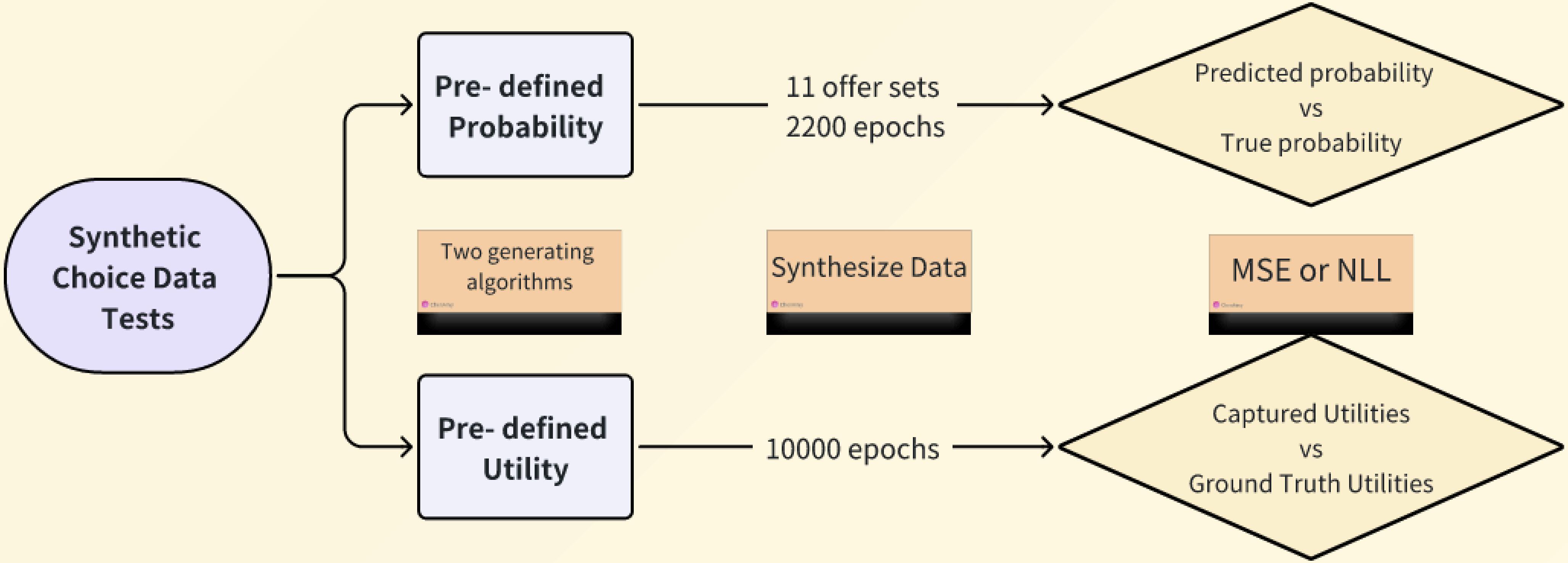
- **Implementation**
  The model is implemented in PyTorch , following a structured pipeline:

```
Model Construction → Data Import → Training → Testing & Validation
```

3. Initial results

3.1 Two Synthetic Choice Data Tests

Basic Frame


Synthetic Choice Data Tests
Pre- defined Probability
11 offer sets
2200 epochs
Predicted probability
vs
True probability
Two generating algorithms
Synthesize Data
MSE or NLL
Pre- defined Utility
10000 epochs
Captured Utilities
vs
Ground Truth Utilities

Test 1 --- Probability

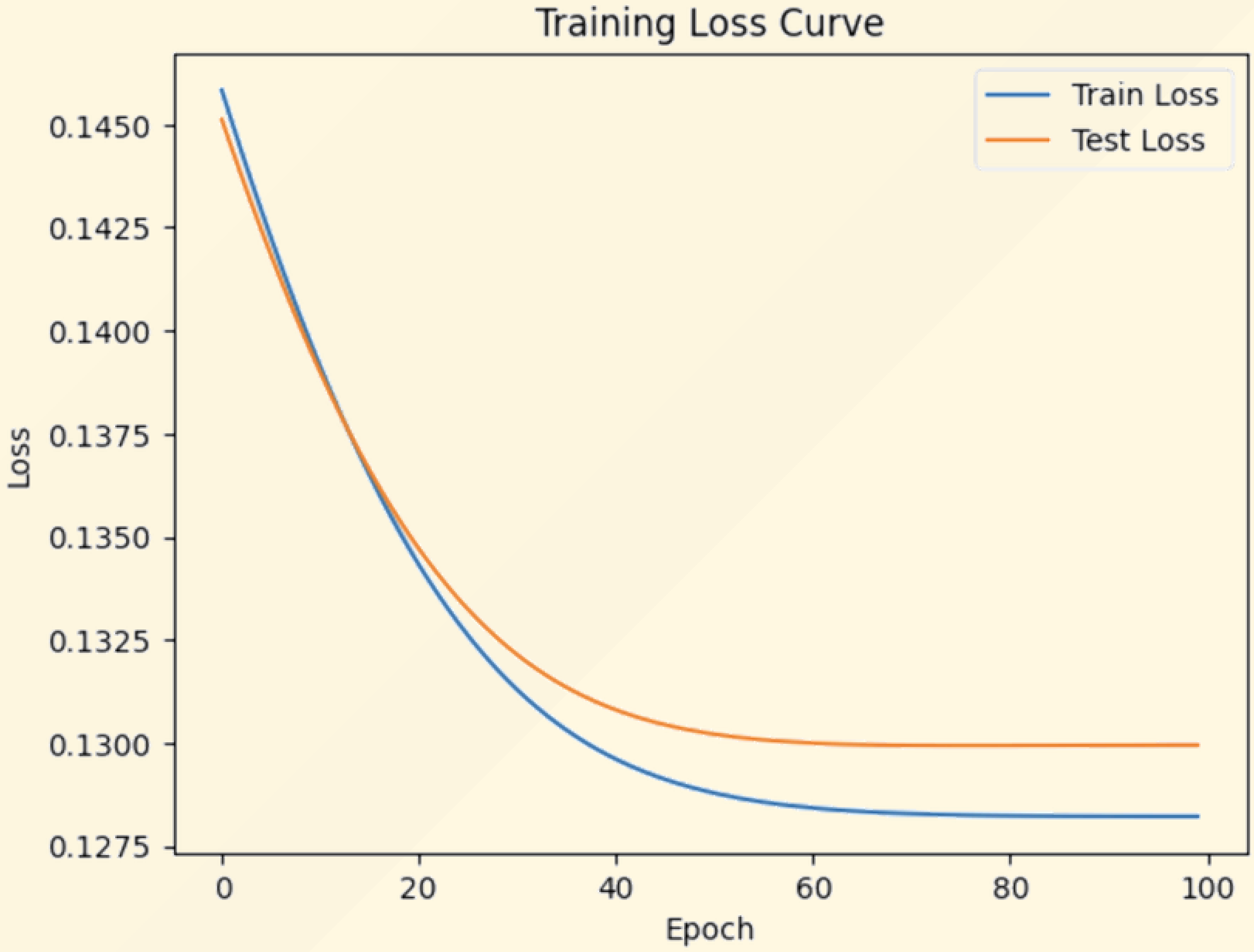
We first test the situation with homogenous pre-defined probability distribution.

```
hypothetical_choice_p = [[0.98, 0.02, 0, 0],
                         [0.5, 0, 0.5, 0],
                         [0.5, 0, 0, 0.5],
                         [0, 0.5, 0.5, 0],
                         [0, 0.5, 0, 0.5],
                         [0, 0, 0.9, 0.1],
                         [0.49, 0.01, 0.5, 0],
                         [0.49, 0.01, 0, 0.5],
                         [0.5, 0, 0.45, 0.05],
                         [0, 0.5, 0.45, 0.05],
                         [0.49, 0.01, 0.45, 0.05]]
```

Performance Metrics

Metric	In-Sample	Out-of-Sample
Original RMSE	0.3581	0.3605
Frequency RMSE	0.1289	0.1418

However, training progress appears to **halt prematurely**, suggesting potential inefficiencies related to **optimizer settings** or **model capacity constraints**.



Training Loss Curve
Train Loss
Test Loss
Loss
Epoch
0.1450
0.1425
0.1400
0.1375
0.1350
0.1325
0.1300
0.1275
0
20
40
60
80
100

Test 2 --- Utility

- In the second test, we predefined the ground truth utilities for each alternative and added Gumbel noise to simulate individual choice randomness.
- Synthetic choice data was generated based on these noisy utilities, and an MNL model was trained to recover the underlying utility parameters.
- The results show that while the captured utilities differ in absolute values, the model successfully learned the correct preference ranking, validating its effectiveness in choice modeling.

```python
def generate_synthetic_choice_data(num_alternatives, num_observations, utilities):

    choice_data = np.zeros((num_observations, num_alternatives))

    for i in range(num_observations):
        # Add random noise to utilities
        noisy_utilities = utilities + np.random.gumbel(size=num_alternatives)

        # Choose the alternative with the highest utility
        choice = np.argmax(noisy_utilities)

        # Update choice data
        choice_data[i, choice] = 1

    return choice_data

num_alternatives = 4
num_observations = 10000
ground_truth_utilities = np.array([0.5, 0.3, 0.2, 0.1])
choice_data = generate_synthetic_choice_data(num_alternatives, num_observations, ground_truth_utilities)
```

- The training loss decreased from **0.1873** to **0.1859**, stabilizing after ~230 epochs, indicating quick convergence.

3.2 Model Limitations & Future Directions

Our current research is conducted within a **highly simplified experimental framework**, leading to a model that lacks sufficient:

- **Feature complexity**
- **Explanatory depth**
- **Predictive robustness**

While the model shows **marginal performance gains** over naive baselines (e.g., simple mean estimation), its **limited utility for real-world applications** is evident.

3.3 Proposed Enhancements

To address these limitations, we propose the following refinements:

1. **Incorporating Item-Specific Attributes:**
   Integrate features such as:
   - **Temporal** (e.g., seasonality effects)
   - **Spatial** (e.g., location-based factors)
   - **Socioeconomic indicators**

2. **Introducing Consumer Segmentation Strategies:**
   Explicitly model **heterogeneous preference patterns** by differentiating between user subgroups, such as: **Price-sensitive consumers** and **Quality-driven consumers**

These enhancements aim to:

- **Improve interpretability**
- **Increase estimation accuracy**
- Better align the framework with the **complex dynamics of real-world decision-making** processes.

3.4 Key Takeaways

- **Stable convergence** observed, but with signs of early training stagnation.
- **Model performance** remains close to naive baselines, highlighting the need for enriched features.
- Future work will focus on **feature complexity** and **consumer segmentation** to enhance both **robustness** and **real-world applicability**.

4. Next Steps

4.1 Detailed Pathways

- Integrating Item-Specific Attributes

To incorporate item-specific features, we will employ two complementary approaches:

- **Linear Regression:** To capture simple, interpretable relationships between item attributes and choice probabilities.
- **Neural Networks:** To model complex, non-linear interactions that may not be easily captured through linear methods.

Both methods will be evaluated to determine their effectiveness in enhancing model accuracy and interpretability.

- Modeling Heterogeneous Consumer Preferences

- We will explicitly model **heterogeneous preference patterns** by differentiating between consumer subgroups (e.g., price-sensitive vs. quality-driven consumers).
- Drawing inspiration from Jagabathula et al. (2020), which explores choice modeling under heterogeneous behaviors, we aim to develop a **Python-based implementation** to bridge the gap left by existing methodologies.

4.2. Key Technical Challenges

While refining our model, we anticipate the following technical hurdles:

- **Solver Selection:** Identifying an efficient and scalable solver tailored to our problem structure.
- **PyTorch Proficiency:** Enhancing our expertise with PyTorch to optimize model performance and manage complex architectures.
- **Large-Scale Data Management:** Addressing GPU memory constraints when handling large datasets, including strategies for **efficient parallelization** across varying choice sets.

4.3 Future Exploration

To refine our approach, we seek:

- **Guidance on PyTorch Best Practices:** Techniques for optimizing model performance and handling large-scale data efficiently.
- **Access to Relevant Case Studies:** Practical examples of projects with similar objectives to inform our methodology.
- **Alternative Modeling Strategies:** Exploration of advanced segmentation methods to capture nuanced consumer behaviors.

5. Group members

Name	Email
Shuhan Zhang	zhang19@sas.upenn.edu
Xinyu Zhang	joyxyz@sas.upenn.edu
Lexuan Chen	amy0305@sas.upenn.edu