Deep Learning for Discovering New Product Opportunities

by

Santiago Perez

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Febuary 2017

Signature redacted

Author ....
Department of Electrical Engineering and Computer Science
Febuary 1, 2017

Signature redacted

Certified by. .....
Glen Urban
Professor of Marketing, Emeritus
Thesis Supervisor

Signature redacted

Accepted by ... .
Christopher J. Terman
Department of Electrical Engineering and Computer Science

Deep Learning for Discovering New Product Opportunities

by

Santiago Perez

Submitted to the Department of Electrical Engineering and Computer Science
on Febuary 1, 2017, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Modeling customer preferences over multivariate attributes has long been an endeavor of marketing research. We provide an updated approach which can learn customer preferences for complex products with multiple multivariate attributes using modern Deep Neural Networks. In turn we outline approaches for framing managerial questions in the form of inference problems. With our empirical application to product identification in credit cards, we conclude Deep Learning results in significantly better performance than the state of the art. Our approach is scalable to Big Data and can derive superior predictive power from the inexpensive unstructured data exhaust of internet commerce.

Thesis Supervisor: Glen Urban
Title: Professor of Marketing, Emeritus

Acknowledgments

There are many people I must thank for making my time as a graduate student a wonderful experience.

First I must thank my advisor Glen Urban who through dozens of meetings and calls molded my career path towards the intersection of computer science and marketing. Glen's patience, foresight, and intuition made this work possible. When I pitched ideas, Glen would distill their essence and uncover a more fundamental story placed in a wider context. I am grateful to have learned under his guidance.

I wish to thank Paramveer Dhillon for guiding me through the statistical evaluation section of my thesis. I am grateful for his perseverance in explaining statistics to me over and over until I achieved understanding. It has also been a distinct pleasure collaborating with Hiro and Hector.

I am thankful to Suruga Bank who supported my research. I admire the partnership they have forged with the institute.

Special thanks to Magid Abraham for seeing the potential in our research and connecting us with his great company. I extend my thanks to ComScore: especially Rebecca Presor, Chip Garate, Brett Baggett, Brian Beiles, and Derek Wolff for their hard work towards providing us stellar data.

And to my parents, who fled a war weary nation and abandoned all familiarity left in their lives to give me the opportunity to pursue my dreams. It is in no small part the desire to vindicate their sacrifices and make them proud that drives my ambitions.

Contents

List of Figures

Chapter 1

Introduction

In the last decade, through the breakthroughs of AlphaGo, game A.I.s have demonstrated the promise of optimal action by approximating reward functions with Deep Neural Networks [1]. The defeat of the world champion in one of the most difficult unconstrained games, Go, serves as a premonition of the disruption of A.I. in business. If A.I.s can learn to act optimally in the setting of games, how long until they learn to do so in business?

Deep Learning is a Machine Learning technique that uses Big Data to train Neural Networks with many layers. It represents a major advance in modeling capability and power. However Machine Learning has already contributed to the art of Internet Marketing. Bayesian Inference powers Google's targeted advertising engine: adsense [3]. Targeted advertising ensures marketing efforts are directed optimally and raises awareness while minimizing cost. By modeling customers with machine learning, we can filter our audience efficiently. Advertisement Morphing [2] tailors the message to the individual and considerably boosts conversion. By once again modeling customers' cognitive styles we can deliver the most compelling message possible. On the other hand, all this work assumes the product is predefined; the optimization of products has received little attention. This thesis seeks to contribute to the problem of product development by applying Deep Learning, a major recent development in Machine Learning.

We propose that the passive data emissions of the internet economy betray the

preferences and desires of the market. We will tap into the inexhaustible unstructured click data and consumer browsing data and demonstrate its value in predicting consumer preferences. By using less unstructured, more abundant data we demonstrate Big Data and Deep Learning techniques result in richer more sophisticated models.

The trend in Machine Learning and Deep Learning in particular is the use of over-parameterized models with increasingly high data critical mass to distill statistical strength for learning. What distinguishes the work presented here from the state of the art is we provide a framework for scalability; scaling the current marketing research process, to fit the next generation of models is prohibitively costly.

1.1 Contributions and Goals

In this work we apply Deep Learning to the problem of product opportunity identification. By applying Deep Neural Networks to a large credit card dataset, we confirm that Deep Neural Networks have superior prediction accuracy than the state of the art. We then develop a paradigm for answering managerial questions and apply it to three classes of questions through an empirical case study:

- **Inference of Customer Feature Vector given Product Attributes:** What are the target demographics for a particular card?
- **Inference of Product Attributes given Customer Feature Vector(s):** What types of cards attracts low income, low credit score consumers vs. high income high credit score consumers?
- **Competitive Market Simulation:** Given a cost model, what cards would be most profitable to introduce?

1.2 Outline

After reviewing background material in Chapter 2, we will introduce the ComScore Dataset in Chapter 3. Learning the preference model is covered in Chapter 4. We

then in Chapter 5 discuss the three general inference capabilities of this framework and showcase an example simulation for each. We end our work in Chapter 6 with a discussion on implications and an outline for future work.

Chapter 2

Deep Learning Background

Machine learning before the resurgence of neural networks involved hand-engineering representations and input features. One would then train a classification layer built on hand-engineered representations and features, which can be thought of as optimizing weights to make the best final prediction.

Deep learning can be thought of jointly learning a representation that optimizes classification accuracy. It attempts to both learn good features, across a hierarchy of ascending complexity and abstraction, and a final classification prediction.

In this chapter we will review the breakthroughs that sprung the resurgence of deep learning. We will also review the general neural network based models, and cover the basic stochastic gradient descent algorithm. We end this chapter with a discussion on Markov Chain Monte Carlo methods..

For more material on Deep Learning, I strongly recommend consulting Deep Learning by Goodfellow et al. [4].

2.1 Why Now?

Deep learning simultaneously solves many of the maladies which ailed traditional machine learning.

2.1.1 Learning Representations

Traditional machine learning often involved handcrafting features and producing artisanal representations which were often incomplete and over-specified. The task of designing representations is often laborious and difficult. Data in the wild often tend to be unstructured and unintelligible to the eye. In this work we shall encounter URLs which seemingly have no significance, but Deep Learning is able to extract significant meaning from them. Deep learning provides an automated way of learning representations customized for a particular prediction problem.

2.1.2 Distributed Representations

Symbolic representations such as a table, or database are often sparse and models tend to lead to models which generalize poorly. Another way to see the issue with these representations is one of dimensionality, because the data is very sparse, models often overfit to the training data. Classical solutions involve either feature engineering or linear transformations. Deep learning finds subsymbolic continuous representations which capture subtle similarities and differences between datapoints.

2.1.3 Learning Multiple Levels of Representation

Deep learning models such as convolutional neural networks [12] trained on images learn similar levels of representations as the human brain. The first layer learns simple edge filters, the second layer captures primitive shapes and higher levels combine these to form objects. Deep models are able to capture nonlinear decision boundaries in the data whereas classical models are restricted to the rigid class of hyperplane decision boundaries.

2.1.4 The Rise of Big Data

Deep models have far more parameters and thus require far more data to train than Multinomial Logistic Regressions. Neural networks have been around of decades.

Until 2006[10], however deep neural networks were often out performed by shallow hand-engineered models. Neural networks require Big Data to gain statistical strength in parameter inference. When the newly minted ImageNet [9] dataset's size reached a critical mass to fit Deep Neural Networks, the performance sprung a new wave of enthusiasm in deep models [11].

2.1.5 A different kind of Moore's Law

The recent years have seen a substantial slow down in CPU clock cycle improvements. We are simply hitting hard limits of what silicon can achieve. However Moore's Law has been preserved through parallelism — we continue to increase our computational power not through speeding up cores but by adding more of them. Graphics Processing Units (GPUs) are computer processors with thousands of cores. Machine learning researchers have hijacked the graphics technology for extremely parallel models. The revival of Deep Learning with Big Data was largely unlocked through the use of GPU accelerated neural networks [11].

2.2 Neural Network

A neural network is made of multiple linear transformation layers with a nonlinearity in between. For our work we choose the Restricted Linear Unit (RELU) [16] as our nonlinearity. The RELU layer is simply the identity function restricted to take positive values.

$$x_{l+1} = f(W_l x_l + b_l)$$

The RELU function is defined below.

$$RELU(x) = x\mathbb{I}\{x > 0\}$$

The parameters for our network consists of a transformation matrix W_l and bias

b_l per layer. Each neural network architecture encodes a family of functions called the parameter space.

2.3 Stochastic Gradient Descent

The learning process is framed as a optimization problem. We use a loss function, a measure of goodness of fit, to search for an optimal model in the parameter space.

The squared error loss function (MSE) is commonly used. We train our neural network using batch gradient descent. In detail, for a single training input output example pair (x, y), we define the cost function with respect to that single example to be:

$$J_{W,b}(x, y) = \frac{1}{2}||x_N - y||^2$$

On a set of m training examples the MSE loss function becomes

$$J_{W,b}(x, y) = \frac{1}{2}\sum_{i=1}^{M}||(x_i)_N - y_i||^2$$

We often combine various loss functions that are tailored to the problem needs. Some loss functions ensure generalizability while others ensure predictive accuracy.

Stochastic Gradient Descent (SGD) updates our parameters using the gradient of the loss function. Two important assumptions are made about our loss function in order to use SGD. The first is that the loss function is continuous, namely parameters that are similar should perform similarly. The second is that the loss function is differentiable with respect to the parameters, we must provide this derivative in order to update our parameters. The update equations, where α is our step size, are as follows

$$W_l = W_l - \alpha \frac{\partial J_{W,b}(x,y)}{\partial W_l}$$
$$b_l = b_l - \alpha \frac{\partial J_{W,b}(x,y)}{\partial b_l}$$

Sometimes momentum and learning rate decay is added to the equations to improve resilience towards local minima. Learning rate decay is loosely equivalent to simulated annealing in the Markov Chain Monte Carlo [17]. The converged behavior of SGD is a random walk around the local minima whose distance (energy in the Hamiltonian sense) is proportional to the learning rate. Therefore we decay the learning rate so that the oscillations of this random walk decrease and we get arbitrarily close to the minima. The update equations for this established version of the SGD algorithm are:

let $\nu = momentum$

let $\eta = decay$

$$W_l = \nu W_l + (1-\nu)\left(W_l - \alpha \frac{\partial J_{W,b}(x,y)}{\partial W_l}\right)$$
$$b_l = \nu b_l + (1-\nu)\left(b_l - \alpha \frac{\partial J_{W,b}(x,y)}{\partial b_l}\right)$$

Every full pass through the data

$$\alpha = (1-\eta)\alpha$$

2.4 Regularization

Regularization is a technique that improves predictive accuracy by augmenting the loss function with an additional loss in order to impose "Occam's Razor" or a bias for simpler solutions on the model. Various penalties result in different types of solutions:

- **l2 penalty:** improves how the generalizes to unseen data by "decaying" unused

weights.

- **l1 penalty:** biases model parameters to be sparse, thereby improving the interpretability of the solution, as well as improving the predictive accuracy.

In a Bayesian framework various regularizations are equivalent to imposing a prior distribution on the solutions. In our work we use a L_2 regularization loss, which assumes the model parameters are sampled from a Normal Prior distribution. This simply adds the L_2 norm of the parameter vector to our loss function, scaled by a training parameter called the weight decay. This encourage our learning algorithm to find simpler models with more predictive power.

2.5 Classification

We have described the general approach to training Deep Neural Network Representations. Deep Learning is an augmentation of established classification techniques with nonlinear representations. Classification is the problem of prediction an outcome or class from data. In computer vision we want to classify images correctly based on their content; In our setting we seek to predict the choices consumers make (the card they select). When we are tasked with a classification task, we add a classifier layer to our representation layers. This is the only distinguishing feature of Deep Neural Networks, the insight of Deep Learning was to transform the input into a nonlinear representation before classification. The two are learned at once, and features are found in the data automatically.

The learned representations are nonlinear, and take the form of high dimensional manifolds. The representation layers generalize the top layer linear classifier to the class of nonlinear classifiers. The representation layers enable Deep Learning to capture subtle nonlinearities in the data.

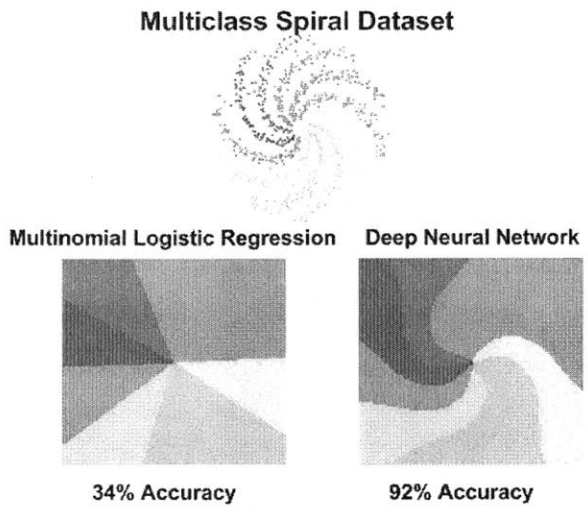


Figure 2-1: We see the class decision boundaries (right) of a Multinomial Logistic Regression and a Deep Neural Network on a pinwheel dataset (left)

Nonlinearities

A quintessential hallmark of models before Big Data is linearity. The scarcity of data and computational power restricted the class of tractable models to Linear models. Although linearity leads to computational efficiency and simpler math, it is a very harmful assumption to make about the underlying structure of the data. Assuming decision boundaries are hyperplanes on the input data leads to very constrained, restricted models which perform poorly. The reason behind this is geometric: In two dimensional space it is hard for data to be arranged in a line. In three dimensional space it is even harder for things to be arranged in a plane. As the dimensionality grows, from two to three, the space to near-the-plane ratio increases. As the dimensionality increases more and more, area expands exponentially faster and points grow farther and farther apart. This issue is referred to as the curse of dimensionality and is the strongest motivation for using nonlinear models as the dimensionality of data increases. The larger the dimensionality of data, the larger the error of a linear fit.

Deep Learning learns a representation that warps the input space such that the top linear classification model's decision boundaries are nonlinear in the input space. Consider the classic pinwheel example. We can see that the Multinomial Logistic

Classifier is restricted to linear decision boundaries on the input space. However the Deep Neural Network untangles the input space until the optimal decision boundaries at the top layer are linear. The top layer sees a linearly separable problem, however, when we look at the decision boundaries in the input space we see a nonlinear classifier. This insight is what propels Deep Learning ahead in performance. It automatically learns a representation of the data that simplifies the classification problem.

We will now discuss the top classification layer.

2.6 Multinomial Logistic Model

Multinomial Logistic regression (or Softmax regression) is a generalization of logistic regression handling multiple classes. In logistic regression we assumed that the labels were binary: $y_i \in \{0, 1\}$. We used such a classifier to distinguish between two kinds of classes. Multinomial Logistic regression allows us to handle $y_i \in \{1, \ldots, K\}$ where K is the number of classes we wish to separate.

Note that this model is referred to as the Softmax classifier in the Deep Learning community — the two are equivalent.

Recall that in logistic regression we have a training set $\{(x_1, y_1), \ldots (x_m, y_m)\}$ of m labeled examples, where the input features are $x \in \mathbb{R}^n$. With logistic regression, in our binary classification task, so our labels took the values $y_i \in \{0, 1\}$, so our hypothesis took the form of

$$h_\theta(x) = \frac{1}{1 + \exp(-\theta^T x)}$$

and model parameters θ were trained to minimize the cost function

$$J_\theta(x, y) = -\left[\sum_{i=1}^{m} y_i \log h_\theta(x_i) + (1 - y_i) \log(1 - h_\theta(x_i))\right]$$

The Multinomial Logistic regression generalizes logistic regression to K classes. Our hypotheses take the form of

$$h_\theta(x) = \begin{bmatrix} P(y=1|x;\theta) \\ \vdots \\ P(y=K|x;\theta) \end{bmatrix} = \frac{1}{\sum_j^K \exp(\theta_j^T x)} \begin{bmatrix} \exp(\theta_1^T x) \\ \vdots \\ \exp(\theta_K^T x) \end{bmatrix}$$

Note that θ is now an $n \times K$ matrix, and each θ_i is a column of length n.

Our generalized Multinomial Logistic cost function now takes the form:

$$J_\theta(x,y) = -\left[\sum_{i=1}^{m}\sum_{k=0}^{K} \mathbb{I}\{y_i = k\} \log\left(h_\theta(x)_k\right)\right]$$

2.7 Markov Chain Monte Carlo

Markov Chain Monte Carlo methods allow us to sample from a probability distribution (posterior) that incorporates our beliefs without looking at the data (prior) and our beliefs from the data (likelihood). This is extremely difficult to do in general — it is known to be NP-Hard [6]. If we could perfectly sample from the posterior probability distribution we could automatically solve every practical problem in computer science.

Why is this so hard? Remark Bayes Formula:

$$P(\theta|x) = \frac{P(x|\theta)P(\theta)}{P(x)}$$

Its numerator, the likelihood and prior are easy to compute and have a closed form. Its denominator is extremely difficult to compute and requires an intractable

integral, the law of total probability:

$$P(x) = \int_{\theta} P(x|\theta)P(\theta)d\theta$$

Markov Chain Monte Carlo (MCMC) allows us to approximate the posterior distribution by creating a random walk-like process which converges in distribution to our posterior. We will focus on the simplest of the MCMC techniques — the Metropolis-Hastings algorithm.

2.8 Metropolis-Hastings

The Metropolis-Hastings Algorithm is essentially a random walk that begins at a random location x_i. We sample a new x_j in any arbitrary manner. This could be from a uniform distribution, or it can be a minor perturbation of x_i. We accept proportional the ratio of the posterior of x_j and x_i. We then repeat the algorithm indefinitely. The acceptance criterion ensures we stay at samples with larger posterior probability for longer, and it warps our sampling distribution into the posterior. A full treatment of the Metropolis-Hastings Algorithm can be found in Hastings et al. [5].

The Full Metropolis Hastings Algorithm

1. For $n = 1, 2, \ldots, N$, set current state to x_i and do
2. Sample from the proposal distribution $q(x_j|x_i)$.
3. Evaluate $\alpha = \frac{q(x_i|x_j)p(x_j)}{q(x_j|x_i)p(x_i)}$
4. Accept the proposal sample, x_j with probability $\min[1, \alpha]$ otherwise keep the current sample x_i as the new sample.

Running the Metropolis Hastings Algorithm produces an approximate series of samples from the posterior distribution. Usually this involves running the Algorithm

for a burn in period of a few hundred samples before we begin to record samples. A major difficulty of MCMC methods is identifying when when the algorithm has converged to the posterior distribution. This is an active research problem.

2.8.1 Stochastic Search

Stochastic Search seeks to optimize a reward function by searching for solutions using MCMC. The algorithm simply runs a MCMC algorithm to gather a large set of samples. However, we modify the acceptance criterion to be the ratio of the two reward functions to favor movements into a high reward regions. The algorithm returns the sample which resulted in the most reward along the random walk.

Chapter 3

ComScore Data Analysis

In the previous chapter we described the theory of deep learning. Before we can apply it to a marketing decision in the context of credit cards, we will introduce the comScore Credit Card Acquisition dataset. In this chapter we outline comScore's methodology, describe the data, and discuss statistics about the dataset.

3.1 Data

ComScore is an American Marketing Data and Analytics company that curates analytics for the worlds largest companies; ComScore helps Enterprises make informed decisions regarding their Marketing efforts and innovations. The ComScore Credit Card Acquisition dataset contains data for 55,000 applications to 15 major bank credit card sites, with an average of 28 browsing history URLs per person. Over 30% of the panelists visit more than one site. Panelists demographics are included as well as the details of the credit card applications that are accepted (e.g. APR, Reward Rate, Amount)

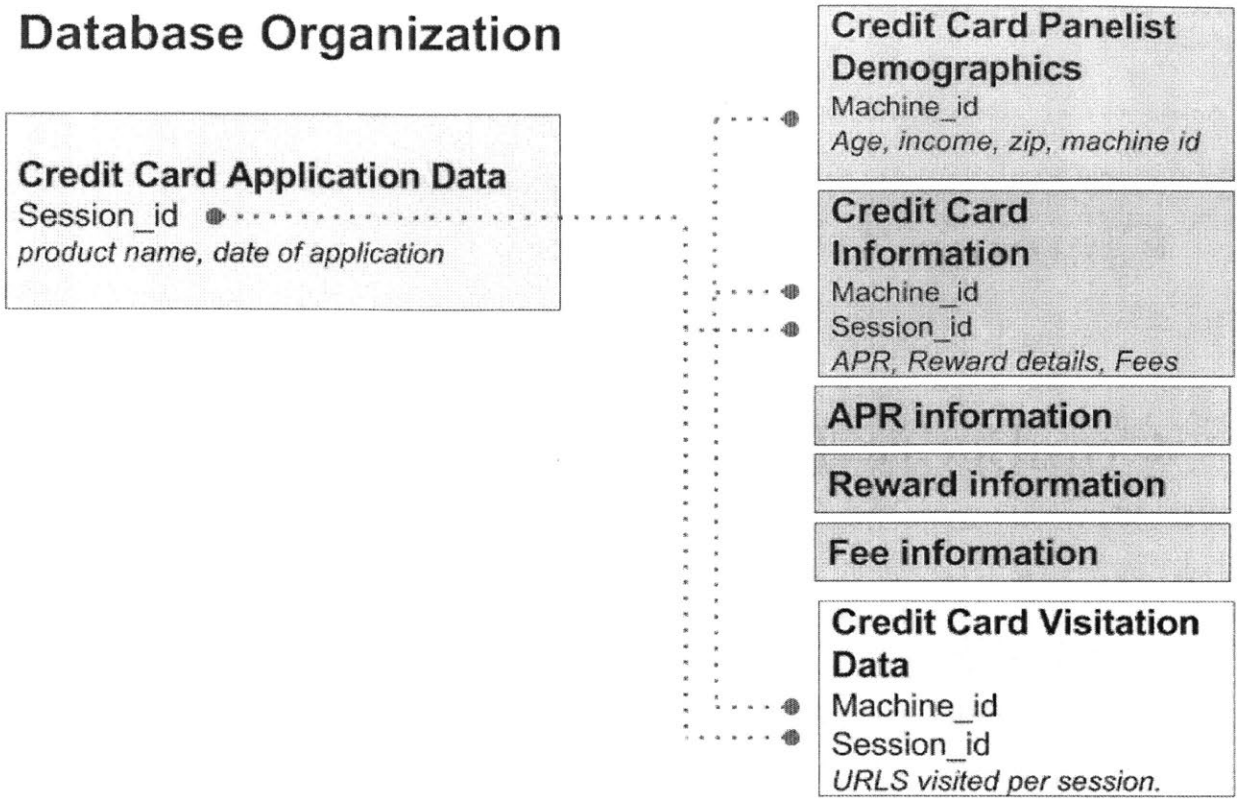


Figure 3-1: The organizational structure of the ComScore Credit Card Acquisition dataset.

3.2 Methodology

ComScore builds a very large panel using aggressive recruiting methodology and accounts for error by using advanced statistical methods and controls [8]. Panelists are paid to participate and install their Unified Data Management software. ComScore developed this proprietary browser extension to defend against cookie deletion and cookie rejection. The Credit Card Acquisition dataset and the analysis below is only representative of one of their products.

3.2.1 Credit Card Application Data

This dataset records a panelist's application for credit card and contains the session id for the application. This dataset has records for 88,524 applications submitted by 55,772 panelists. The bank domains represented include: 'fni-stl.com', 'bankofamerica.com', 'citibankonline.com', 'discovercard.com', 'wellsfargo.com', 'capitalone.com', 'americanexpress.com', 'citicards.com', 'chase.com', 'applyonlinenow.com', 'partnercardapply.com', 'barclaycardus.com', 'usbank.com', 'accountonline.com'. The filename for the database is Credit_Card_Application_Data_FINAL.txt.

Example Row

April 2015 , 2015-04-01 01:04:07.087 , 177051334 ,2639159865552,capitalone.com Platinum , 1 , 1

Schema

1. **Month:** the month the application was submitted
2. **Event_time:** the time stamp of the initiation if there was no submission or the time stamp of the submission (GMT)
3. **Session_id:** unique identifier for the session
4. **ProductDetail :** name of the card (standardized)
5. **Init:** 1 means that a panelist initiated the application
6. **Submit:** 1 means that a panelist submitted the application, 0 means that a panelist abandoned the application

3.2.2 Credit Card Visitation Data

This dataset records for 179,603 URL navigations in particular sessions. It contains 7,738 panelists with a mean number of URLS per panelist of 23.21. The standard deviation of URLs per panelists is 29.66. The bank domains with URLs are: 'bankofamerica.com', 'discovercard.com', 'capitalone.com', 'americanexpress.com', 'chase.com', 'applyonlinenow.com', 'barclaycardus.com'.

The filenames for the databases are Credit_Card_Visitation_Data_q1_2015.txt, Credit_Card_Visitation_Data_q1_2016.txt, Credit_Card_Visitation_Data_q2_2015.txt, Credit_Card_Visitation_Data_q2_2016.txt.

Example Row

203774723, 8205829755262666019,2016-03-11 01:11:49.037, services2.capitalone.com///keepAliveSinglePage

Schema

1. **Machine_id:** unique identifier for each panelist.
2. **Session_id:** unique identifier for the browsing session.
3. **Event_time:** time stamp of the visit (GMT).
4. **URL:** includes the host, directory, and page portions of the URL.

3.2.3 Credit Card Panelist Demographics

This dataset records the demographics of 55,772 panelists. Many panelists submit multiple demographics surveys over time. There are 70,270 demographics survey records. The filenames for the database is Credit_Card_Demos_Data_FINAL.txt .

Example Row

17523872, March 2016, US; 25k-39.999k Louisiana, 35-44, 70115, <600

Schema

1. **Machine_id:** unique identifier for each panelist.
2. **Month:** the month the survey was submitted
3. **HH_income:** Household income.
4. **State:** the panelist's state of residence.
5. **HH_age:** Head of household age.
6. **Zip Code:** the zip code of the panelist's home.
7. **Risk_break:** the credit score of the panelists.

3.3 Auxiliary Credit Card Information

The Credit Card Acquisition Info datasets are provide auxiliary data about the credit card awarded to the accepted applicants. All records are indexed by the app_id attribute.

3.3.1 APR information

This dataset Provides 213,613 records detailing the APR terms of the card. The total number of panelists with a record is 45,997. The filename for the database is

Credit_Card_CCAcqDtl_APRinfo_final.txt.

Example Row

8714537, 1874857256000003830, citicards.com, Purchase, 0%, 13.24%, 21 months, 1

Schema

1. **machine_id:** unique identifier for each panelist.
2. **session_id:** unique identifier for the browsing session.
3. **domain_name:** URL of the bank.
4. **aprType:** the record type, aspect of APR being detailed.
5. **IntroAPRRate:** the APR rate the card begins with.
6. **OnGoingAPRRate:** the APR rate the card has after the introductory period.
7. **APRDuration:** the duration of the introductory period.
8. **app_id:** the unique identifier for the application

3.3.2 Fee Information

This dataset provides 252,528 datapoints detailing fee information for the card. The cards are awarded to 46,231 applying panelists in the dataset. The filename for the database is Credit_Card_CCAcqDtl_FEEinfo_final.txt.

Example Row

10390288, 3797221850298904303, chase.com, Balance Transfer Fee,5.00%,$5 $5, 2

Schema

1. **machine_id**: unique identifier for each panelist.
2. **session_id:** unique identifier for the browsing session.
3. **domain_name:** URL of the bank.
4. **feeType:** the type of fee being recorded.
5. **feePercent:** the percentage fined, if the fee is a percentage.
6. **feeAmount:** the fined amount, if the fee is an absolute amount.
7. **feeMin:** minimum absolute amount to be fined.
8. **app_id:** the unique identifier for the application

3.3.3 Reward Program Information

This datast provides 45,159 datapoints detailing the reward programs for the cards awarded to 20,082 applying panelists in the dataset. The filename for the database is Credit_Card_CCAcqDtl_MAINinfo_final.txt.

Example Row

7661215, 62959841, capitalone.com, Ongoing, 1%, cash back, 1

Schema

1. **machine_id**: unique identifier for each panelist.
2. **session_id:** unique identifier for the browsing session.
3. **domain_name:** URL of the bank.
4. **rewardcategory:** details if the fee was ongoing or a one time award..
5. **rewardamount:** the amount awarded often as a percentage.
6. **rewardtype:** the general, e.g. miles, points, credits.
7. **app_id:** the unique identifier for the application

3.3.4 Auxilliary Information Header

This datset provides the indexing information for all other card information records. There are 79,839 records for 50,638 applying panelists. The filename for the database is Credit_Card_CCAcqDtl_MAINinfo_final.txt.

Example Row

13703835, 52474229, citicards.com, Sears, 2

Schema

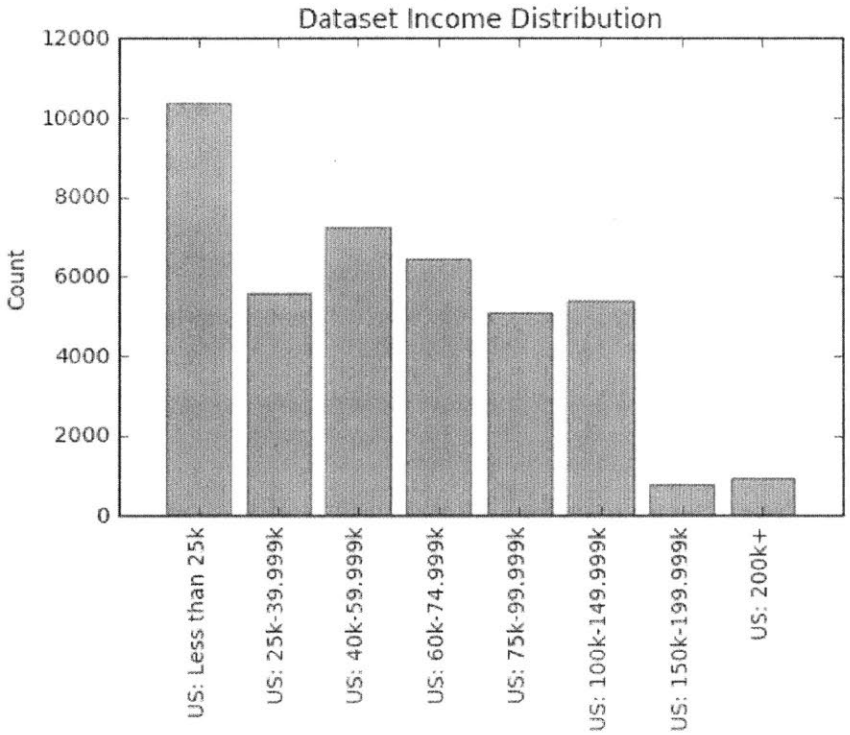
1. **machine_id**
2. **session_id**
3. **domain_name**
4. **productdetail**
5. **app_id**

Dataset Statistics

In this section we will cover some statistics about the dataset we derive from the comScore data. We use a small portion of the comScore data which is at the intersection of users with Demographics, Visitation, and Auxiliary Card information. This represents a dataset of 58,043 datapoints and is a subset of the comScore panelists. The statistics here are for that subset, and are not statements about the general comScore panel.

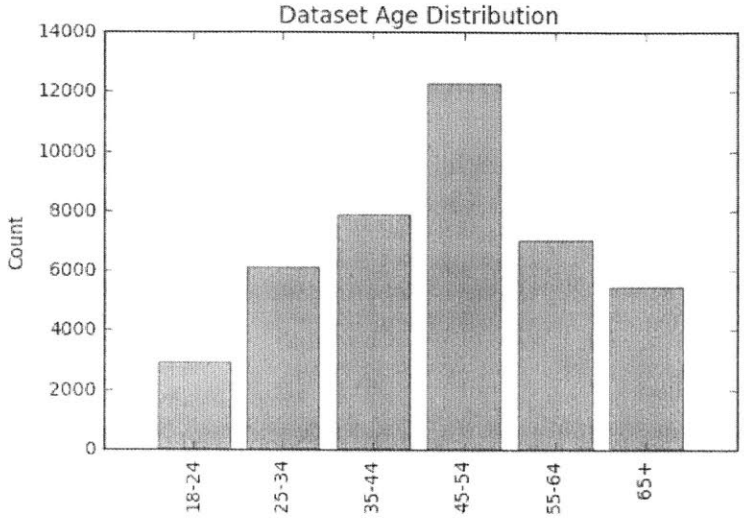
Incomes

We present the incomes for the dataset. There is a good representation for all general income levels less than $150k.



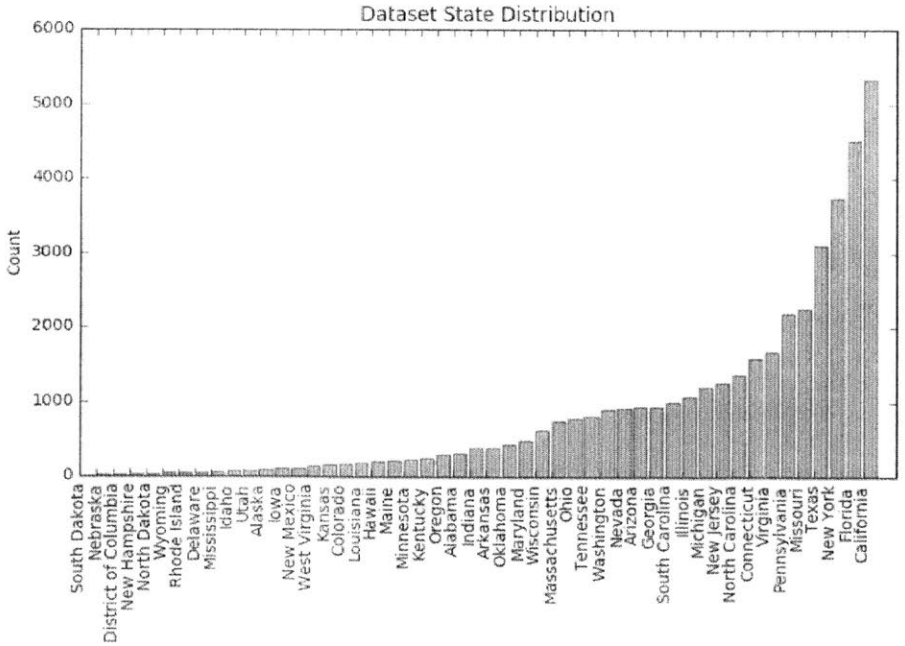
Age Distribution

The age distribution for our dataset peaks at 45-54.



Locations

The location distribution of panelists in the dataset matches the general distribution in United States. Population with over representation in California and Florida.

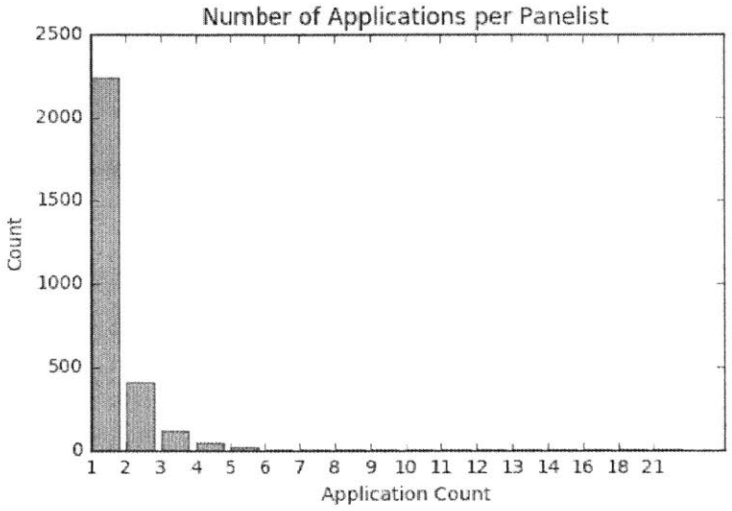


Application Statistics

In this section we will focus on the credit card application statistics.

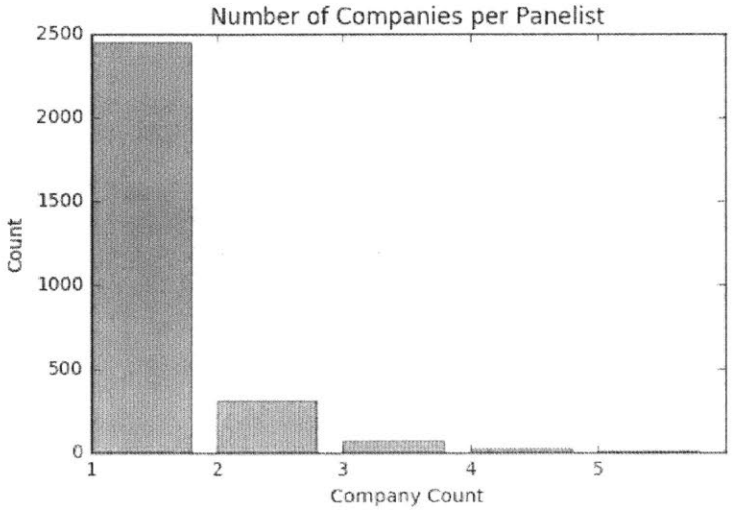
Number of Accepted Applications per Panelist

Since our dataset requires cards to have been approved, about one third of the panelists in our dataset were granted more than one card.



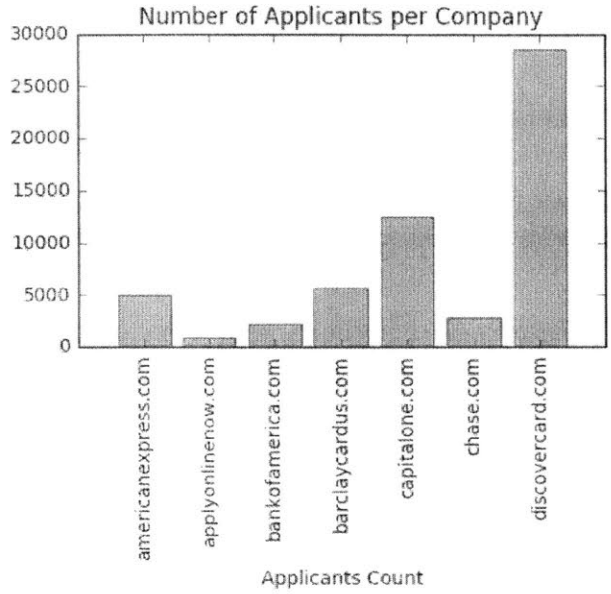
Number of Companies per Panelist

<ost applicants apply to cards at only one bank or company.



Number of Applications per Company

We see a power law distribution in companies represented in our dataset. The most represented company is Discover.



Links distribution

The most common URLs tend to be rather mundane and are the loaded guts of the websites. Often these files are used to govern the website behavior and control the appearance of the website. These are example URLs which account for the top 1000 URLs:

1. `applynow.chase.com/FlexAppWeb/styles/flexapp/css/blank.html`,

   A seed template which styles the rest of the pages on chase.com.

2. `www.chase.com/apps/services/tags/https/applynow.chase.com/FlexAppWeb/renderApp.do`,

   A URL which calls a java API call. This tells the backend to serve the application.

3. `application.capitalone.com/icoreapp/jsp/gmcLanding.faces`,

A photo icon.

However the mid-frequency URLs tend to carry information about the interests of the consumer. These higher variance URLs are often indicative of the consumer's interests and background.

1. `www.chase.com/apps/services/tags/https/www.chase.com/online/military/military-personal.htm`,

   A landing page for military personnel.

2. `www262.americanexpress.com/business-card-application/prospect-app/business-platinum-charge-card/prospectsubmit/0-9-0`,

   The initial page of the Chase Business Platinum card application.

3. `creditcards.chase.com/slate-credit-card/learnmore-apply1`,

   The Slate Card information page

4. `www.chase.com/resources/fraud-prevention`,

   Information on Chase's fraud prevention program.

The higher variance, less frequent URLs carry the most information about the panelists interest. Furthermore we can see the wealth of information that lays untapped in browsing history.

Although most browsing information tends towards the mundane browser mechanics necessary to operate the website, we are interested in the dark data. The humdrum of internet commerce produces an extraordinary amount of unused browsing data. Classical models are ill equipt to handle the data as often the relationships are statistically complex and nonlinear.

In the next chapter we set the stage for a comparison of Deep Learning and Logistic Regression and demonstrate the ability of Deep Learning to find useful representations of data emissions.

Chapter 4

Preference Modeling

4.1 Motivation

The main contribution of this thesis is in exploring the use of more advanced models to answer managerial questions formulated as inference problems. To perform Bayesian inference we need to learn a preference model of customers for credit cards. Deep Learning can learn rich representations of customers and capture subtle variations of preferences.

4.2 Preference Model

Given a feature vector which encodes a customer, we wish to learn a distribution over card attribute preferences. The preference model of a card c given customer x is the product of all attributes a likelihoods in the attribute set $\mathcal{A}$.

$$p_{C|X}(c, x) = \prod_{a \in \mathcal{A}} p_{C_a|X}(c_a|x)$$

We will explore three different models to learn the distribution with interesting consequences. We will also evaluate their performance with Bootstrap sample (explained in the performance section).

1. **Baseline** We use a baseline which predicts the most common card in the training data. For our dataset this was the Discover It Card with a cashback rewards program. Because each bootstrap sample randomly splits the dataset, the accuracy of the baseline (percentage of cards which is the Discovery it Card) varies from bootstrap sample to sample.

2. **Multinomial Logistic Regression (MLR)** We learn each attribute distribution is shallow $p_{C_a|X}(c_a|x)$ model independently and assume the likelihood factors over card attributes.

3. **Deep Neural Network Regression (DNN)** We learn each attribute distribution $p_{C_a|X}(c_a|x)$ model over a deep neural network. This uses Deep Learning to model customers with a rich hierarchical representation.

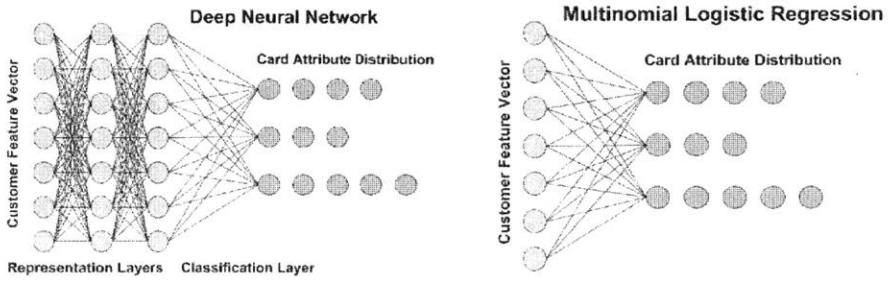


Figure 4-1: Visualization of the model architectures. We can see that the Deep Neural Network can be thought of as a Multinomial Logistic Regression on top of a Deep Representation.

A similar Multinomial Logistic and Deep Learning model performance would suggest that there is little mutual information between various attributes, and the dataset has few nonlinearities. It could also suggest that we have insufficient data to attain statistical strength for a Deep Learning model (as deep models have more parameters).

4.2.1 Independence structure

The representation layers of the Deep Neural Network model create interdependence between the various attribute distribution layers and allow us to make predictions about unseen attribute combinations. A Multinomial Logistic Regression model can not capture interdependence between attributes and models them as separate models.

This independence structure constrains the class of questions we can ask. A Multinomial Logistic Regression can not answer managerial questions about the interplay of attributes such as "Would a higher reward rate make customers more willing to accept a higher interest rate?"

The Deep Neural Network models captures a representation where we can perform inference on unseen cards through its similarity to learned cards in the representation space.

4.3 Datasets

We are interested in investigating the ability of various models to distill information that lies in the unstructured unintelligible links data. Two datasets are considered:

1. **Demographics (NOLINKS):** A dataset of 58k pairs of customer feature vectors and chosen card attribute vectors. The customer feature vectors include only demographics information.

2. **Demographics+Links (LINKS):** A dataset of 58k pairs of customer feature vectors and chosen card attribute vectors. The customer feature vectors both demographics information as well as a 30 minute session browsing history encoded as a 0-1 vector.

4.4 Training

Training algorithms search for the best model by iteratively changing the parameters to improve a loss function.

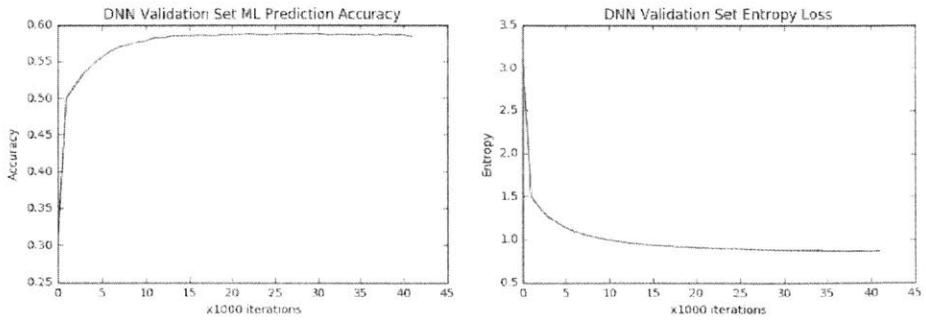


Figure 4-2: The training statistic on the validation set. We use the entropy loss criterion to terminate training and we see convergence occurs after 40,000 iterations.

The Multinomial Logistic Regression model was trained using the classical L-BFGS [19]. L-BFGS is a classic second order optimizer.

We trained the Deep Neural Network using Stochastic Gradient Descent provided by the Torch [18] optim package. The performance reported here was achieved with the following learning parameters.

- **learning rate:** $\alpha = 0.05$,
- **learning rate decay:** $\eta = 0.0001$
- **weight decay:** $\lambda = 0.0001$
- **momentum:** $\nu = 0.1$

Furthermore we use batches of size 100, and GPU accelerates training. Batches increase the statistical strength of our gradients and improve convergence speed and stability of the algorithm.

4.5 Evaluation

We use two criteria to evaluate the performance of the models. The maximum likelihood prediction rule captures the prediction accuracy of the models. The cross entropy criterion is a measure of how well the models capture the probability distribution of the attributes.

1. **Maximum Likelihood Prediction:** We predict the attributes which have the highest likelihood. The value of this criterion is the number of correct attributes in the prediction.

2. **Cross Entropy Criterion** The cross entropy criterion captures the surprise of the model by the outcome. Namely, the number of bits necessary to encode the result given the predictive distribution. It is a measure of distance between distributions.

$$\mathcal{L}(x, c) = \sum_{i=1}^{N} \log \left(p_{a|x}(c_i|x) \right)$$

4.5.1 Performance

To evaluate the performance of each model we employ a simple Monte Carlo technique called **Bootstrap Sampling**. The idea is simple — If we have a statistic, such as a performance metric, we can estimate the true distribution of the performance metric by generating N datasets, each a different random training-validation-test set split. We can then estimate p-values for our hypothesis. Namely, Deep Neural Networks have superior performance over Multinomial Logistic Regressions.

We gathered 100 bootstrap samples with a 50-25-25 training-validation-test set split and report the mean evaluation statistics. All p-values were well below 0.001.

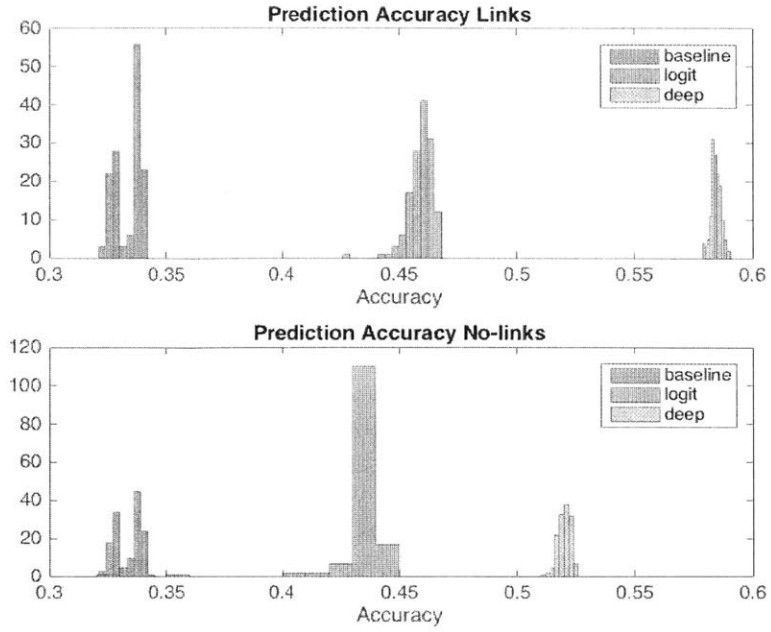


Figure 4-3: 100 Bootstrap sample performance on the LINKS dataset (Above) and the NOLINKS dataset (below).

NOLINKS Bootstrap Samples Statistics

Model	Prediction Accuracy(Standard Deviation)	Cross Entropy
Baseline	33.41%(0.55%)	∞
MLR	43.45%(0.53%)	2.1028
DNN	**52.01%(0.21%)**	**1.2367**

LINKS Bootstrap Samples Statistics

Model	Prediction Accuracy(Standard Deviation)	Cross Entropy
Baseline	33.45%(0.55%)	∞
MLR	45.74%(1.32%)	1.7314
DNN	**58.50%(0.27%)**	**0.8721**

As we can see the DNN model outperformed the MLR model on all datasets. Furthermore, we observe a superior cross entropy on the DNN which demonstrates that Deep Learning models are better able to capture the attribute distribution.

We also remark that the DNN model is able to distill more information from the

unstructured browsing session than the MLR model. We see that the increase in performance across the two datasets is significantly larger in the DNN.

4.6 Conclusion

In this section we outlined a method for capturing a preference model that predicts customer choices. We outlined various training algorithms and demonstrated that Deep Neural Networks significantly outperform the Multinomial Logistic Regression model.

In the next chapter we will see how to utilize the learned preference model to answer a variety of managerial questions.

Chapter 5

Managerial Applications

5.1 Motivation

In this section we frame several managerial questions as inference problems empowered by sophisticated Deep Learning models. There are three general inference problems we will tackle, offering an example managerial question for each.

- **Inference of Customer Feature Vector given Card Attributes:** What are the target demographics for a particular card?
- **Inference of Card Attributes given Customer Feature Vector(s):** What types of cards attract low income, low credit score customers vs. high income high credit score customers?
- **Competitive Market Simulation:** Given a cost model, what cards would be most profitable to introduce?

5.2 Inference of Customer Feature Vector given Card Attributes

Given a card we wish to infer the customer demographics distribution. We approach this problem by applying Markov Chain Monte Carlo methods to sample from the

posterior distribution. The posterior over customer feature vectors X given a particular card attribute set c is simply Bayes rule with the model likelihood and the testset prior distribution..

$$p(X|C = c) = \frac{p(c|X)p(X)}{P(c)}$$

Feature vectors for our model include demographics information and browsing history. Inference problems involving browsing history may inform choices pertaining to customer acquisition. Browsing history may evaluate advertisement venues, as well as inform the optimization of websites.

We will explore inference of customer demographics by comparing the Jetblue and Delta cards. Both cards offer similar rewards schemes yet differ drastically in their target demographics. The JetBlue MasterCard offers 5,000 miles and a 1 mile per dollar reward. The Delta Reserve card offers 10,000 miles and a 1 mile per dollar reward. We see the JetBlue card is more popular with students 18-24 earning $60k-74k. The Delta Reserve card on the other hand appeals to a more adult customer base with the strongest affinity for 55-64 year olds earning $25k or less.

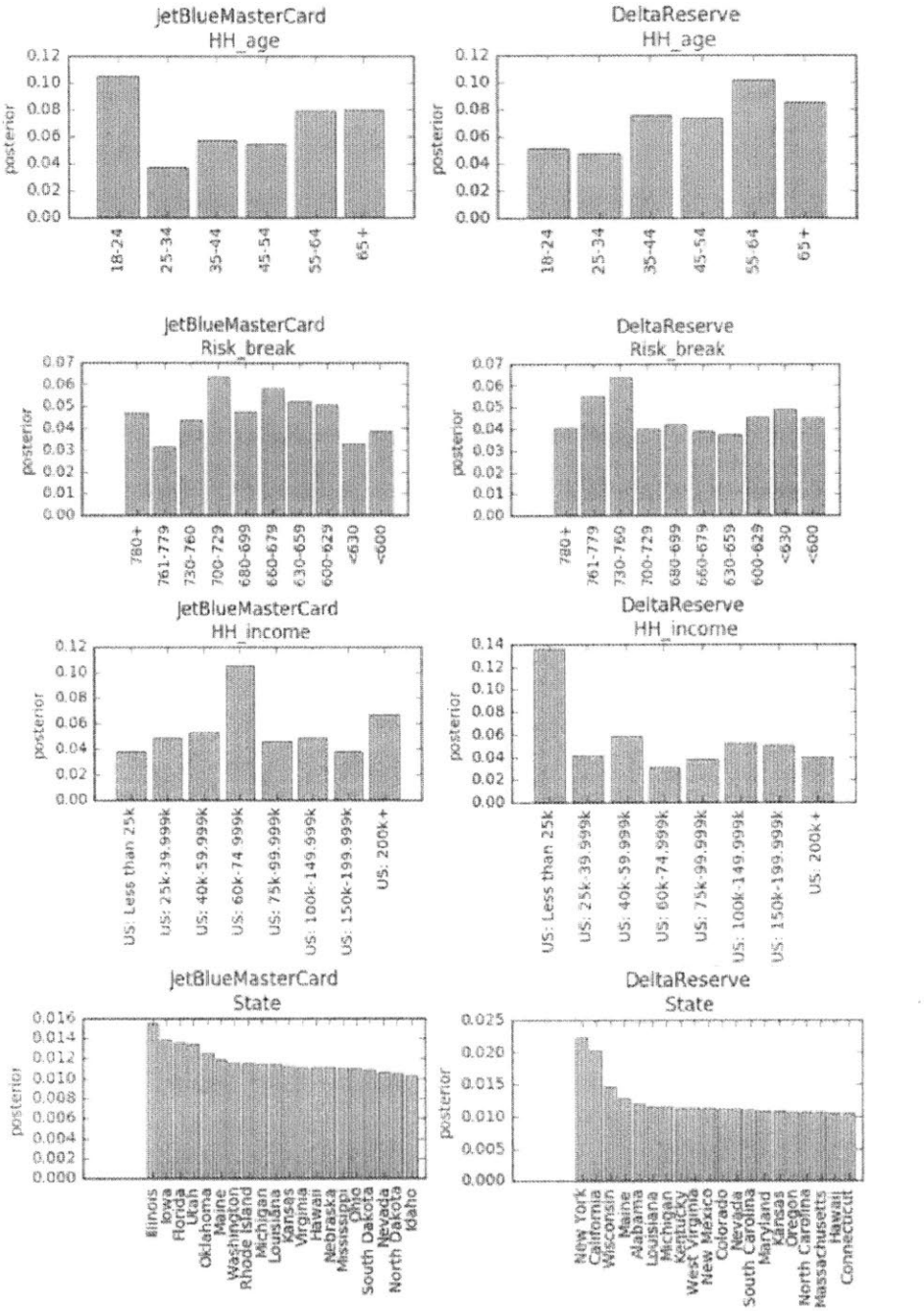


Figure 5-1: Target demographics of the Delta Reserve Card vs. the JetBlue MasterCard.

Next we will compare the hotel card category. We pick two similar cards which attract very different customer demographics, the Marriot Rewards Premier Business

card and the Hilton Honors card. Both cards offer 10 points per dollar spent at the hotel chain. However, we see that the Marriott Rewards Premeier Business card has a much stronger affinity for the 23-34 year olds with income more than $200K, and 780+ credit scores, whereas the Hilton Honors card has a much broader customer base. Part of this difference may be due to over-fitting as the dataset has few points for individuals in this income bracket.

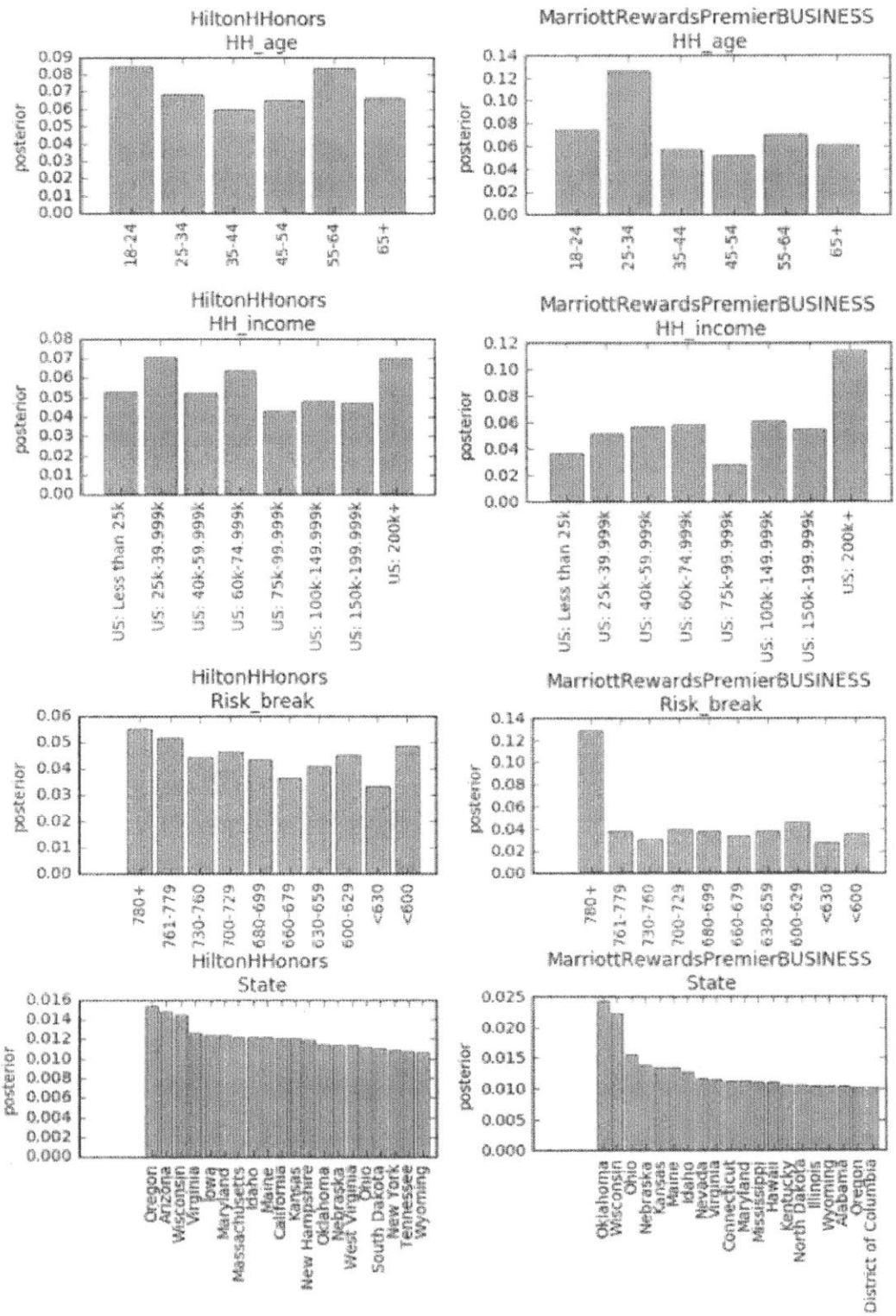


Figure 5-2: Target Demographics of the Hilton HHonors card and the Marriott Rewards Premier Business card.

5.3 Inference of Card Attributes given Customer Feature Vector(s)

Given a customer feature vector inference on card attribute preferences is performed as a forward pass of the Deep Neural Network model. We choose two samples to show the subtle changes in preferences in two customers from the test set.

1. **high income high credit score:** We can see that the model has high affinity for cash reward business cards, and premium hotel cards. The model suggests a mid-low APR range with high probability (14.24%). The reward schemes with highest probability is are cashback, points, and miles systems.

Product Detail	APR
Classic Platinum MasterCard	14.24%
Barclaycard Rewards Mastercard	10.99% -14.24%
Marriot Rewards Premier Business Card	

2. **low income low credit score:** The model also predicts a high interest rate for the customer (19.49%). The reward schemes with highest likelihood are travel related.

Product Detail	APR
Disney Reward Premium Visa	19.49%
Delty Sky Miles Gold	19.24%
Venture One Rewards Visa	

5.4 Competitive Market Simulation

In this section we will explore new product introduction strategy using Stochastic Search methods. Stochastic Search methods have a rich history in A.I., ranging from the record breaking chess program Deep Blue [7] , to the Monte Carlo Tree Search algorithm employed by Alpha Go [1]. Stochastic Search explores a search space by moving in a random walk biased towards areas that have the best reward, and keeping

the best strategy so far. In other words, stochastic search seeks the best strategy in a search space, and does so proportional to the observed reward in the area.

We will use Stochastic Search to find the best strategy to introduce a new card into the market by using our preference model to approximate card adoption and a cost model to compute the reward function.

5.4.1 Cost Model

Our profit model is computed as a percentage of expenditure. We assume that each customer spends 10% ($s_{customer} = 0.1$) of their annual income ($i_{customer}$) using the card. We charge 1% in transaction costs ($t = 0.01$). We assume that customers borrow 50% of their expenditures $b_{customer} = 0.5$, and we charge a card dependent interest rate (a_{card}). The card dependent rewards rate (r_{card}) is deducted as a cost of the product. Furthermore, we assign a default probability to each customer proportional to their credit score($d_{customer}$). (see appendix).

$$t = \textbf{transactionCharge}$$
$$b_x = \textbf{customer borrowing rate}$$
$$a_{card} = \textbf{card interest rate}$$
$$d_x = \textbf{defaultRate}$$
$$r_{card} = \textbf{reward rate for card}$$
$$s_x = \textbf{Monthly spend as a percentage of income}$$
$$i_c = \textbf{customer income}$$
$$Reward(customer, card) = (t + a_{card} * b_{customer} - d_{customer} - r_{card}) * s_{customer} * i_{customer}$$

5.4.2 Stochastic Search Algorithm

Our Search Algorithm receives a set of customer feature vectors from the test set which represent our market, and a set of N competition cards as a list of attributes

per card. We generate a random card to introduce, x. Given our current card x we randomly change an attribute and use our preference model to sample customer card choices and compute the profits for the new proposed card. We accept according to a Metropolis Hastings acceptance criterion. After 1000 MCMC samples from the reward distribution, return the card with the highest estimated reward.

5.4.3 Simulation Results

We constrained the card space to include only three attributes Rewardamount, Rewardtype, OnGoingAPRRate. These were allowed to only take values which could be priced by the reward function. For example, we removed reward attributes such as 'Amazon Gift Card'. The test market contained $14,511$ customer feature vectors. We chose the five most popular competitor cards.

1. **Mixed Market:** Our first experiment is to optimize a card in a varied market that has strong competitors in every category. We selected the most popular card of each category. The algorithm identifies opportunity in competing with the existing cashback card. We see that by reducing our APR to 12.9% we capture a predicted \$15.9M in market share.

Card	Rewards	Rewards rate	APR	Profit(Pre-existing)
1	Cashback	1%	17.24%	\$8.94M(\$20.9M)
2	Points	1%	19.24%	\$6.48M(\$8.6M)
3	Miles	1%	15.99%	\$2.2M(\$3.9M)
new	Cashback	1%	12.9%	\$15.9M

2. **Competitive Market:** We restrict our existing market to the three most popular cashback cards in our dataset. We see that the algorithm arrives at a solution that has the lowest APR. Although this suggestion seems trivial, there is a trade-off between profit and customer share in optimizing APR. Our result underscores the importance of the cost model.

Card	Reward	Amount	APR	Profit(pre-existing)
1	Cashback	5%	23.24%	$3.18M($6.2M)
2	Cashback	1%	23.99%	$5.87M($10.8M)
3	Cashback	5%	16.99%	$9.49M($13.99M)
new	Cashback	1.5 %	12.99%	$13.2M

Conclusion

We have developed a paradigm for answering managerial question by framing them as inference problems. We translate frameworks developed to allow Game A.I.s to act optimally to the context of making marketing decisions. We have shown the potential applications of this framework with empirical data.

Chapter 6

Conclusion

We have presented a framework for learning about customer product preferences from inexpensive Big Data exhaust. The power of our approach is that we are able to train Deep Neural Networks by embracing unstructured Big Data. A surprising amount of predictive power can be distilled from the data exhaust of ordinary internet commerce. In turn, these sophisticated models can be used to answer managerial questions regarding product opportunity discovery.

6.1 A Leap of Faith

Perhaps the most unsettling of implication of the work presented here is that so much of our predictive advantage comes from uninterpretable data exhaust in models which seem over-parameterized. Deep Learning may give us the ability to automate the search of relevance and importance in the input data, but by having so many parameters it may actually blind us from the interpretation of the unstructured data.

It suggest future methods might let go of interpretability and take a leap of faith. When Deep Learning revolutionized computer vision, the research community abandoned symbolic models which took constructive approaches to identifying objects in images because it became obvious that although we see, say, a cat we lack an ability to explain how and why we came to that conclusion. When intuition was abandoned, connectionist models like Deep Neural Networks were found to converge to strikingly

similar networks as those found in the early vision system in the human brain. Deep Neural Networks form continuous representations that are otherwise inaccessible.

6.2 Future Research

The comScore dataset is quite restricted in size and depth. Furthermore this work assumes that consumers implicitly reject all other cards by acquiring a card. Websites like Credit Karma and Nerd Wallet would give this research a rigorous notion of the consideration set. The input to the Deep Neural Network could contain a set of flags for which cards are being considered.

6.2.1 Sequential Modeling

The Monte Carlo techniques we use to answer managerial questions represent a single iteration of the techniques used in Deep Reinforcement Learning systems like AlphaGo. In the setting of a Game A.I., we repeat the process of searching for optimal actions and sequentially observe their outcome rewards. In our setting we can not simulate the true market response and therefore we can only estimate one move ahead. Improving our method to tackle a sequential control process is difficult in an academic setting and seems strictly an applied problem. But perhaps sequential modeling of browsing history might result in more predictive power. Recurrent Neural Networks require even larger datasets and are perhaps out of the scope of the comScore Dataset.

6.2.2 Tackling other Products

The credit card domain has a clear intuition as the structured product space of attributes. A card constitutes of a reward program, an interest rate, and borrowing limit policies that operate rather independently and can be mixed. Other products with more amorphous product spaces, such as cars, remain out of reach.

However, credit cards are a capricious product space themselves. Brand loyalty

can be difficult to capture and integrate into answering managerial questions. We approached this problem by marginalizing the distribution over all brands and ignoring it in our inference. However this might lead to suggestions that Alaskan Airlines should introduce a cashback card to compete against Bank of America. Integrating brand information is an important future work for this project.

6.3 Conclusion

Our framework for learning about customer preferences from Big Data has the potential to scale to levels which are prohibitively costly with current Market Research and Analytics methods. Moreover it is capable of extracting from the dark knowledge in unstructured data exhaust. We have shown that our predictive models can be successfully used to answer managerial products regarding product opportunity discovery.

In Targeted Advertisement, enterprises select their audience through sophisticated models. In Advertisement Morphing, enterprises can customize their messages. Our framework empowers enterprises to optimize their products in an automated manner, allowing for customization to the needs of the customer.

Bibliography

[1] Silver, David and Huang, Aja and Maddison, Chris J and Guez, Arthur and Sifre, Laurent and Van Den Driessche, George and Schrittwieser, Julian and Antonoglou, Ioannis and Panneershelvam, Veda and Lanctot, Marc and others. *Mastering the game of Go with deep neural networks and tree search*, Nature, Volume 529, 484–489 , 2016, Nature Publishing Group

[2] Hauser, John R. et al. *Website Morphing.* MARKETING SCIENCE 28.2 (2009): 202-223. 2009 INFORMS

[3] Scott, Steven L., et al. *Bayes and big data: The consensus Monte Carlo algorithm.* International Journal of Management Science and Engineering Management 11.2 (2016): 78-88.

[4] Ian Goodfellow and Yoshua Bengio and Aaron Courville. *Deep Learning*, Book in preparation for MIT Press, **http://www.deeplearningbook.org** ,2016

[5] Hastings, W. Keith. *Monte Carlo sampling methods using Markov chains and their applications.* Biometrika 57.1 (1970): 97-109.

[6] Dagum, Paul, and Michael Luby. *An optimal approximation algorithm for Bayesian inference.* Artificial Intelligence 93.1 (1997): 1-27.

[7] David Levy, and Monty Newborn *How Computers Play Chess.* Computer Science Press(1991). ISBN 0-7167-8121-2.

[8] *Review of comScore Methodology.* Advertising Research Foundation (ARF). 2001.

[9] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg and Li Fei-Fei. *ImageNet Large Scale Visual Recognition Challenge.* IJCV, 2015.

[10] Hinton, Geoffrey E., and Ruslan R. Salakhutdinov. *Reducing the dimensionality of data with neural networks.* Science 313.5786 (2006): 504-507.

[11] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. *Imagenet classification with deep convolutional neural networks.* Advances in neural information processing systems. 2012.

[12] LeCun, Yann, et al. *Gradient-based learning applied to document recognition.* Proceedings of the IEEE 86.11 (1998): 2278-2324.

[13] David E. Rumelhart, James L. McClelland and PDP Research Group (1987). Parallel Distributed Processing: *Explorations in the Microstructure of Cognition*

[14] Erhan, Dumitru, et al. *Why does unsupervised pre-training help deep learning?.* Journal of Machine Learning Research 11.Feb (2010): 625-660.

[15] Vincent, Pascal, et al. *Extracting and composing robust features with denoising autoencoders.* Proceedings of the 25th international conference on Machine learning. ACM, 2008.

[16] Nair, Vinod, and Geoffrey E. Hinton. *Rectified linear units improve restricted boltzmann machines.* Proceedings of the 27th International Conference on Machine Learning (ICML-10). 2010.

[17] Treadgold, Nicholas K., and Tamas D. Gedeon. *Simulated annealing and weight decay in adaptive learning: the SARPROP algorithm.* IEEE Transactions on Neural Networks 9.4 (1998): 662-668.

[18] Collobert, Ronan, Koray Kavukcuoglu, and Clement Farabet. *Torch7: A matlab-like environment for machine learning.* BigLearn, NIPS Workshop. No. EPFL-CONF-192376. 2011.

[19] Zhu, Ciyou, et al. *Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization.* ACM Transactions on Mathematical Software (TOMS) 23.4 (1997): 550-560.