Wireless earphone recommendation system

Jiwon Lee

1. Introduction

This project means to make a recommendation system to give one a help in buying a wireless earphone. There are various types of factors you should consider before buying a wireless earphone, and I wanted to make a system to shorten the concern. For this recommendation system, only 6 factors were in concern, the weight of wireless earphone, the cost, the sound quality, battery time, and the brand of the wireless earphone.

There are usually two types of recommendation type: content-based filtering system and collaborating filtering system. Content-based filtering recommends user items that are similar to items user has preferred. However, collaborative filtering is much frequently used than the content-based filtering. Collaborative filtering has two types; nearest neighbor based collaborative filtering and latent based collaborative filtering. Collaborative filtering recommends item to a user according to user behavior. Nearest neighbor collaborative filtering means to recommend user items the user have not used yet. Latent factor based collaborative filtering uses matrix factorization to minimize the storage space.

However, I did not have to fill in blanks according to other items, or other users, but had all the data and needed to classify it into 6 types of wireless earphone. Therefore, I used classification system instead of well-known recommendation system. I used Softmax Regression to solve this multi-class classification system. Unlike logistic regression, which is used for binary classification, Soft Regression is used when multi-class exists. Softmax returns the result, or the probability of each class as sum of 1. Therefore, the class with the highest probability becomes the final output.

Keras of Tensorflow provides the Softmax function, so I used the built-in function in this project.

2. Methods

2.1. Data collection

Data was collected through survey done with Google Form. I was unable to gather data from the actual sales record. Each 5 best sale wireless earphone were chosen in Coupang, Interpark, and Naver sales record, without overlap, 11 wireless earphones were chosen. The 11 earphones were (Airpod 1, Airpod 2, Airpod 3, Buds 1, Buds 2, Buds pro, Penton VIBER, SONY WF-1000XM4, QCY T13, QCY T5).

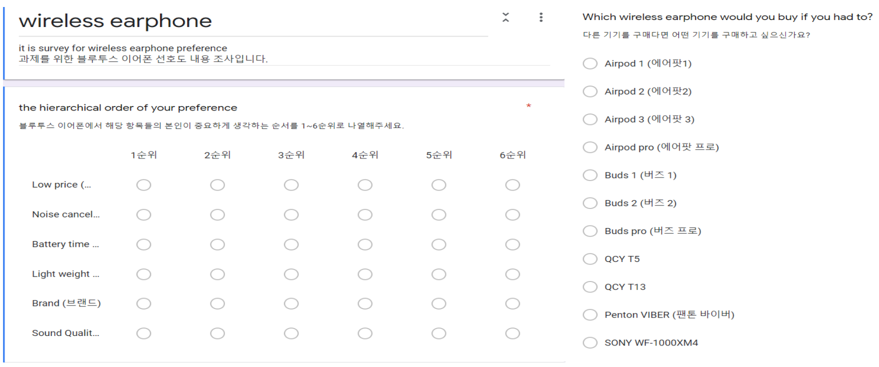

Figure 1. Google form survey questions

The respondents were asked to answer the hierarchical order of the factors each respondent thought important when buying a wireless earphone and the device they wish to buy according to the order they have chosen. Survey was done from November 26th to November 29th, for 4 days, and 92 answers were given. The raw data is shown in figure 2.

	A	B	C	D	E	F	G	H
1	타임스탬프	the hierarchical order of y	the hierarchical order of y	the hierarchical order of y	the hierarchical order of y	the hierarchical order of y	the hierarchical order of y	Which wireless earphone
2	11-26-2021 21:39:52	1순위	4순위	3순위	5순위	6순위	2순위	Airpod pro (에어팟 프로)
3	11-26-2021 21:40:06	4순위	2순위	1순위	1순위	3순위	1순위	Airpod 3 (에어팟 3)
4	11-26-2021 21:43:50	3순위	5순위	4순위	6순위	2순위	1순위	Buds pro (버즈 프로)
5	11-26-2021 21:47:07	4순위	3순위	5순위	6순위	2순위	1순위	Buds pro (버즈 프로)
6	11-26-2021 21:47:38	3순위	4순위	6순위	5순위	1순위	2순위	Buds pro (버즈 프로)
88	11-29-2021 16:25:37	5순위	6순위	4순위	3순위	2순위	1순위	Airpod 3 (에어팟 3)
89	11-29-2021 16:41:34	3순위	4순위	2순위	5순위	6순위	1순위	Buds pro (버즈 프로)
90	11-29-2021 16:49:34	6순위	3순위	1순위	4순위	5순위	2순위	SONY WF-1000XM4
91	11-29-2021 17:39:21	5순위	6순위	1순위	2순위	4순위	3순위	SONY WF-1000XM4
92	11-29-2021 18:34:25	4순위	6순위	3순위	5순위	1순위	2순위	Airpod 3 (에어팟 3)
93	11-29-2021 20:46:13	6순위	2순위	3순위	5순위	1순위	4순위	Airpod pro (에어팟 프로)
94								

Figure 2. Raw survey data

In this case, there are 6 factors and 7 classes. I excluded data that had classes that of less than 3 responses. (Airpod 1, QCY T5 had no response, Buds 1 had 1, Penton VIBER had 2). Thus, there were 86 valid data. Figure 4 shows the 86 valid data.

	A	B	C	D	E	F	G	H
1	userid	Low price	Noise cancelling	Battery time	Light weight	Brand	Sound Quality	wish device
2	1	1	4	3	5	6	2	Airpod pro
3	2	3	5	4	6	2	1	Buds pro
4	3	4	3	5	6	2	1	Buds pro
5	4	3	4	6	5	1	2	Buds pro
6	5	4	5	2	1	6	3	Airpod pro
83	82	3	4	2	5	6	1	Buds pro
84	83	6	3	1	4	5	2	SONY WF-1000XM4
85	84	5	6	1	2	4	3	SONY WF-1000XM4
86	85	4	6	3	5	1	2	Airpod 3
87	86	6	2	3	5	1	4	Airpod pro

Figure 3. 86 valid data

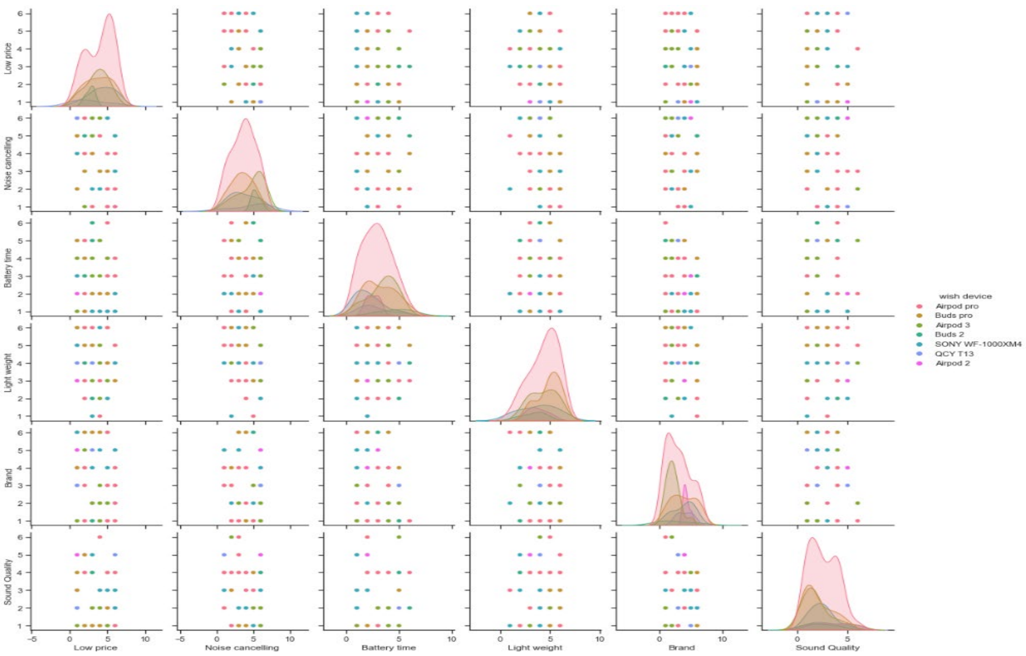

Figure 4. pair plot

As vector matrix, Weight vector had dimension of 7*6, input vector of 6*,1 and bias of 7*1, and output vector of 6*1. The pair plot in figure 4 shows the scattered plot between each input features. Clear relationship is not shown in the graph.

2.2. Data preprocessing

```
# data preprocessing
data['wish device'] = data['wish device'].replace(['Airpod pro','Buds pro', ₩
                                                   'Airpod 3','SONY WF-1000XM4', ₩
                                                   'Airpod 2', 'QCY T13', 'Buds 2' ],₩
                                   [0,1,2,3,4,5,6])

data_X = data.iloc[:,1:7].values
data_y = data['wish device'].values

(X_train, X_test, y_train, y_test) = train_test_split(data_X, data_y, train_size=0.8, random_state=1)

y_train = to_categorical(y_train, num_classes = 7)
y_test = to_categorical(y_test, num_classes = 7)
```

Figure 5. Data processing code

First, to classify the output vectors, they were substituted to integer labels from 0 to 6, according to their population (shown in figure 6). They were each respectively assigned from ['Airpod pro','Buds pro', 'Airpod 3','SONY WF-1000XM4', 'Airpod 2', 'QCY T13', 'Buds 2'] to [0,1,2,3,4,5,6]). To process these integer output data, one-hot encoding was used. In this case, ‘Airpod pro’ of label 0 and ‘Buds pro’ of label 1 does not have close relevance. The output (0 and 1) does not have any close relevance than output (1 and 2). Therefore, to strip the relevance of each label from each other, one-hot-encoding was used. Unlike integer encoding, one-hot encoding will distribute equivalent error between each output.

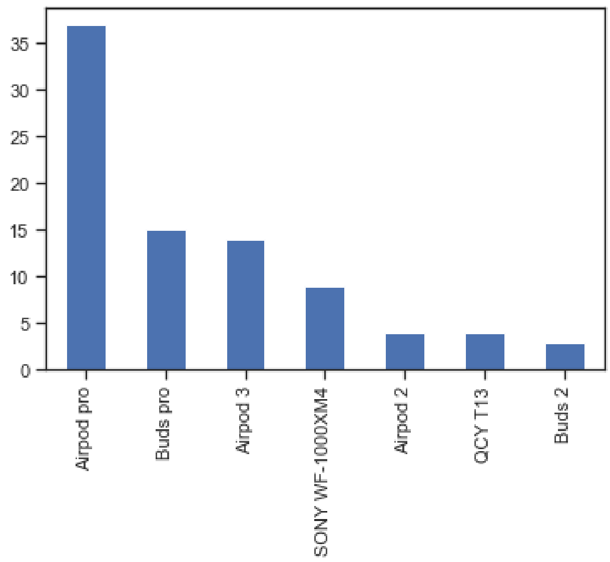

Figure 6. Wish device population

In keras framework, the function for one-hot encoding is given; the to_categorical function. In this case, it was necessary to set the num_classes parameter as 7. Due to small size of data, class with little data may not be included in either test or train dataset. Therefore, it was required to set the num_classes value as fixed.

The data was split into train and test dataset with proportion of 8:2 with random state fixed with every training trials. Array of the hierarchical order was used as data_X and the wish device conversed into integer number was used as the target, data_y. data_X has 6 features, so it has 6*1 dimension and data_y has 7 outputs, so it has 7*1 dimension.

2.3. Model

```
# create model
model = Sequential()
model.add(Dense(7, input_dim=6, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

# train the model
history = model.fit(X_train, y_train, epochs=200, batch_size=1, validation_data=(X_test, y_test))
```

Figure 7. Creating model and training code

For this system, I used Softmax Regression to solve the multi-class Classification problem, which is generated from logistic regression for multi-classification problems. Softmax estimates the probability of k class with an input of k dimension vector. Softmax function defines the probability for each class as following equation.

$$p_i = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}} for\ i = 1,2, \dots k$$

Softmax function of k class returns result of following vector

$$softmax(z) = \left[\frac{e^{z_1}}{\sum_{j=1}^{k} e^{z_j}}\ \frac{e^{z_2}}{\sum_{j=1}^{k} e^{z_j}}\ \dots\ \frac{e^{z_k}}{\sum_{j=1}^{k} e^{z_j}}\right] or\ i = 1,2, \dots k$$

Each probability adds up to 1.

Built-in sequential model was used. Optimizer parameter was chosen as 'sgd' or 'adams'. Metrics parameter was chosen as 'accuracy' to check the accuracy of the system.

For cost function, cross entropy function was used.

$$cost = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k} y_j^{(i)}\log(p_j^{(i)}) = -\frac{1}{n}\sum_{i=1}^{n}[y_j^{(i)}\log\left(p_j^{(i)}\right) + \left(1 - y_j^{(i)}\right)\log\left(1 - p_j^{(i)}\right)]$$

y refers to the real value, k refers to the number of class. y_j is the j index of the one-hot vector, and p_j is the probability of sample data being the j class. Cross entropy is the cost function of Softmax regression. It has the same form of cost function of logistic regression, only differs in that in logistic regression, due to binary classification, k is substituted to 2. It can be chosen in Sequential model by having 'loss' parameter as 'categorical_crossentropy'. The model was initially trained with adams optimizer, and then it was tested under various conditions.

3. Results & Analysis

The model was tested initially with adam optimizer after 200 epochs with batch size of 1. Model reached accuracy of 0.3888888955116272 and loss was 1.3660. Loss decreased exponentially along repeated train as shown in figure 8.

Then the model was tested with different epoch numbers, and batch sizes. The results are shown in the table 1 and 2. It was tested with batch numbers of 1, 2, 4, 8, and 16. As batch number increased, the system reached the highest accuracy at batch size of 2, where the accuracy was 0.4444444477558136. accuracy decreased with batch size over 2. Then, the system was tested with different epoch numbers of 50, 100, 200, 400, and 800. With change of epoch number, the system showed best accuracy in 200 epoch. 50 epoch had the same accuracy, but loss was higher. It is shown in figure 9, figure 10.

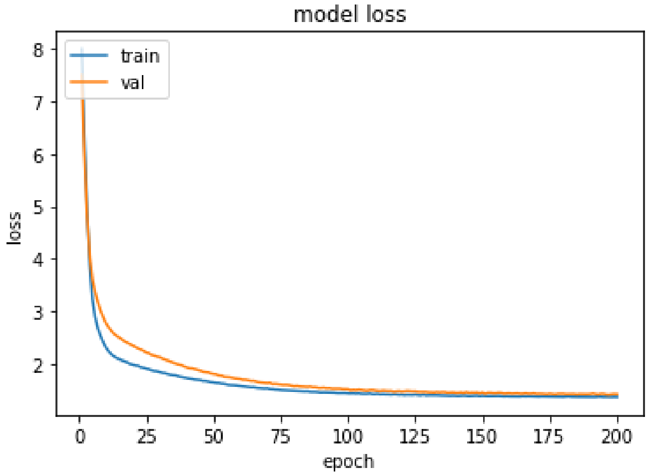

Figure 8. Loss of the model with adam optimizer after 200 epochs with batch size of 1

Then, the model was tested with different optimizer, with ‘sgd’ optimizer. It was again trained with different epoch numbers and batch sizes in the same condition with ‘adam’ optimizer. Under 200 epoch, As batch size increased from 2 to 4, the accuracy also increased from 33% to 39%, but it remained the same over 4. As training epoch increased, the accuracy fluctuates, and so does the loss as shown in figure 11.

While ‘adam’ optimizer shows decreased loss with increasing epoch, ‘sgd’ optimizer shows fluctuating loss with increasing epoch. The loss decreases finally, but its value fluctuates. It is shown in figure 10, figure 11.

Table 1. accuracy at 200 epochs at with adam and sgd optimizer along different batch size

Batch size	Adam epoch 200	sgd epoch 200
1	0.3888888955116272	0.3333333432674408
2	0.4444444477558136	0.3333333432674408
4	0.3888888955116272	0.3888888955116272
8	0.2777777910232544	0.3888888955116272
16	0.1666666716337204	0.3888888955116272

Table 2. accuracy at batch size of 1 with adam and sgd optimizer along different epoch number

Epoch	Adam batch1	sgd batch 1
50	0.3888888955116272	0.3333333432674408
100	0.2777777910232544	0.4444444477558136
200	0.3888888955116272	0.3333333432674408
400	0.3333333432674408	0.4444444477558136
800	0.3333333432674408	0.3333333432674408

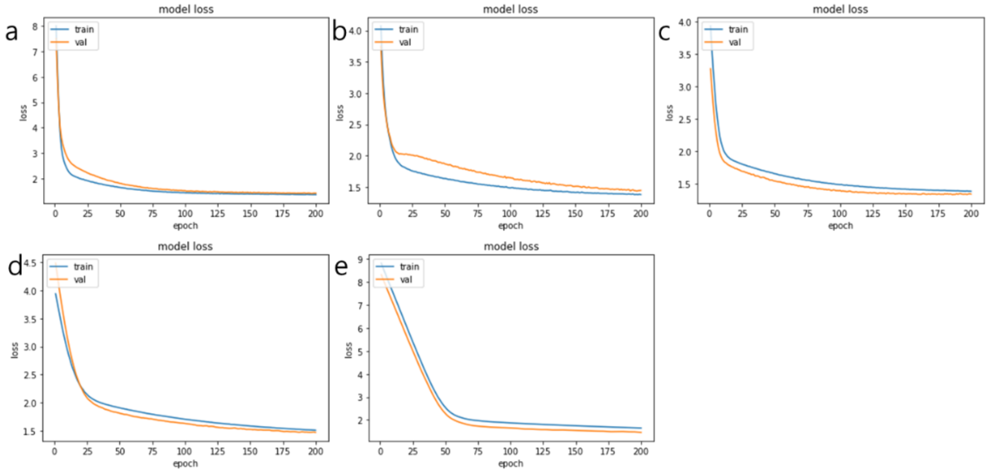

Figure 9. Adam epoch 200 (a) batch size 1 (b) batch size 2 (c) batch size 4 (d) batch size 8 (e) batch size 16

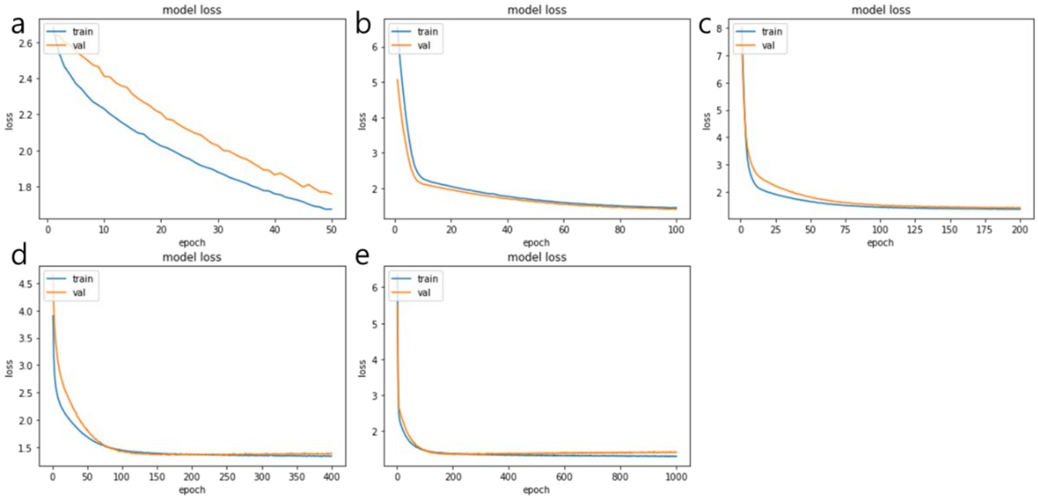

Figure 10. Adam batch 1 (a) 50epoch (b) 100 epoch (c) 200 epoch (d) 400 epoch (e) 1000 epoch

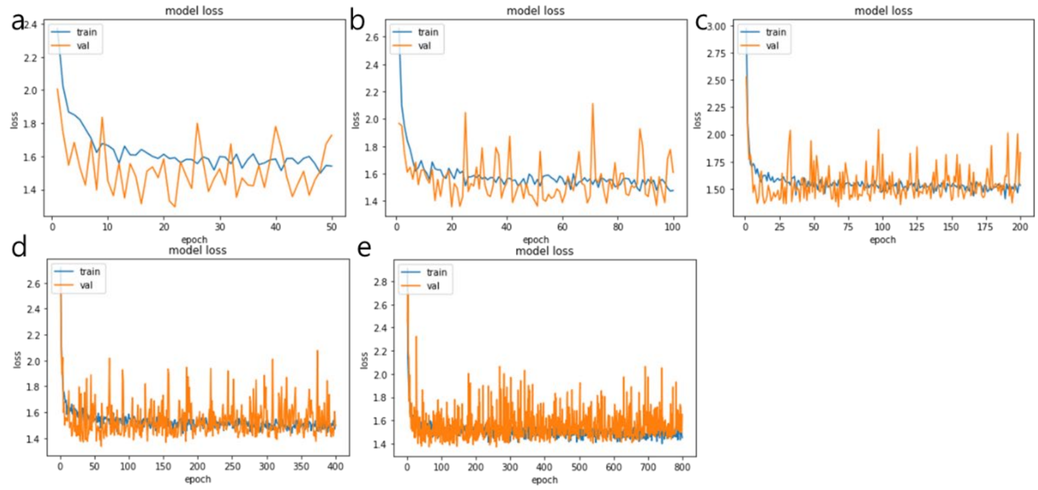

Figure 11. sgd batch 1 (a) 50epoch (b) 100 epoch (c) 200 epoch (d) 400 epoch (e) 1000 epoch

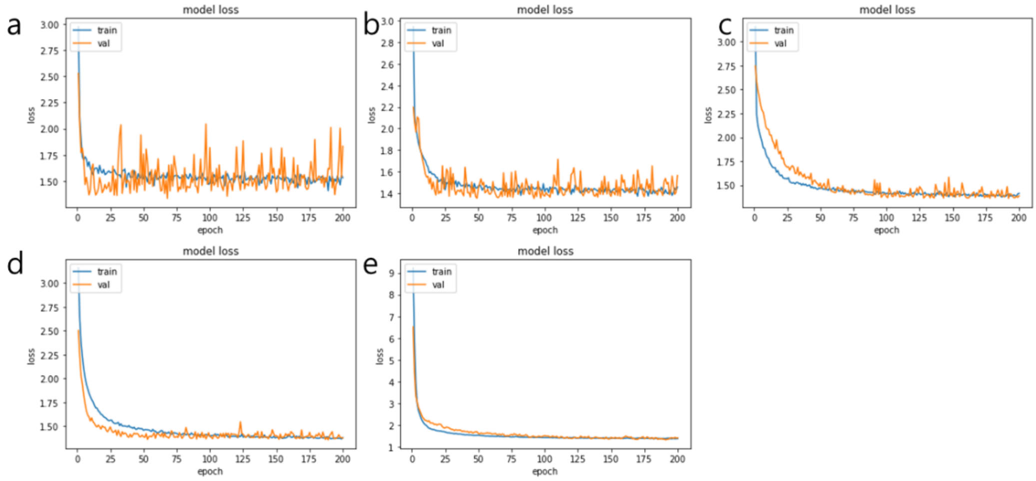

Figure 12. sgd epoch 200 (a) batch size 1 (b) batch size 2 (c) batch size 4 (d) batch size 8 (e) batch size 16

The accuracy of the project was low; even the highest accuracy was lower than 50%. These are some of the estimated reasons for the low accuracy. First, the dataset used in this project had many problems. Due to the small size of the data and low credibility of data, data itself has low meaning. There were responses that were far from the actual merits of each product, and some class had very little dataset. Also, the result would show better performance if the survey asked respondents to check the importance as scaled number, instead of as their hierarchical order. It would have much clear meaning. Also, the actual value of each product should be given as well. Second, I used one-hot encoding to preprocess the data to strip the relevance of the output values. However, there actually are some relationships between the output vectors, but it is neglected in one-hot-encoding.

4. Discussion

The review paper of Praditya·, N. W. (2021) reviews a kind of recommendation system using Hybrid method. Hybrid method is a combined method of Collaborative Filtering and Content-Based Filtering. Content-Based Filtefing cannot recommend new types of items that user have not rated yet. Also, Collaborative Filtering cannot recommend new item if no user has rated them yet. Hybrid method was coined to cover the weakness of these methods. By reviewing materials, authors concluded that their hybrid recommendation system was appliable to any field with context of recommending system, and it does produce efficient results and it can lead to increase in sale if the system is used industrially.

The paper of Munkhdalai, L. (2021) propose a novel partially interpretable adaptive Softmax regression model of class imbalance issue in the application of credit scoring. The proposed model had two main components: linear component which was Softmax regression and non-linear component which was Multi-Layer Perceptron. They first computed linear transformation of input variables and weight parameters of Softmax regression to obtain a logit for each observation. Then, they performed neural network to augment the logit to adapt them for each observation to solve imbalance problem. The linear part explains the fundamental relationship of input and output variables, while the non-linear part serves to improve the prediction performance by identifying the non-linear relationship between features of input. The model showed best performance from their comparison groups.

Paper of Madhiarasan M.(2016) analyzes the impact of hidden neurons estimation in multi layer perceptron neural network. The paper aims to estimate the required number of hidden neurons for wind speed forecasting network. There is possibility of either over or under fitting due to random selection of hidden neurons. When neural network is designed with very few hidden neurons, the network may undergo the problem of under fitting, and when designed with too many hidden neurons, it may undergo over fitting. In both cases, it will decrease the accuracy of the network. Therefore, choosing the adequate number of hidden neurons is important, and complicated process. The multilayer perceptron network comprises and input layer, hidden layer, and output layer. These networks learn linear and nonlinear relationships between the input and output vectors because of the presence of hidden layer neurons and their nonlinear transfer function. The network was tested under various number of hidden neurons, and they developed new criteria for estimating the number. The developed networks achieved better accuracy with reduced error, improved stability and faster convergence.

Introducing MLP, of multilayer perceptron network would have increased the accuracy of my network as well. The current network lacks to compute the relationships between the input feature nor the relationships between the outputs.

5. Reference

1. Praditya·, N. W.,Priscila Yuni, Permanasari, A. E., & Hidayah, I. (2021). Literature review recommendation system using hybrid method (collaborative filtering & content-based filtering) by utilizing social media as marketing. *Computer Engineering and Applications Journal, 10*(2), 105-113.

2. Munkhdalai, L., Ryu, K. H., Namsrai, O.-E., & Theera-Umpon, N. (2021). A Partially Interpretable Adaptive Softmax Regression for Credit Scoring. *Applied Sciences*, 11(7), 3227.

3. Madhiarasan M., Deepa S. N. (2016). Comparative analysis on hidden neurons estimation in multi layer perceptron neural networks for wind speed forecasting. *Artif Intell Rev* 48, 449–471.