

# Table of Contents

- Key Features:
- Improved Version:

I am working on a project of choice model. Help me improve the following overview for my weakly report.

In this week, wo add item features to our regression to specify the utility value of different items. In this way can we increase the explanation power of our model and do better in producing the possibility of people buying something not included in their previous offer set. We also apply the real-world dataset from Expedia and divide them into train and test subset.

We use both linear regression and a neural network structure to analyze the problem.

Here's a revised version of your overview with clearer phrasing and improved flow:

This week, we incorporated item features into our regression model to better specify the utility values of different items. By doing so, we aim to enhance the explanatory power of our model and improve its ability to predict the likelihood of people purchasing items not included in their previous offer set.

Additionally, we utilized a real-world dataset from Expedia, dividing it into training and testing subsets for model evaluation. To analyze the problem, we applied both linear regression and a neural network framework, comparing their performance and insights.

Let me know if you'd like me to tweak it further!

For the following description of problem background, make it fluent.

In Expedia, there are thousands of hotels with different features including price, number of rooms, space and so on. Not all of them may be shown to every consumer. We only know the choice of each consumer with the given offer set. We want to explore how will consumers choose within the general set. How can we predict the possibility that the consumer choosing a hotel that haven't appeared in their offer set? That's what we are interested in. We believe utility value may be an helpful term. Based on our economic knowledge, we would like to assume that consumers are rational and they make decisions based on the utility value of different hotels. The utility value is decided by the feature of items. We hope this work may model consumer's choice better and help develop a better recommendation system to consumers.

Here's a more fluent and polished version of your description:

---

Expedia features thousands of hotels with various attributes, such as price, number of rooms, and space. However, not all hotels are shown to every consumer; we only observe each consumer's choice from their specific offer set. Our goal is to explore how consumers make decisions within the broader general set and to predict the likelihood of a consumer selecting a hotel that was not part of their initial offer set.

We believe the concept of utility value can be a useful framework for this analysis. Drawing from our economic knowledge, we assume that consumers are rational and base their decisions on the utility value of different hotels, which is determined by their features. Through this work, we aim to better model consumer choice behavior and contribute to the development of more effective recommendation systems for consumers.

---

Let me know if you'd like further adjustments!

If you are an expert in economics or orie,  
improve the paragraph

Here's a more refined version tailored for an audience in economics or operations research:

Expedia offers thousands of hotels, each characterized by attributes such as price, number of rooms, and available space. However, not all hotels are presented to every consumer; we only observe a consumer's choice within their personalized offer set. Our objective is to model consumer decision-making over the broader universal set of options and to predict the likelihood of a consumer selecting a hotel that was not included in their initial offer set.

To address this, we draw upon the concept of utility maximization, a cornerstone of economic theory. Assuming rational behavior, we posit that consumers evaluate hotels based on their utility, which is determined by the features of each option. By estimating these utility values, we aim to build a model that captures consumer choice behavior more accurately and supports the development of enhanced recommendation systems, thereby improving the match between consumer preferences and offered options.

This version incorporates technical phrasing and a stronger theoretical foundation. Let me know if you'd like it adjusted further!

improve the following illustration.  
\$\$  
\sigma(w)\_i = \frac{e^{wx\_i}}{\sum\_{j=1}^K e^{wx\_j}}  
\$\$

In this function  $x_i$  is a vector represents the features of hotel  $i$  and  $w$  is the weight vector.  $wx_i$  gives the utility of every hotel. We apply item features to the multinomial logit model and get the result.

Here's an improved and more formal explanation of your illustration:

The function

$$\sigma(w)_i = \frac{e^{wx_i}}{\sum_{j=1}^K e^{wx_j}}$$

represents the probability that a consumer selects hotel  $i$  from a choice set of  $K$  hotels, based on a multinomial logit model.

Here,  $x_i$  is a feature vector representing the attributes of hotel  $i$  (e.g., price, number of rooms, location), and  $w$  is the weight vector capturing the importance of each feature. The term  $wx_i$  represents the utility of hotel  $i$ , as perceived by the consumer.

By incorporating item features into the multinomial logit framework, we compute the probability of each hotel being selected. This approach allows us to model consumer choice behavior more effectively and derive insights into the relative importance of different hotel attributes.

Let me know if further clarification or additions are needed!

Next, I will give you a code calculate utility value based on neural network instead of weighted sum of features. Explain the frame of the neural network.

```
class Deep_MNL(nn.Module):
    def __init__(self, input_dim):
        super(Deep_MNL, self).__init__()
        self.layer1 = nn.Linear(input_dim, 100,
dtype=torch.float64, bias=False).to(device)
        self.layer2 = nn.Linear(100, 100,
dtype=torch.float64, bias=False).to(device)
        self.layer3 = nn.Linear(100, 1,
dtype=torch.float64, bias=False).to(device)
        # self._initialize_weights()

    def _initialize_weights(self):
        nn.init.constant_(self.layer1.weight, 0)
        nn.init.constant_(self.layer1.bias, 0)
        nn.init.constant_(self.layer2.weight, 0)
        nn.init.constant_(self.layer2.bias, 0)
        nn.init.constant_(self.layer3.weight, 0)
        nn.init.constant_(self.layer3.bias, 0)

    def forward(self, x):
        x = self.layer1(x)
        x = torch.sigmoid(x)
        x = self.layer2(x)
        x = torch.sigmoid(x)
```

```
x = self.layer3(x)
return x
```

Here's an explanation of the neural network structure used in your code:

---

The `Deep_MNL` class implements a neural network to calculate utility values, replacing the traditional weighted sum of features with a more flexible and nonlinear approach. The network is structured as follows:

#### 1. Input Layer:

- The input layer takes a vector of features for each hotel, with dimensionality specified by `input_dim`.
- This vector represents the attributes of the hotel, such as price, location, and other relevant factors.

#### 2. Hidden Layers:

- Layer 1: The first hidden layer ( `self.layer1` ) has 100 neurons and applies a linear transformation to the input features, without a bias term. The output is then passed through a sigmoid activation function to introduce nonlinearity.
- Layer 2: The second hidden layer ( `self.layer2` ) also has 100 neurons. It further processes the output of the first layer using another linear transformation followed by a sigmoid activation function.

#### 3. Output Layer:

- Layer 3: The final layer ( `self.layer3` ) reduces the dimensionality to 1. It computes a linear combination of the outputs from the second hidden layer, producing the predicted utility value for the hotel.

#### 4. Forward Propagation:

- The `forward` method defines the flow of data through the network. The input is sequentially processed by `layer1`, `layer2`, and `layer3`, with sigmoid activations applied after the first two layers.

#### 5. Optional Weight Initialization (Commented Out):

- The `_initialize_weights` method provides an option to initialize the weights and biases of the network to zero. This ensures control over the starting parameters, although it is currently not used in the provided code.

### Key Features:

- **Nonlinear Transformations:** By using sigmoid activation functions, the network can capture complex, nonlinear relationships between hotel features and their utility values.
- **No Bias Terms:** Bias terms are explicitly set to `False`, simplifying the network but potentially limiting flexibility in some cases.
- **Deep Learning Framework:** The network consists of multiple layers, allowing it to learn hierarchical representations of the input features, improving its ability to predict utility values in complex scenarios.

This neural network replaces the simple weighted sum in traditional multinomial logit models with a more expressive structure, enabling it to capture more nuanced patterns in the data. Let me know if you'd like additional clarification or modifications!

Explain the frame of neural network in a brief way.

The `Deep_MNL` neural network calculates utility values using a three-layer structure:

1. **Input Layer:** Takes the feature vector of a hotel (*input\_dim*).
2. **Hidden Layers:** Two fully connected layers (`layer1` and `layer2`), each with 100 neurons and sigmoid activation for nonlinearity.
3. **Output Layer:** A single neuron (`layer3`) produces the utility value for the hotel.

The network models complex, nonlinear relationships between features and utility, replacing the traditional weighted sum approach in multinomial logit models.

A little more detailed.

The `Deep_MNL` neural network calculates utility values using the following structure:

1. **Input Layer:**
  - Takes a feature vector of a hotel (*input\_dim*) that represents attributes such as price, location, and size.
2. **Hidden Layers:**
  - **Layer 1:** A fully connected layer with 100 neurons, applying a linear transformation to the input features. A sigmoid activation function introduces nonlinearity.

- Layer 2: Another fully connected layer with 100 neurons, processing the output of Layer 1 with another linear transformation and sigmoid activation.

### 3. Output Layer:

- Layer 3: A fully connected layer with a single neuron, which outputs the predicted utility value for the hotel.

### 4. Forward Propagation:

- Data flows sequentially through `layer1`, `layer2`, and `layer3`, with sigmoid activations applied after the first two layers.

This structure captures nonlinear relationships between hotel features and utility values, offering more flexibility and accuracy compared to traditional linear models.

Improve the following part.

Next, we consider that the weighted sum of features may not catch some inlinear relationship between feature and utility value. Therefore, wo use neural network to get more insight.

The Deep\_MNL neural network calculates utility values using the following structure:

Input Layer:

Takes a feature vector of a hotel (`input_dim`) that represents attributes such as price, location, and size.

Hidden Layers:

Layer 1: A fully connected layer with 100 neurons, applying a linear transformation to the input features. A sigmoid activation function introduces nonlinearity.

Layer 2: Another fully connected layer with 100 neurons, processing the output of Layer 1 with another linear transformation and sigmoid activation.

Output Layer:

Layer 3: A fully connected layer with a single neuron, which outputs the predicted utility value for the hotel.

Forward Propagation:

Data flows sequentially through `layer1`, `layer2`, and `layer3`, with sigmoid activations applied after the first two layers.

This structure captures nonlinear relationships between hotel features and utility values, offering more flexibility and accuracy compared to traditional linear models.

Here's an improved version of your description:

Next, we consider that the weighted sum of features may fail to capture nonlinear relationships between features and utility values. To address this limitation, we use a neural network to gain deeper insights.

The Deep\_MNL neural network calculates utility values through the following structure:

- 1. Input Layer:  
Accepts a feature vector of a hotel (*input\_dim*) that represents attributes such as price, location, and size.
- 2. Hidden Layers:
  - Layer 1: A fully connected layer with 100 neurons that applies a linear transformation to the input features. Nonlinearity is introduced using a sigmoid activation function.
  - Layer 2: Another fully connected layer with 100 neurons, which processes the output from Layer 1 with a linear transformation followed by a sigmoid activation.
- 3. Output Layer:
  - Layer 3: A fully connected layer with a single neuron, producing the final utility value prediction for the hotel.
- 4. Forward Propagation:
  - Data flows sequentially through `layer1` , `layer2` , and `layer3` . Sigmoid activation functions are applied after the first two layers to model complex, nonlinear relationships.

By replacing the traditional weighted sum with this neural network structure, we can better capture the intricate relationships between hotel features and their utility values, providing a more flexible and accurate framework for modeling consumer choice behavior.

This version improves readability, flow, and precision. Let me know if you'd like any further refinements!



You are a professor in Economics and ORIE, help improve the following paragraph about future direction. In the future, we plan to take matrix completion into our consideration. As there are numerous features for hotels, the time complexity of running MNL with features is huge. For easiness, we may run unsupervised learning first and assign hotels and consumers into different types. We can apply matrix completion based on clustered dataset.

### **Improved Version:**

In future work, we plan to incorporate matrix completion into our approach. Given the high dimensionality of hotel features, running a multinomial logit (MNL) model with all features can be computationally expensive. To address this, we propose leveraging unsupervised learning techniques to efficiently classify both hotels and consumers into distinct clusters. By grouping similar entities, we can reduce the complexity of the problem while preserving meaningful patterns in consumer choice behavior. Matrix completion can then be applied within these clustered datasets to infer missing preferences and improve recommendation accuracy, making the model both scalable and robust.

---

This version enhances clarity, academic tone, and methodological rigor. Let me know if you'd like any refinements!