Discovering Consumer Preferences from Choices

***Abstract*— In a simulated choice based conjoint analysis study, I compare the performance of softmax regression and neural networks to infer utility from observed choices and predict the top ranked item of a set. I compare the models across the number of observation, and the complexity of the data generation function.**

CONTENTS

I. INTRODUCTION

Consumers make choice according to their preferences. A rational consumer will weight the utility of the alternative options they have and choose the option that maximized their utility. However, in many cases we can only observe the choices that a consumer makes, not their utility function. Therefore in the paper I attempt to infer utility from observed choices.

Utility estimation is a valuable task because an estimate of utility can be used in many applications from dynamic pricing to product line design.

The input to my algorithm is a set of products with know features. I then use softmax regression and neural networks to output a vector of probabilities used to predict the top rank item in the set of products. I explore two main aspects of the models I use: How do they perform when faced with different utility functions, and how do they perform as the number of training observations varies.

In this paper I simulate a rational consumer in a Choice Based Conjoint Analysis study. Code is available at link in references.

A. Joint Project Statement

This project a joint effort with CME291 Masters Research advised by Ashwin Rao. The problem formulation and simulation framework was joint between the classes. The softmax regression derivation was for CME291, but used ideas I learned in CS229. The neural network model and nonlinear utility simulation is for CS229. The comparison between softmax and neural network models is for CS229.

II. RELATED WORKS

The study of consumer decision making is a significant topic of study in fields like Marketing Science and Operations Research. Usually, the utility estimate is a input for another task like pricing, product design, or target market selection. In the marketing, the problem of utility estimation falls under the broad scope of conjoint analysis; including the optimal design of experiments and surveys as well as the estimation tool. In operations research this problem falls into the topic of demand modeling, and the approach I use in the paper would be referred to as a choice based or discrete choice model.

Luce and Tukey in 1964 [4] and 1965 [5] were the first to develop a rigorous mathematical framework for analysis choice. Luce introduced the multinomial logit model (Softmax regression) used in this paper.

Beginning in the late 1970s [8] McFadden contributed significantly to the study of "microecono-metric analysis of choice behavior of consumers who face discrete economic alternatives" [9]; for which he earned the Nobel prize in economics.

The state of the art models for component utility estimation now include generalizations of the MNL model, and new methods. A 2018 paper by Berbeglia , Garassino and Vulcano [10] provides an overview and comparison of several discrete choice methods.

The latent class multinomial logit allows for K classes with class dependant utilities and can be applied to problems where the consumers are heterogeneous. However this model can be challenging to optimize, and the authors above use expectation maximization in their comparisons.

The mixed multinomial logit model allows for "random taste variation, unrestricted substitution patterns, and correlation in unobserved factors over time". It is a generalization of the latent class multinomial logit model with a continuous mixing distributions of the classes. [10]

Finally, Markov Chains can also be applied to this problem where each state is a product in the available set. The transition matrix encodes which product is the favorite, and the probability of transitioning to a different product if the

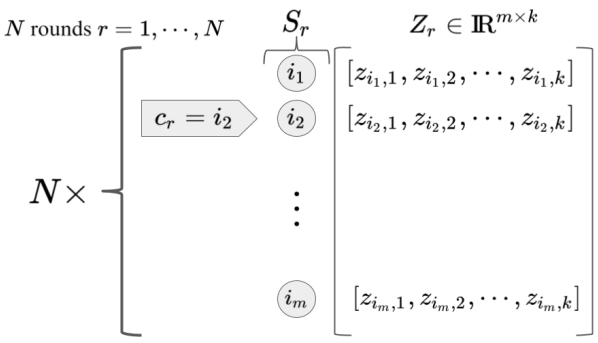

Fig. 1. Data Structure

main product is not available. However there are $O(n^2)$ parameters to estimate, which can cause overfitting. [11]

All the methods above provide powerful generalizations from the softmax regression presented in this paper, but are much harder to estimate. The simple simulation I create in this paper does not need such advanced methods.

III. DATA SET AND FEATURES

A core feature of this paper is that I simulate my data, which allows me to compare the performance of the methods as the number of observations varies without incurring experimental costs. This is valuable because it is a well known limitation that deep learning requires many observations and may not be appropriate for all tasks.

Fig 1 shows a visualization of the data available from the simulation. A buyer is presented with N rounds of products, $r \in 1, \cdots, N$. In each round the buyer has m products to choose from. There are n of products in the simulated universe, $i \in 1, \cdots, n$. Each product has k features. The set of products available in a round r is S_r. For each round the product feature matrix $Z_r \in \mathbb{R}^{m \times k}$ contains the features of the products in that round. The buyer selects a product c_r, which can be represented at a one-hot vector $c_r \in \mathbb{R}^m$. A single simulation is N rounds, each round containing a feature matrix Z_r and a one-hot classification vector c_r.

How does the buyer choose between the products? I simulate a buyer with a utility U for product i_t which has features z_{i_t}.

$$U(i_t) = \theta^T z_{i_t} + g(z_{i_t}) - \rho_{i_t} + \varepsilon_{i_t,r} \tag{1}$$

The utility has a component which is liner in features, $\theta^T z_{i_t}$, and a non linear component $g(z_{i_t})$. The price ρ has a utility of -1; by making this simulation assumption we can measure all utilities in terms of dollars. I split ρ_{i_t} out in (1) for clarity, but for derivations I treat it as a column in Z_r with $\theta_\rho = -1$ without detailed discussion. When modeling I assume that $\theta_\rho = -1$ and subtract price from utility in all cases. Finally there is a random error $\varepsilon_{i_t,r}$ following a Gumbel distribution which varies on each measurement. While not justified here the Gumbel error in utility allows me to use the softmax function for simulating the choice.

Given the deterministic components for m products in set S_r we get a score vector $u^{(r)} \in \mathbb{R}^m$, from which the softmax function yields the probability of selecting product i_t.

$$P(c_r = i_t | S_r, Z_r, g, \theta) = \frac{e^{\theta^T z_i + g(z_i)}}{\sum_{j=1}^m e^{\theta^T z_j + g(z_j)}} = \frac{e^{u_i^{(r)}}}{\sum_{j=1}^m e^{u_j^{(r)}}} \tag{2}$$

Importantly, the probability of selecting a product depends on the other products in the set, but is order invariant. This requires specific consideration when modeling.

There are two sources of randomness in this simulation: which products are selected for S_r, and the randomness in utility (1).

Simulation Specifics: In this simulation $Z_r \sim \text{Unif}(0, 100)$, $\theta \sim \text{Unif}(0, 10)$[1], $\rho_{i_t} \sim N(U(i_t), 4)$, and are fixed across all simulations. A zero product equivalent to a non purchase choice, with 0 utility and feature values of 0, is included without justification in all S_r, and is counted in m. I vary the number of rounds $N \in \{10, 100, 1000\}$.

For the linear utility simulations $g(z_{i_t}) = 0$. In the non linear utility simulations there is a \$50 utility gain (jump) for each feature that is above 50 points.

$$g(z_{i_t}) = \sum_{j=1}^{k} 50 \cdot \mathbb{1}[z_{i_t,j} > 50] \tag{3}$$

IV. METHODS

The machine learning task in this paper is to predict the top ranked product in a set of products. Both methods presented in this paper fit a utility function to compute utility scores, and then use the softmax function to assign probabilities given the scores. Interestingly, we are using classification data to train a regression model, and will be evaluating the performance using ranking metrics. An important limitation is that the models must be permutation invariant, due to the nature of the problem. This rules out SVMs and most simple decision tree methods.

A. Oracle

Because the data is simulated, I can use an *Oracle* model which knows the true utility for each product $U(i_t)$. The Oracle model will also use the softmax function to generate probabilities as in (2) The performance of the Oracle model will be the baseline to judge the difficulty of the tasks, and will be the baseline model.

B. Softmax Regression

Softmax regression, also knows as multinomial logistic (MNL) regression is a generalization of logistic regression to the case where we want to handle multiple classes. [1]

[1]this is a footnote, include copies of theta

I develop a gradient descent method to maximize the likelihood.

$$
\begin{aligned}
L(\theta) &= P\left[(c_1|S_1)\bigcap\cdots\bigcap(c_r|S_r)\middle|\theta,Z\right] \\
&= \prod_{r=1}^{N}\prod_{i\in S_r}\mathbb{1}\left[c_r=i\right]P(c_r=i|S_r) \\
l(\theta) &= \sum_{r=1}^{N}\sum_{i\in S_r}\mathbb{1}\left[c_r=i\right]\log\left(P(c_r=i|S_r)\right) \\
&= \sum_{r=1}^{N}\sum_{i\in S_r}\mathbb{1}\left[c_r=i\right]\log\left(\frac{e^{\theta^T z_i}}{\sum_{j=1}^{m}e^{\theta^T z_j}}\right) \\
&= \sum_{r=1}^{N}\log\left(\sigma_{c_r}(Z_r,\theta)\right)
\end{aligned}
$$

I use the notation of the score vector $u^{(r)}$ with an element indexed as $u_j^{(r)}$, use σ as notation for the softmax function.

$$
\sigma_i(Z_r,\theta)=\sigma_i(u^{(r)})=\sigma_i=\frac{e^{u_i^{(r)}}}{\sum_{j=1}^{m}e^{u_j^{(r)}}}
$$

The derivative of the softmax function can be written using the Kronecker delta function δ_{ij}. Where $\delta_{ij}=1$ if $i=j$, and $\delta_{ij}=0$ otherwise so that $\frac{\partial\sigma_{c_r}}{\partial u_s^{(r)}}=\sigma_{c_r}(\delta_{c_r s}-\sigma_s)$ Here I find the update rule for a single element d of predicted θ

$$
\begin{aligned}
\frac{\partial l(\theta)}{\partial\theta_d} &= \sum_{r=1}^{N}\frac{\partial\log(\sigma_{c_r}(u^{(r)}))}{\partial\sigma_{c_r}(u^{(r)})}\cdot\frac{\partial\sigma_{c_r}(u^{(r)})}{\partial\theta_d} \\
&= \sum_{r=1}^{N}\frac{1}{\sigma_{c_r}}\sum_{p=1}^{m}\left(\sigma_{c_r}(\delta_{c_r s}-\sigma_s)\frac{\partial u_{c_r}^{(r)}}{\partial\theta_d}\right) \\
&= \sum_{r=1}^{N}\frac{1}{\sigma_{c_r}}\cdot\left(\sigma_{c_r}Z_{c_r d}-\sum_{p=1}^{m}(\sigma_{c_r}\sigma_p)Z_{pd}\right) \\
&= \sum_{r=1}^{N}\left(Z_{c_r d}-\sum_{p=1}^{m}\sigma_p Z_{pd}\right)
\end{aligned}
$$

This update rule for softmax regression uses information from all products in the set S_r, the predicted probabilities σ_i, and the product that was selected c_r to update $\hat{\theta}$. It is equivalent a single layer linear neural network with no activation function and cross entropy loss.

C. Neural Networks

To create a deep learning model workflow that is invariant to the order products are presented in, I trained a neural network to estimate the utility of each product. Figure 2 shows a visualization of the forward propagation step. The red bar shows how a products feature vector z_{i_t} is transformed to utility and then into loss. After calculating the predicted utility of each product, the Softmax function generates probabilities for each round and S_r, and I find the Negative Log Likelihood Loss [2] shown in equation (4).

$$
\text{NLL loss}(c_r,u^{(r)})=-\log(\sigma_{c_r}(u^{(r)})) \tag{4}
$$

I used a fully connected neural network with three hidden layers with bias, all with reLU activation functions.

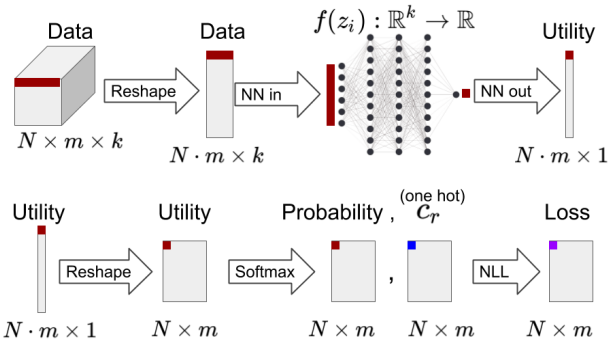

Fig. 2. NN Structure

- Input Layer: k neurons, the product feature values
- First Hidden Layer: 100 neurons
- Second Hidden Layer: 100 neurons
- Third Hidden Layer: 100 neurons
- Output Layer: 1 neuron, the utility score

I recognize that the number of layers and neurons in each layer is are important parameters to tune, however I did not attempt that tuning in this paper. I use the network above in all cases to have a consistent comparison and save on computational time.

V. Results

There are two dimensions of interest in the comparison between the softmax regression and neural network:

(i) How much data do the models need?
(ii) How well do the models perform if the underlying utility function is complex?

To test the data requirement I train each model with $N=\{10,100,1000\}$ observations and compare them. We should expect the neural network to perform worse than softmax regression when there are few observations, because the high number of parameters will cause overfitting. The linear model underlying softmax regression will give it the advantage on small training sets.

To test the performance when facing more complex tasks, I simulate a consumer with a nonlinear utility function, as explained in Section III. In this case the non-linear part of utility (3) is a indicator function that provides a jump discontinuity of additional utility when an attribute is >50. The softmax model should struggle to find and fit this type of nonlinearity. The neural network model should be better able to fit the discontinuity with the nonlinear activation functions and higher number of parameters.

Metrics: The task is to predict which product the consumer will choose from a set, by inferring the utility. Ranking quality metrics are most applicable, because the consumer will choose the highest ranking product by their utility. The metrics reported here are the Top 1 Recall (5), and the Mean Reciprocal Rank (6). Top 1 Recall measures the percentage of the time the predicted top item is the observed selected item c_r. Mean Reciprocal Rank is the average reciprocal rank of the observed selected item c_r; as score of 1 is perfect.

$$\frac{1}{N}\sum_{r=1}^{N}\mathbb{1}\left[\arg\max_{i}\sigma_i(u^{(r)})=c_r\right] \tag{5}$$

$$\frac{1}{N}\sum_{r=1}^{N}\frac{1}{\text{rank}(c_r)} \tag{6}$$

Irreducible Error: The utility equation (1) has an error term $\varepsilon_{i_t,r}$, simulating the random portion of a consumer's preferences. Any prediction will include this variance in preferences. Thus I present an Oracle model that knows the utility of the products perfectly as an upper bound on the metrics.

Test Train Split: As discussed above all data is simulated, so I do not suffer external data constraints. The training set is a collection of N rounds, each round containing a product feature matrix Z_r and observed choice c_r. I test all models on a test set of $N = 10,000$ observations. The test set does vary between model training, but the Oracle results show that the variation is low.

Training Methodology: I used all data in a single batch for all models. I used a learning rate of 0.001 for the softmax model and 0.00002 for the neural network; these learning rates were selected by "dialing it in" so that the training loss monotonically decreased.

A. Linear

This sections considers when the buyer derives a utility from consumption that is linear in the features of the products. Table I reports the results for the softmax regression, II shows the results for the neural network, and 3 visualizes the Top 1 Recall across training data regimes

Findings:

(i) Observation: At $N = 10$ Both models overfit, but the softmax regression performs better on the test set.

(ii) Observation: At $N = 100$ The softmax regression does not overfit (training and test accuracy close), but suffers from bias. However the neural network still shows evidence of over fitting.

(iii) Conclusion: Hypothesis confirmed, Softmax regression performs better in low data regimes.

(iv) Observation: At $N = 1000$ Both softmax and neural network have minimal differences between training and test recall. However, the neural network suffers from bias where the softmax model achieves a level of accuracy that is close to optimal.

(v) Conclusion: Simple linear models perform better than more complex ones if the underlying model is linear.

TABLE I

SOFTMAX VS LINEAR UTILITY

	Recall			Mean Reciprocal Rank	
N	Oracle	Train	Test	Oracle	Test
10	.6139	.7668	.4087	.5324	.4896
100	.6181	.6067	.5980	.5301	.5258
1000	.6172	.6153	.6157	.5333	.5333

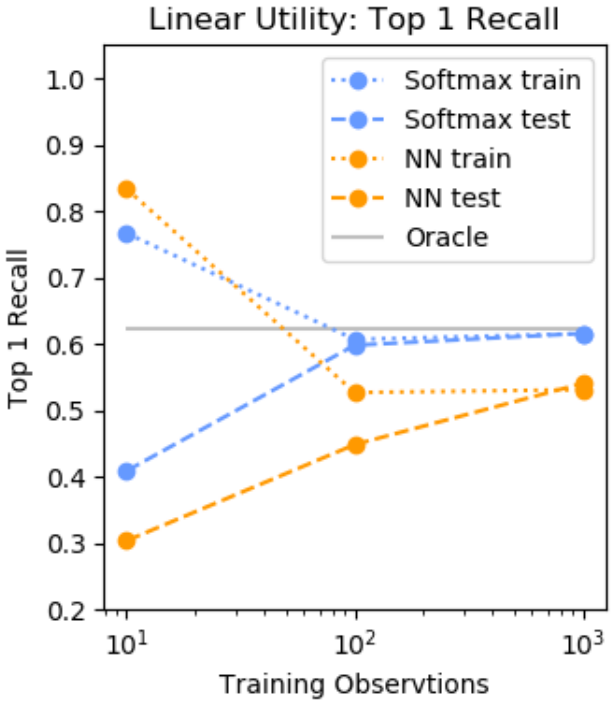

Fig. 3. Linear Recall

TABLE II

NN VS LINEAR UTILITY

	Recall			Mean Reciprocal Rank	
N	Oracle	Train	Test	Oracle	Test
10	.6139	.8333	.3035	.5324	.4846
100	.6181	.5266	.4482	.5301	.5069
1000	.6172	.5310	.5404	.5333	.5223

B. Non Linear

This sections considers when the buyer derives a utility from consumption that is not linear in the features of the products, but also includes a sum of indicator functions 3. Table III reports the results for the softmax regression, table IV shows the results for the neural network, and figure 4 visualizes the Top 1 Recall across training data regimes. Note that the performance of the Oracle model is much higher than the linear case; this is because the magnitude of the utilities under the non-linear function are much higher than the linear case, while the variance of the error term $\varepsilon_{i_t,r}$ does not change. This does not hinder our interpretation of the results.

Findings:

(i) Observation: At $N = 10$ Both models overfit, but the neural networks overfits so that softmax regression has better test performance.

(ii) Observation: At $N = 100$ Both models overfit, but the neural network performs better than the softmax. This is different from the linear utility simulation.

(iii) Observation: At $N = 1000$ Both softmax and neural network have minimal differences between training

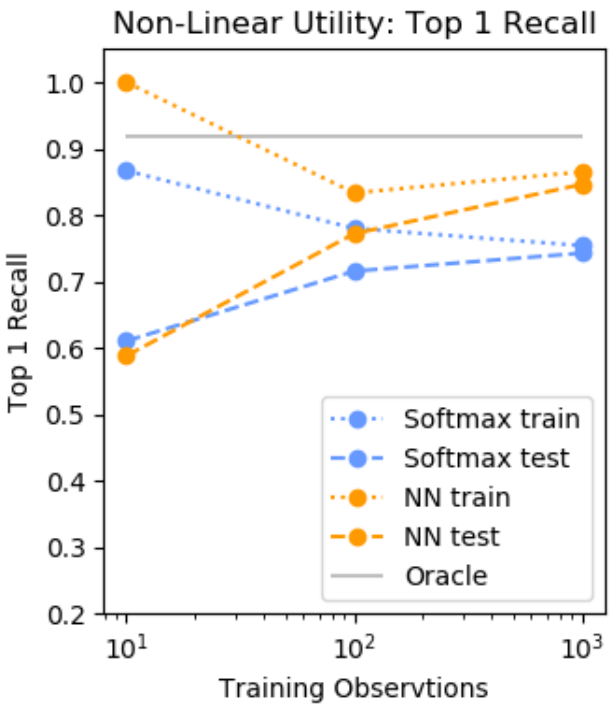

Fig. 4. Non-Linear Recall

and test recall. The SM suffers from bias where the NN achieves higher accuracy. However neither model achieves optimal prediction accuracy.

(iv) Conclusion: The Neural network finds non linear trends even in moderate data regimes. I would expect it to achieve optimal performance with more data.

TABLE III
SOFTMAX VS NON LINEAR UTILITY

	Recall			Mean Reciprocal Rank	
N	Oracle	Train	Test	Oracle	Test
10	.9178	.8667	.6103	.6493	.6062
100	.9145	.7800	.7155	.6465	.6249
1000	.9198	.7540	.7427	.6492	.6356

TABLE IV
NEURAL NETWORK VS NON LINEAR UTILITY

	Recall			Mean Reciprocal Rank	
N	Oracle	Train	Test	Oracle	Test
10	.9178	1.00	.5879	.6493	.6004
100	.9145	.8333	.7719	.6465	.6399
1000	.9198	.8650	.8462	.6492	.6495

VI. CONCLUSION

In this project I compared softmax regression and neural networks based on their performance fitting two functions, and the number of training observations necessary. This comparison occurred in an interesting context of estimating utility from a limited observation of ranking. I confirmed the basic hypotheses that neural network models require more data than softmax regression, and that softmax regression works best when fitting linear models. It was good to see how those two factors interact to see which model fits best. While working with the neural network was interesting, it is not interpretative which is likely unacceptable in the context of marketing studies where people want to know why something is good.

Future Work: There are several basic but computationally intensive steps I would have liked to do but lacked the time. First I would have tune the hyper-parameters of the neural network like layer depth and number of nodes to reduce overfitting in low data regimes. I would also have explored feature engineering (perhaps kernalized the linear features) to introduce non-linearity the linear softmax regression.

The next step is to build a more complex situation, or work with real experimental data when I need the state-of-the art-methods I explain in section II. Finally, gradient decision trees may provide a permutation invariant method with explainable interpretations for the coefficents.

Code available at: https://drive.google.com/file/d/1gwYbEZ2w9_XUeg6_3AqZU84aSIWVk33W/view?usp=sharing

REFERENCES

[1] http://deeplearning.stanford.edu/tutorial/supervised/SoftmaxRegression/

[2] Paszke, Adam, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. "Automatic differentiation in pytorch." (2017).

[3] V.R. Rao, Applied Conjoint Analysis, DOI 10.1007/978-3-540-87753-0_1, © Springer-Verlag Berlin Heidelberg 2014

[4] Luce, R. D., Tukey, J. W. (1964). Simultaneous conjoint measurement: A new type of fundamental measurement. Journal of mathematical psychology, 1(1), 1-27.

[5] Luce, R. Duncan. "Individual choice behavior." (1959).

[6] Green, Paul E., and Vithala R. Rao. "Conjoint measurement-for quantifying judgmental data." Journal of Marketing research 8, no. 3 (1971): 355-363.

[7] Kruskal, Joseph B. "Analysis of factorial experiments by estimating monotone transformations of the data." Journal of the Royal Statistical Society: Series B (Methodological) 27, no. 2 (1965): 251-263.

[8] McFadden, Daniel. "Econometric models of probabilistic choice." Structural analysis of discrete data with econometric applications 198272 (1981).

[9] McFadden, Daniel. "Economic choices." American economic review 91, no. 3 (2001): 351-378.

[10] Berbeglia, Gerardo, Agustín Garassino, and Gustavo Vulcano. "A comparative empirical study of discrete choice models in retail operations." Available at SSRN 3136816 (2018).

[11] Blanchet, Jose, Guillermo Gallego, and Vineet Goyal. "A markov chain approximation to choice modeling." Operations Research 64, no. 4 (2016): 886-905.