Title
Optimal Decisions with Multiple Agents of Varying Performance

Author
Silverman, Noah

Publication Date
2013

Peer reviewed|Thesis/dissertation

University of California

Los Angeles

Optimal Decisions with Multiple Agents of Varying Performance

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Statistics

by

Noah Silverman

2013

Abstract of the Dissertation

Optimal Decisions with Multiple Agents of Varying Performance

by

Noah Silverman

Doctor of Philosophy in Statistics

University of California, Los Angeles, 2013

Professor Rick Paik Schoenberg , Chair

In this dissertation, I look at four distinct systems that all embody a similar challenge to modeling complex scenarios from noisy multidimensional historical data. In many scenarios, it is important to form an opinion, make a prediction, implement a business decision, or make an investment based upon expected future system behavior. All systems embody an amount of uncertainty, and quantifying that uncertainty using statistical methods, allows for better decision making. Three distinct scenarios are discussed with novel application of statistical methods to best quantify the endogenous uncertainty and aid in optimal decision making. Two chapters focus on predicting the winners of a horse race, one on predicting movement of a stock index, and the fourth on molding response from an online advertising effort.

The first horse racing chapter uses a hierarchical Bayesian approach to modeling running speed, using a novel grouping of races into "profiles" and then pooling information across those profiles to improve predictive accuracy. The second horse racing chapter implements a novel conditional logistic regression that is modified by frailty parameter derived from winning payoff, and then regularized with a LASSO. High speed parallel algorithms, running on a GPU, were hand coded to

optimize tuning LASSO parameters in rapid time.

The chapter on stock index prediction explores the application of ensemble filters. I show how an ensemble of filters on individual member stocks is a better predictor of index direction than a filter directly on the index.

The chapter on advertising explores how the clicks and sales from an AdWords campaign may be modeled with a re-paramaterized Beta distribution to better capture variance. Empirical data from a live campaign is studied, with a hierarchical Bayesian framework for brand features solved using a Metropolis within Gibbs algorithm.

The dissertation of Noah Silverman is approved.

Robert Zeithammer

Ying Nian Wu

Mark S. Handcock

Rick Paik Schoenberg , Committee Chair

University of California, Los Angeles

2013

To my Rick Schoenberg . . .
who taught me more than he realized.
He was my first statistics teacher,
supported my choice to pursue a Phd.,
and became my committee chair.
His kindness, support, and passion for
statistics has helped shape my career,
and I will always be grateful.

Table of Contents

List of Figures

List of Tables

Vita

1998-2008	Founded AllResearch, Inc. The first company to track news on the Internet, and also the first company to ever track trademark abuse of luxury brands on The Internet
2011	Founder and Chief Scientist at Smart Media Corporation, a statistical consulting firm.
2011	M.S. Statistics University California Los Angeles
2012	Published A Hierarchical Bayesian Analysis of Horse Racing in The Journal Of Prediction Markets Volume 6.
2013	Presented research on Mathematical Models of Bitcoin Pricing at The Bitcoin 2013 Conference.
2013	Presented paper on Conditional Logistic Regression with Frailty for Horse Race Prediction at The 15th Annual Conference on Gaming and Risk.
2013	Published "Predicting Horse Race Winners Through Conditional Logistic Regression with Frailty" in The Journal Of Prediction Markets Volume 7.
2013	Published "Welcome to the E-Penny Opera - A Discussion of Crypto-Currencies" in The Wilmot Magazine.
2013	Published "The E–Penny Opera: Act II" in The Wilmot Magazine.

CHAPTER 1

Introduction

In business, medicine, investing, gambling, and many other fields, one often needs to make a decision when there is imperfect, incomplete, or noisy information. The general problem exists when there is a finite resource to be allocated in a scenario where there is a potential loss or gain depending out a random future event or events. To further complicate things, this decision must be made when there are multiple "agents" acting collectively, either cooperatively, or competitively. The decision may require choosing the *best* agent, or *allocation* of resources between multiple agents.

The term "bet" and "investment" are often used interchangeably in academic literature. A commonly quoted joke is that "If you made a profit, it was an investment. If you lost money, it was a bet". Generally any action based on an event or events with uncertain outcome, where the goal is a gain or profit can be thought of as a bet. (Even buying a "blue chip" stock is simply a bet with a very low expectation of losing. Nothing is guaranteed in life.) For this dissertation I will refer to both bets and investments as simply bets.

The motivation for this dissertation is to study optimal decision making under a variety of scenarios involving multiple actors. The classification of scenario types is illustrated in figure 1.1

With a single agent, standard models such as linear, logistic, or Poisson regression can be readily adopted to most situations. Generally data about historical performance of the agent is used to model how the phenomena of interest

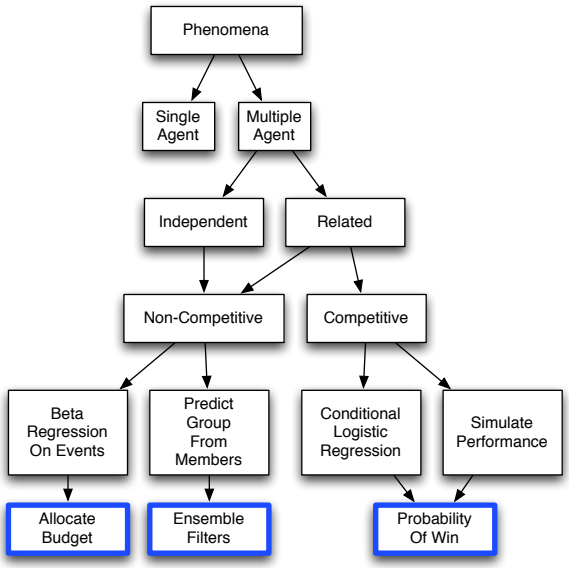

Figure 1.1: Categorization of Scenario Types

responded to the various independent covariates at the time of each past performance measurement. Then, data about the covariates for the current state of inquiry are used to model expected present or future performance. Once an estimate of future performance, or probability of an outcome has been calculated, a rational decision may be made.

Depending on the nature of decision required, different methods have been developed to determine bet (or investment) strategy based upon the aforementioned predicted outcomes. In the case of a binomial outcome, with a bet and potential profitable return, the decision is often reduced to what percentage of asset to bet on the outcome. (In the case of actual gambling on a game of chance, or investing in a stock, the decision is literally how many dollars to bet on the next outcome.) John Kelly [Kel56] provides an elegant solution to this decision process, commonly referred to as "The Kelly Criterion", provided that you can accurately estimate the probability of outcome. Applications of the Kelly Criterion, to both single outcomes and multiple outcomes have been well discussed in academic literature.

With multiple agents, either acting competitively, cooperatively, or independently, both the prediction and decision processes become much more complex. When looking at competing agents, we can either model each agent's performance, and then estimate the probability that their performance is the best, or we can directly model their probability of winning the contest. To model performance, I look at a hierarchical Bayesian model in Chapter 2. If I am directly modeling the probability of an agent winning the contest, then a Conditional Logistic Regression model may be used as detailed in Chapter 3. If agents are related, but acting independently over time, then the application of ensemble filtering may provide insight into the movement of the group as a whole. This is described in detail in Chapter 4. When considering the probability of events in a complex system, a re-paramaterized Beta Distribution captures variance much better than the more traditional logistic, as explore in Chapter 5.

1.1 Horse Racing

Two of the four papers included in this dissertation relate to horse racing. Horse racing is a parimutuel type of gambling event. A group of horses compete in a race where there is a single winner. There are a variety of bets, based on different winning outcomes or combinations of outcomes available to the public. All of the bettor's money is placed in a "pool" which is subsequently divided amongst the bettors who chose correctly. One interesting side effect of this structure is that the race courses report, in near real time, the fraction of betting pool that would be awarded if a chosen horse wins. This is a perfect proxy for the public's confidence in a given horse and their expected probability of that horse winning.

At first glance, this may seem an unusual system to study, but the structure of racing can be considered a proxy for financial markets that provides many unique advantages:

- Races always involve multiple agents. There are generally 8-14 horses per race.
- Races are decided very quickly, generally within 60 seconds or so.
- Many distinct pieces of useful data are tracked for every race. I have over 189 variables in my data set.
- There is a large volume of very high quality data readily available. For example, The Hong Kong Jockey Club provides several years of historical data for free on their website.
- I have the public's predictions and choices for every horse in every race in history. This provides an interesting second layer of information.

1.2 Stock Market Indices

A stock market index is a derivative representation of a group of stocks. These stocks in a group index are generally chosen because they are related in some way. For example a group of stocks may be a representative selection of technology companies, a selection of automative companies, or a selection of utility companies. The index is then a weighted combination of each stock's price with the intention that it will represent the overall value or direction of the industry being represented. The most well know index is the S&P 500, which is comprised of 500 stocks chosen to represent the American economy. At the most direct level of usage, investors will often use an index as a representative proxy for how an area of interest is performing. Further along in complexity are an entire family of investment products that are based directly on the index itself, with no regard for the underlying companies. For example, it is possible to buy and sell fictional "shares" directly in the S&P 500 index. Subsequently, there is an entire class of investors very interested in predicting future performance of the stock index.

While each stock's price may move stochastically between time periods, the index can be thought of as a smoothed filter that is representative of the industry as a whole. This relationship creates an interesting opportunity to estimate the future price of the index by analyzing the behavior of the individual component stocks. Analogous to a school of fish, the overall direction of the school can be seen as a composite of each fish's individual stochastic swimming path.

In Chapter 4, I propose a method of estimating future stock prices through a novel application of an ensemble of filters on the component stocks.

1.3 Search Engine Advertising

Search engine advertising, mainly on Google, has become the dominant form of Internet advertising, generating the most revenue for publishers and the best results for advertisers. While it may seem unusual to base a statistical study on a single company, in this case Google dominates the industry by such a large margin that data from other providers is literally insignificant. In 2011, 96% of Google's $37.9 Billion in revenues came directly from advertising. (Need authoritative source for numbers.)

Google's main advertising product is known as Adwords and is composed of 11 ads present on any page listing the results of a search query. The pricing model is one of "pay-per-click" (PPC) where advertisers only pay if a browser clicks on their ad. Advertisers offer a bid of the maximum PPC they are willing to pay along with a daily budget of the maximum amount they are willing to spend. Generally, there are multiple advertisers bidding on a given search in Google. Google has an internal algorithm designed to maximize their profit by selecting the order in which ads are displayed. (Ads in positions below 11 are placed on later pages of the search results, a very undesirable position.)

One great advantage of Internet advertising vs. traditional media is that ad-

vertisers have no cost to modify ad creatives, and receive feedback on resulting performance very quickly. Advertisers are generally unsure of the optimal ad creative, unsure of the ideal search phrases to bid for, and unsure of the optimal bid PPC price. Furthermore, advertisers may intentionally have different ad variants or keyword targets in order to capture a broader demographic. The resulting scenario is that most advertisers have a large portfolio of many ads aimed at many keywords.

The competitive structure of PPC bidding, combined with uncertainty of optimal ad variant or keyword choice leads to a very complex optimization problem well beyond the traditional ad management skills of most businesses. In chapter 5, I discuss the situation in detail and propose an algorithm to optimize parameter settings given the stochastic nature of the system.

CHAPTER 2

A Hierarchical Bayesian Analysis of Horse Racing

2.1 Introduction

Horse racing is the most popular sport in Hong Kong. Nowhere else in the world is such attention paid to the races and such large sums of money bet. It is literally a "national sport". Popular literature has many stories about computerized "betting teams" winning fortunes by using statistical analysis.[Kap03] Additionally, numerous academic papers have been published on the subject implementing a variety of statistical methods. The academic justification for these papers is that a parimutuel game represents a study in decisions under uncertainty, efficiency of markets, and even investor psychology. A review of the available published literature has failed to find any Bayesian approach to this modeling challenge.

This study will attempt to predict the running speed of a horse in a given race. To that effect, the coefficients of a linear model are estimated using the Bayesian method of Markov Chain Monte Carlo. Two methods of computing the sampled posterior are used and their results compared. The Gibbs method assumes that all the coefficients are normally distributed, while the Metropolis method allows for their distribution to have an unknown shape. I will calculate and compare the predictive results of several models using these Bayesian Methods.

2.2 Overview of Parimutuel Racing

At the racecourses in Hong Kong, the games are truly parimutuel. The betters all place their bets in a "pool" which is subsequently divided amongst the winners immediately at the end of each race. Various pools exist representing different betting combinations, but for this paper I will focus on the "win pool" which represents bets on a given horse to win the race. Unlike a casino, the track does not bet agains the public but takes a fixed percentage from each betting pool. (18% in Hong Kong.) The pool is divided amongst the winning betters proportionally, based upon the amount they bet. A large tote-board at the track displays the expected winnings per dollar bet for each horse. These are commonly called the "odds", and are often mistakenly interpreted, by naive betters, as a horse's probability of winning. What the posted odds do represent are a measure of the aggregate public opinion about a horse's likelihood to win the race. Empirical study shows that there is a 40% correlation between the public's opinion as represented by payoff odds and a horse's finishing position. The market may be considered weakly efficient.

2.3 Literature Review

Since the advent of horse racing, people have searched for a way to profit from the game. In 1986, Boltman and Chapman describe a multinomial logit model that forms the basis for most modern prediction methods.[BC86]. In that paper they describe a logistic regression model for predicting the "utility" of a horse. Mildly positive results (profits) were produced.

In 1994 Chapman published a second paper that refined the concepts of his first paper, while applying it to the horse racing industry in Hong Kong.[Cha94] He compared his predicted results to the public's and found that a combination of

the two produced the most profitable results. In 1994, Bill Benter published what many consider to be the seminal work on the subject titled, "Computer Based Horse Race Handicapping and Wagering Systems"[Ben94]. In the paper, Benter develops a two-stage prediction process. In stage one, he uses a conditional logit to calculate the "strength" of a horse. In the second stage, he combines the strength measure with the public's predicted probability using a second conditional logit function. Benter reports that his team has made significant profits during their 5 year gambling operation. (Unlike the other academics discussed here, Benter actually lived in Hong Kong and conducted a real betting operation.)

In 2007, Edelman published an extension of Benter's two-stage technique. Edelman proposes using a support vector machine instead of a conditional logit for the first stage of the process. Edelman's rationale is that a SVM will better capture the subtleties between the data. He theorizes that the betting market is near-efficient and that the bulk of the information about a horse is already contained in its Market odds for the race. His method simplifies Benters in that it only combines odds from a horse's last race with the outcome and conditions of that race and the conditions of the race today.

Lessman and Sung, in 2007 then expanded on Edelman's work by modifying the first-stage SVM process[LSJ07]. They theorized that because only jockey's in the first few finishing positions are trying their hardest, information from later finishers is not accurate as they are not riding to their full potential. The authors develop a data importance algorithm named Normalized Discounted Cumulative Gain where they assign weights to horse's data as a factor of their finishing position. The result is that data from the first place finishers is more important than the latter finishers. This NDCG is used to tune the hyperparameters of the SVM which is then subsequently used as part of the traditional two-stage model.

In 2009, Lessman and Sung published another paper expanding on their work from 2007[LSJ09]. Where previous work has focused on regression of finishing po-

sition, they chose to train an SVM based on classification [win,lose] of race results. Their argument for this approach is that it eliminates pollution of the data by potentially corrupt rank orderings, especially among minor placings. Additionally, they pre-process the data by standardizing the continuous variables *per race*.

2.4 Data Collection and Description

Historical data from September 2005 through December 2009 were downloaded directly from the Hong Kong Jockey Club's official website. The data are 36,006 observations from 2,973 distinct horse races. (A race may have anywhere from 8 to 14 entrants.) The data was loaded into an SQL based database (MySQL). From this database of historical data, the variables were calculated and formatted into a second CSV file, using a Perl script, appropriate for direct import into R.

The single outcome variable used in this study, is the running speed of a horse expressed in meters per second. A detailed description of the covariates is included in The Code Book 2.2 and a sample of the data is included in Table 2.

There are two race courses in Hong Kong (Sha Tin and Happy Valley). Sha Tin has two separate tracks (turf and dirt), so there are a total of three possible tracks a race may compete on. Furthermore, horses are divided into "classes" based upon ability. The racing stewardship attempts to create a fair race by grouping horses of similar ability into the same class. Lastly, there are several distances for which races are run. All combined, there are 73 different combination of course, track, class and distance, each of which will be referred to as a *race profile*. The distribution of speed run in each profile is notably different. This may be visualized by overlaying plots of the density of speed of all race profiles onto a single plot. (Figure 2.1) and the boxplots of speed stratified by profile(Figure 2.2)

The distinctly different shape of the speed distributions suggest that a Bayesian hierarchical regression model would be well suited for this study. Following the

methods of Lessman And Sung.[LSJ09], who centered their data per race, I centered the data *per profile* using the following formula:

$$\tilde{X}_{ki}^{j} = \frac{X_{ki}^{j} - \bar{X}_{k}^{j}}{\sigma_{k}^{j}} \tag{2.1}$$

Where $\tilde{X}_{ki}^{j}$ (X_{ki}^{j})denotes the new (original) value of attribute t of runner i in profile j and the mean $\bar{X}$ as well as the standard deviation σ_{k}^{j} are calculated over the horses in profile j.

Speed, the dependent variable shows some correlation with the other variables, as described in Table 2.1. The data was divided into a *training set*, which consists of races prior to January 1st, 2009, and a *test set* consisting of races run during 2009. A Bayesian MCMC approach will be used to estimate the running speed of a horse, based on the covariates in the training set. Then model performance will be tested on the test data set.

2.5 The Hierarchical Model

The goal is to capture and describe the between-profile heterogeneity of the observations. As Hoffs[Hof09] gives in Chapter 11, the within-profile sampling model is:

$$Y_{ij} = \beta_{j}^{T} x_{ij} + \epsilon_{ij}, \{\epsilon_{ij}\} \sim \text{i.i.d normal}(0, \sigma^{2}) \tag{2.2}$$

Where x_{ij} is a vector of variables for observation i in group j, β_j are the coefficients of the regression, and ϵ_{ij} are errors.

Which gives the within-group regression model of:

$$\beta \sim \text{MVN}(\theta, \Sigma) \tag{2.3}$$

$$Y_{i,j} = \beta_{i,j}^{T} x_{ij} + \epsilon_{ij} \tag{2.4}$$

$$= \theta^{T} x_{i,j} + \gamma_{j}^{T} x_{ij} + \epsilon_{ij} \tag{2.5}$$

With Θ and β are fixed and unkonw parameters to be estimated, and γ_j representing random effects that vary for each group.

2.6 Priors

An important step in Bayesian modeling is the calculation of the prior. An OLS regression was performed for each group and the resulting β_j from each group were used as the prior for the Hierarchical model.

$$\beta_j = \theta + \gamma_j \tag{2.6}$$

$$\gamma_1 \ldots, \gamma_j \sim \text{i.i.d Multivariate Normal}(0,\Sigma) \tag{2.7}$$

2.7 Conditional Distribution

$$Var\left[\beta_j|y_j, X_j, \sigma_j^2, \theta, \Sigma\right] = (\Sigma^{-1} + X_j^T X_j/\sigma^2)^{-1} \tag{2.8}$$

$$E\left[\beta_j|y_j, X_j, \sigma_j^2, \theta, \Sigma\right] = (\Sigma^{-1} + X_j^T X_j/\sigma^2)^{-1}(\Sigma^{-1}\theta + X_j^T y_j/\sigma^2) \tag{2.9}$$

$$\{\theta|\beta, \Sigma\} \sim \text{MVN}(\mu_m, \Lambda_m) \tag{2.10}$$

$$\Lambda_m = (\Lambda_0^{-1} + m\Sigma^{-1})^{-1} \tag{2.11}$$

$$\mu_m = \Lambda_m(\Lambda_0^{-1}\mu_0 + m\Sigma^{-1}\bar{\beta}) \tag{2.12}$$

$$\{\Sigma|\theta, \beta\} \sim \text{inverse-Wishart}(\nu_0 + m, [S_0 + S_\theta]^{-1} \tag{2.13}$$

$$\Sigma_\theta = \sum_{j=1}^{m}(\beta - \theta)(\beta - \theta)^T \tag{2.14}$$

2.7.1 Fitting the model with MCMC

2.8 Implementation of Gibbs method

First, I wrote a custom R script to simulate draws from the posterior using the Gibbs method. The initial tests showed some autocorrelation from the MCMC

chains, so the code was adjusted to only sample one out of every 10 runs of the chain. A total of 300,000 iterations through the chain produced 30,000 samples from the posterior. The chain converged well after 200,000 iterations, so the final 100,000 were used as samples form the converged posterior. Storing one out of every 10 iterations gave me a chain of 10,00 draws from the converged posterior. Additionally, I calculated the Residual Sum of Squared Error (RSS) for each run and stored the results along with each posterior sample. This chain of 30,000 RSS errors allowed me to track the accuracy of the inference. The residual sum of squares for this method converged to: 988.637 which gives a predicted σ^2 of .0350 for an individual horse.

2.9 Implementation of Metropolis Hastings method

Next, I wrote custom R code to generate draws from the posterior using Metropolis Hastings. Following the same model as the Gibbs technique above, The Metropolis acceptance ratio was used as described by the formula:

$$r = \frac{p(\theta^*|y)p(\theta^*)}{p(\theta^s|y)p(\theta^s)} \tag{2.15}$$

Initially the acceptance ratio was low, so the variance of Σ was adjusted through trial and error to $0.5 \cdot \Sigma$ which produced an reasonable acceptance ratio of 0.54. A total of 200,000 iterations were run. The chains converged after 100,000 iterations, so the resulting100,000 were used as samples from the converged posterior. SInce I am storing 1 out of every 10, the ending posterior chain was 10,000 long. The residual sum of squares for the this method was 1033.35 which gives a predicted σ^2 of 0.0363. This was, surprisingly, slightly higher than the RSS from the Gibbs technique.

2.10 Results and conclusion

Predicted speeds were calculated for each horse in the test data set, to measure predictive ability of the Gibbs model. 10,000 β were drawn and then used in a regression with the covariates for each horse. The maximum a-posteriori (MAP) of the resulting regression for each horse was stored as "predicted speed". The variance of this predicted speed was 0.0397. The horse with the fastest predicted speed won his race 21.63% of the time. This is better than a random choice, which would produce an expected winner between 7.14% and 12.5% (Depending on the number of horses in a race). However, simply betting on the horse with the highest predicted speed was not enough to profit. (The ultimate goal is not to guess winners, but generate profit.)

As a further step, a conditional logit as calculated to combine our predicted speed with the public's odds estimate. As this is the "standard" for other predictive models. The coefficients for that model were 6.4186 for the public odds and 4.0541 for the hierarchical Bayesian predicted speed.

As a further test of performance, the expected value for each bet was calculated as $ev = \text{payoff if won} x \text{probability of winning}$. There were 3,323 horses in the test set with positive expected value (out of 8,618 possible.) If \$1 had been bet on each horse with a positive expected value, the return would have been 2919 resulting in a net loss of \$404 (-12.15%). While the predictive errors are small, the model is still not good enough to generate profit without further refinement.

2.11 Tables and Figures

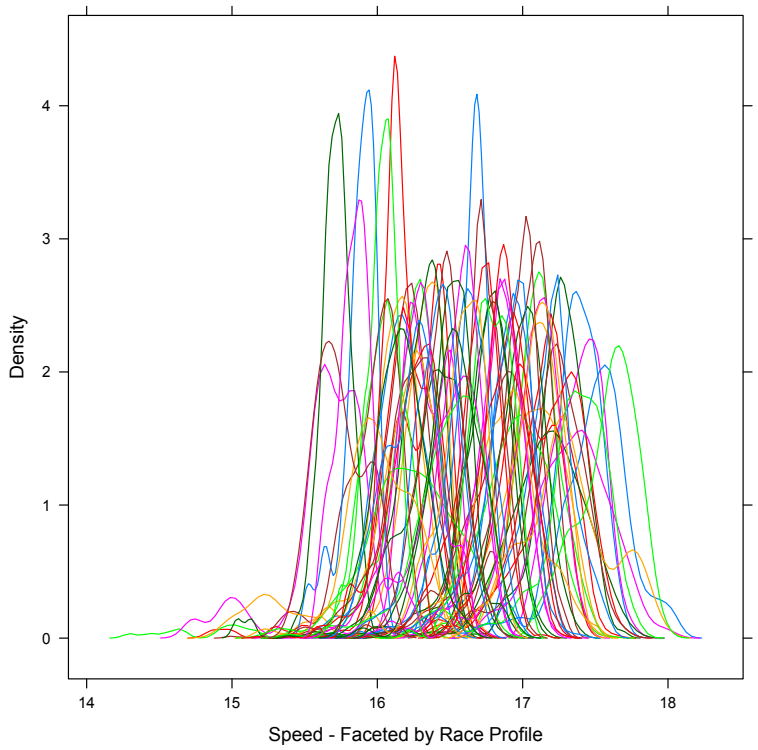

Figure 2.1: Density distributions of speed stratified by profile

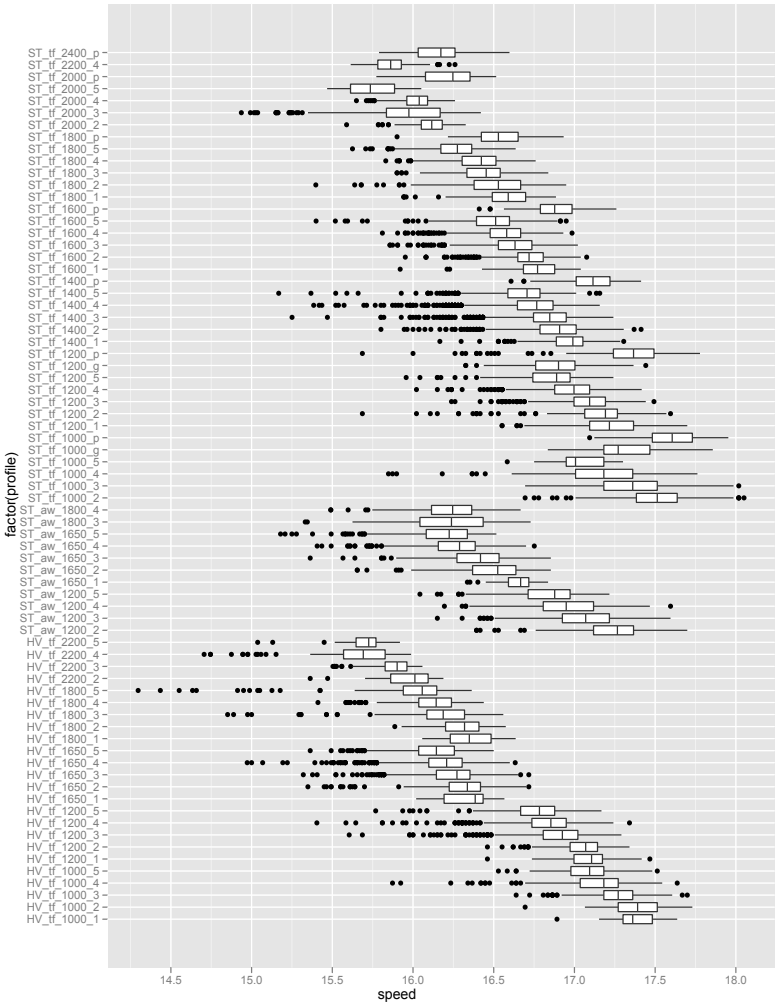

Figure 2.2: Boxplots of speed stratified by race profile

Table 2.1: Correlation of variables to speed

Name	Correlation
last_rank	0.077898638
last_run_1	0.091421415
last_run_2	0.106251733
last_run_3	0.110888381
last_odds_prob	0.111767807
last_distance	-0.562025513
last_weight	0.035323525
last_draw	0.007378011
last_speed	0.563767210
last_percentage	-0.016219487
perc_won	0.159669358
last_4_perc	-0.024847700
last_4_rank	0.128202630
last_4_odds_prob	0.147430995
rest	0.136668197
distance	-0.799783155
weight	0.005454088
draw	0.022529607
total_races	-0.187955730
bad_runs	-0.077079847
jockey_rides	-0.030171224
dist_last_30	-0.217824843

Table 2.2: Data Code Book

Name	Mean	SD	Low	High
speed	16.648026	0.421332	14.297061	18.050542
odds_prob	0.082121	0.081307	0.002158	0.745455
last_rank	0.000457	1.000038	-1.672634	1.577826
last_run_1	0.000310	0.999845	-1.613067	1.594531
last_run_2	0.000365	0.999841	-1.623586	1.597088
last_run_3	0.000191	0.999820	-1.657603	1.597270
last_odds_prob	0.000544	0.999850	-0.976207	8.181944
last_distance	-0.000505	0.999136	-1.679878	3.419010
last_weight	-0.000349	1.000267	-2.866383	1.830325
last_draw	0.000408	1.000169	-1.566456	1.879832
last_speed	0.000237	0.999575	-5.522472	3.813677
last_percentage	0.000407	0.999548	-7.595383	2.543021
perc_won	-0.000762	0.996582	-0.766225	7.839123
last_4_perc	0.001131	0.997601	-7.240364	2.922523
last_4_rank	0.000107	0.999540	-2.348644	2.335103
last_4_odds_prob	0.000203	0.999670	-1.132926	9.931562
rest	0.000100	1.000661	-0.991545	5.016491
distance	-0.002648	0.996482	-1.726967	3.412664
weight	0.000140	0.999623	-2.827238	1.807332
draw	0.000548	1.000079	-1.568536	1.882718
total_races	0.001505	1.000234	-1.321604	4.875548
bad_runs	0.000264	1.000246	-0.293411	8.160743
jockey_rides	0.000219	0.999542	-0.651017	10.310366
dist_last_30	0.000271	0.999912	-1.167798	6.151917

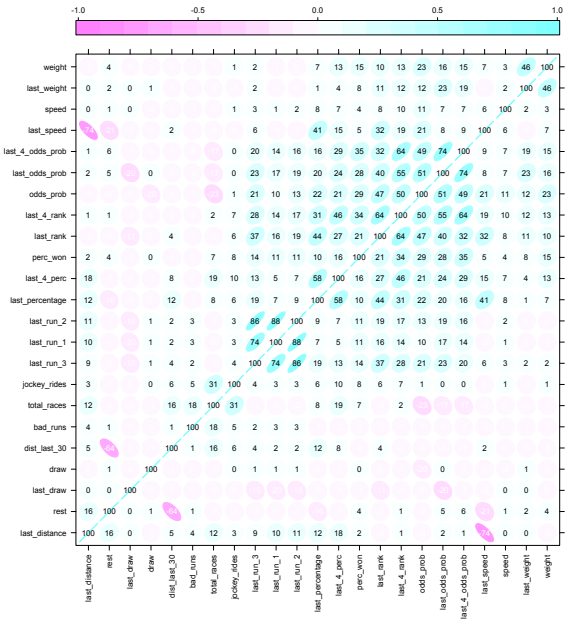

Figure 2.3: Graphical representation of correlation of variables

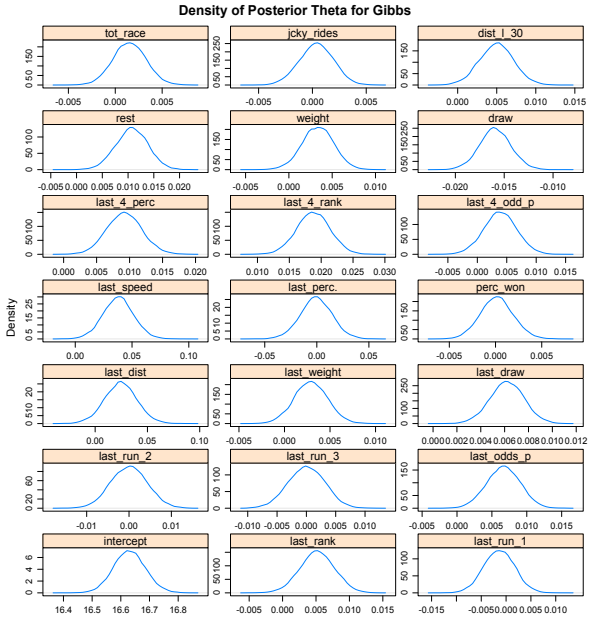

Figure 2.4: Posterior density of theta for predictor variables - Gibbs

Table 2.3: Sample of data

	1	2	3	4	5	6	7	8	9	10
1	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
last_rank	-0.98	-1.34	1.48	-1.29	0.56	1.53	1.53	0.75	-0.30	-0.37
last_run_1	0.15	-0.36	-1.60	-0.72	1.32	0.87	1.12	-0.36	0.13	1.61
last_run_2	-1.30	0.12	-1.30	-1.59	1.58	-1.59	1.10	-0.61	-1.10	1.01
last_run_3	-1.37	0.36	0.13	-1.37	1.62	-0.91	1.37	-0.40	-0.66	1.32
last_odds_prob	0.40	0.16	0.63	-0.39	-0.78	-0.62	-0.28	-0.14	-0.91	0.16
last_distance	-1.12	-1.12	-1.12	1.06	1.06	0.82	-0.15	-1.12	-1.12	1.06
last_weight	-0.15	1.36	-0.83	0.40	-0.56	0.81	0.67	-0.83	-1.11	0.54
last_draw	0.95	-1.12	-1.12	0.95	-0.86	1.73	-0.08	-1.38	0.18	0.18
last_speed	0.10	0.53	0.40	-1.30	-1.09	-0.08	0.65	0.04	-0.44	-1.75
last_percentage	-0.43	-1.00	-0.88	0.40	0.76	0.14	0.72	-1.79	-2.57	-0.38
perc_won	-0.90	-0.90	-0.90	-0.90	-0.90	-0.90	-0.90	-0.90	-0.90	0.94
last_4_perc	0.55	-0.96	-0.00	-0.37	-1.28	-0.73	-0.29	-1.69	-2.03	0.08
last_4_rank	0.00	-0.32	1.25	-0.33	-0.50	0.96	1.31	1.16	-1.24	0.20
last_4_odds_prob	0.77	0.42	0.50	0.65	-0.35	0.21	-0.24	-0.34	-1.07	0.27
rest	0.72	1.52	1.00	2.53	0.90	1.16	1.16	0.79	0.79	3.80
weight	-0.20	0.26	1.03	-0.65	0.26	0.88	0.72	0.88	-0.20	-0.65
draw	-0.33	1.42	0.67	0.17	0.92	-1.34	1.67	-1.59	0.42	-1.08
total_races	-0.24	-0.30	-0.24	-0.54	-0.97	-0.24	-0.67	-1.10	-1.16	-0.48
jockey_rides	-0.19	-0.59	-0.19	-0.59	-0.39	-0.39	-0.59	-0.59	0.21	-0.59
dist_last_30	-1.06	-1.06	-1.06	-1.06	-1.06	-1.06	-1.06	-1.06	-1.06	-1.06

CHAPTER 3

Predicting Horse Race winners through A Regularized Conditional Logistic Regression with Frailty

3.1 Introduction

Since first proposed by Bill Benter in 1994, the Conditional Logistic Regression has been an extremely popular tool for estimating the probability of horses winning a race. We propose a new prediction process that is composed of two innovations to the common CLR model and a unique goal for parameter tuning . First, we modify the likelihood function to include a "frailty" parameter borrowed from epidemiological use of the Cox Proportional Hazards model. Secondly, we use a LASSO penalty on the likelihood, where *profit* is the target to be maximized. (As opposed to the much more common goal of maximizing likelihood.) Finally, we implemented a Cyclical Coordinate Descent algorithm to fit our model in high-speed parallelized code that runs on a Graphics Processing Unit (GPU), allowing us to rapidly test many tuning parameter settings. Historical data from 3681 races in Hong Kong were collected and a 10-fold cross validation was used to find the optimal outcome. Simulated betting on a hold out set of 20% of races yielded a return on investment of 36.73%.

Conditional logistic regression has remained a mainstay in predicting horse racing outcomes since the 1980's. In this paper, we propose and apply novel

modifications of the regression model to include parameter regularization and a frailty contribution that exploits winning dividends. The model is trained using 4 years of horse racing data from Hong Kong, and then tested on a hold-out sample of races. Simulated betting produces a return on investment significantly higher than other published methods with Hong Kong races.

3.2 Background

3.2.1 Horse Racing

Horse racing is one of the oldest forms of gambling in existence. It is conducted as a parimutuel style wagering contest. For a given race, each person bets on his choice to win the race. All monies bet are grouped into a pool. The racetrack takes a fixed percentage of the money pool in each race as its profit. The remaining funds are proportionally distributed amongst the bettors, who selected the winning horse, according to the amount they bet. A bettor's potential winnings, if a chosen horse wins, are erroneously named the "odds" on a horse. These "odds" do not represent the true odds or probability in the statistical sense, but are simply an indication of the percentage of bettors who favor the given horse. We prefer to think of these "odds" as *bettor implied confidence*.

This poses an interesting opportunity. Traditional casino games generally exist in a setting where the player bets against the establishment in a negative expectation game. Even with perfect skill, the player is bound to eventually lose his or her bankroll. With parimutuel betting, players are betting against each other, and it is possible to achieve a positive expectation bet if a winning horse can be chosen with more accuracy than the public. Many books and academic papers have been published in an attempt to model this system. [Bay07] [BC86] [Hau08] [Kap03] [LSJ07] [LSJ09] [LSJ10]

3.2.2 Conditional Logistic Regression

Logistic regression is a form of a generalized linear model for working with binary outcomes [MN89]. In simple terms, logistic regression models the probability of an event occurring. The regression is linear in the log of the event odds. With the covariates X represented by an $N \times K$ dimensional matrix, with each row representing the covariates of a single outcome (Here a single horse's features.), β as a column of k regression coefficients, and p_i as the probability of a positive event outcome, the format of logistic regression may be represented as:

$$\ln\left(\frac{p_i}{1-p_i}\right) = x_i\beta \tag{3.1}$$

$$\left(\frac{p_i}{1-p_i}\right) = e^{x_i\beta} \tag{3.2}$$

The probability of an event occurring may be then be represented by the inverse logit transform

$$p = \frac{1}{1+e^{-X\beta}} \tag{3.3}$$

Horse racing is an event where there is a single winner within a group of race competitors. A race $r = \{1, 2, \ldots, R\}$ is run between several horses $h = \{1, 2, \ldots, H\}$ with a single horse winning. The features of horse h in race r may be represented in a k dimensional vector identified by X_{rh}, The coefficients of the winning horse may then be represented by X^w_{rh}, with the superscript w indicating that horse h won race r. For each horse h in a race r, the estimated probability of winning that race is "conditioned" on the winning probabilities within race r summing to 1.

$$p_{rh} = \frac{e^{X_{rh}\beta}}{\sum_{h\in r} e^{X_{rh}\beta}} \tag{3.4}$$

The likelihood and log likelihood over all races then become

$$\ell(\beta) \propto \prod_{r=1}^{R} \frac{e^{X_{rh}^{w}\beta}}{\sum_{h \in r} e^{X_{rh}\beta}} \tag{3.5}$$

$$\ln(\beta) \propto \sum_{r=1}^{R} \left[X_{rh}^{w}\beta - \ln(\sum_{h \in r} e^{X_{rh}\beta}) \right] \tag{3.6}$$

3.2.2.1 L_1 penalized conditional logistic regression

Tibshirani [Tib96] proposed a method for variable selection for a Cox proportional hazards model through penalization of the regression coefficients under an L_1 norm. As the partial likelihood of the Cox model takes the similar form to the conditional logistic regression likelihood, and has the same partial likelihood. So we apply the same likelihood and shrinkage method.

Tibshirani describes penalizing the log likelihood, with a LASSO prior on β such that $\sum |\beta_j| \leq \lambda$ where λ is a user selected parameter. As λ is increased, the values of *beta* are pushed toward zero. [Tib97] By combining this shrinkage technique with cross validation, a model may be derived that maximizes likelihood while resisting over-fitting.

The log likelihood with this shrinkage factor is then

$$\ln(\beta) \propto \sum_{r=1}^{R} \left[x_{rh}^{w}\beta - \ln(\sum_{h \in r} e^{x_{rh}\beta}) \right] - \lambda \sum_{k} |\beta_k| \tag{3.7}$$

3.3 Current models for predicting Horse Racing

Applying a conditional logistic regression to a horse race was first suggested by Boltman and Chapman [BC86] in 1986. Subsequently, it has been used, in a modified form by a majority of published work afterward. [Ben94] [ED07] [LSJ07] [LSJ09] Each successive author has used a "two stage" form of this model. A "strength" factor, indicated by α, is calculated and then combined with the payoff

dividend (generally in the form of a conditional logistic regression.) The likelihood of this conditional logit is

$$\prod_{r=1}^{R} \frac{e^{\alpha_{rh}^{w}\beta_1 + p_{rh}^{w}\beta_2}}{\sum_{h \in r} e^{\alpha_{rh}\beta_1 + p_{rh}\beta_2}} \tag{3.8}$$

The concept here is to combine a previous model predicted "strength" of a horse, represented by $a = \{1, 2, \ldots, A\}$ with the "odds implied probability" for a horse $p = \{1, 2, \ldots, P\}$. Payoff dividends for a horse can be converted to implied probabilities with the formula: $p(x_h) = \frac{1}{d_h}$. Recall that this value is best described as the *bettor implied confidence*, as it is not a true probability of outcome, but an aggregate estimate of the public's opinion. Additionally, the "odds" reported at the racetrack represent the payoff with a track-take deducted. (This deduction is the profit of the racetrack hosting the event.) Subsequently, the dividend implied probability can be be calculated with the adjusted formula $p(x_h) = \frac{1-\tau}{d_h}$, with τ representing the track take from the parimutuel pool.

A series of authors have published successive variations to the original Boltman and Chapman model. Each publication has attempted to improve the calculation of the horse "strength" measure that is represented by a in the conditional logistic regression equation 3.8.

- in 1994 Benter [Ben94] used a conditional logistic regression to estimate horse strength.
- in 2006 Edelman [ED07] used a support vector regression on the horse's finishing position to estimate horse strength.
- in 2008 Lessmann and Sung [LSJ07] use a support vector classifier, and subsequent distance from the hyperplane to estimate horse strength.
- in 2010 Lessmann and Sung [LSJ09] use CART to estimate horse strength.

Despite different methods for calculating the "strength" of a horse, all of the authors used equation 3.8, a standard conditional logistic regression, as a second stage of their prediction algorithm.

3.4 Application of a Frailty Model to Horse Racing

As noted previously, the conditional logistic regression likelihood shares similarities to a Cox Proportional Hazards model. The Cox model has been extended to account for the "frailty" of a subject, first suggested by Gillick [Gil01]. This frailly was introduced to account for the fact that while a hazard may be proportionally equal for the population being studied, an individual may be more or less sensitive. This individual sensitivity is termed "frailty" and is represented here by ω. The likelihood involving a frailty term is then:

$$\ell(\beta) \propto \prod_{r=1}^{R} \frac{e^{X_{rh}^{w}\beta+\omega_{rh}^{w}}}{\sum_{h\in r} e^{X_{rh}\beta+\omega_{rh}}} \tag{3.9}$$

Our empirical analysis shows about a 40% correlation between public estimated probability and the true probability of a horse winning. In can be reasoned that the public is not completely naive, but has some degree of knowledge in picking winners. Subsequently, we propose using the public's knowledge to strengthen model accuracy. The "desirability" index w_{rh} for a horse h in race r may be calculated as a function of the posted "dividend" d_{rh} of a horse.

$$\omega_{rh} = 1 - (\frac{1}{d_{rh}}) \tag{3.10}$$

In essence, this can be considered a reverse-frailty model, as we are giving the horses with higher paying dividends a higher score. We are, in effect, creating a "strength" model instead of a frailty model. Profit from a race is maximally realized when betting on a horse that the public does not expect to win. Horses with the highest paying dividends, by definition, have the lowest public confidence.

Looking for betting opportunities where the public is wrong is a strategy toward profit. By setting our strength(frailty) as the opposite of public confidence, we are looking for those horses we have a high confidence of winning that also pay high dividends. (Low public confidence of winning.)

The modified likelihood and log likelihood then become:

$$L(\beta) \propto \prod_{r=1}^{R} \frac{e^{X^w_{rh}\beta+\omega^w_{rh}}}{\sum_{h\in r} e^{X_{rh}\beta+\omega_{rh}}} \tag{3.11}$$

$$L(\beta) \propto \sum_{r=1}^{R} \left[X^w_{rh}\beta + \omega^w_{rh} - \log(\sum_{h\in r} e^{X_{rh}\beta+\omega_{rh}}) \right] \tag{3.12}$$

Our model includes 186 variable, many of which may be correlated. In an effort to avoid over-fitting, and and reduce the effect of colinearity, a regularization parameter is included. Applying the L1 shrinkage factor as described by Tibshirani gives a regularized log likelihood of:

$$L(\beta) \propto \sum_{r=1}^{R} \left[X^w_{rh}\beta + \omega_{rh} - \log(\sum_{h\in r} e^{X_{rh}\beta+\omega_{rh}}) \right] + \lambda \sum_{k=1}^{K} |\beta_j| \tag{3.13}$$

Using cyclical coordinate descent to fit this log likelihood requires the calculation of the first and second partial derivatives for each β_j

$$\frac{\partial L}{\partial \beta_j} = \sum_{r=1}^{R} x^w_{rh} - \frac{\sum_{h\in r} x_{rh} e^{x_{rh}\beta+\omega_{rh}}}{\sum_{h\in r} e^{x_{rh}\beta+\omega_{rh}}} + \lambda \tag{3.14}$$

$$\frac{\partial^2 L}{\partial \beta_j^2} = \sum_{r=1}^{R} \left[\frac{\sum_{h\in r} x^2_{rh} e^{x_{rh}\beta+\omega_{rh}}}{\sum_{h\in r} e^{x_{rh}\beta} + \omega_{rh}} - \left(\frac{\sum_{h\in r} x_{rh} e^{x_{rh}\beta+\omega_{rh}}}{\sum_{h\in r} e^{x_{rh}\beta+\omega_{rh}}} \right)^2 \right] \tag{3.15}$$

3.5 Choice of Shrinkage Factor λ

Ultimately, the goal of this process is not to predict winners with the most accuracy, but to maximize profit, which is not necessarily the same. This paper uses

a frailty modified conditional logistic regression that is regularized by a shrinkage factor λ. While all of the coefficients may be calculated using the aforementioned cyclical coordinate descent, the choice of λ is left up to the user. We performed a 10-fold cross-validated training process, with a random 20% of the data held out for testing each time, for different values of λ, calculating the mean return-on-investment for the hold out set of each 10-fold run. λ is chosen to maximize ROI, which is defined as:

$$\text{ROI} = \frac{1}{10}\sum_{i=1}^{10}\left[\sum_{r}^{R} b_{rh}^{w} \cdot d_{rh}^{w} - \sum_{h \in r} b_{rh}\right] \tag{3.16}$$

with b_{rh}^{w} representing the amount bet on the winning horse h in race r, b_{rh} representing the amount bet on horse h in race r, and d_{rh}^{w} representing the dividend collected if horse h wins race r.

3.6 Parallel Computing

Given the large data set of 40,000 cases of 186 variables each, computing the regularized logistic regression can be slow even on state of the art hardware. Additionally, adding the polynomial expansion magnifies the 186 variables to 36100. The optimal LASSO penalty, λ can't be calculated ahead of time, so must be discovered by iterating over a wide range of values. We performed a 10-fold cross validation for each of 100 different levels of λ resulting in 1,000 runs of the model fitting regression.

In order to speed up the model fitting process, and to allow for faster experimentation, custom software was written in C++. This software takes advantage of the Graphics Processing Unit (GPU) in the author's desktop computer. A basic level GPU costs about $250 and provides hundred of CPU cores optimized for parallel processing and are quickly finding many uses in statistics [ZLS10] [SSZ12]. Using the Thrust[HB10] library provided a fast and easy way to harness this computing power. (Specifics of the software used will be discussed in a forthcoming

paper.)

Fitting this form of model involves the cyclical update of each coefficient many times. Each of those updates may be described as a series of transformations and subsequent reductions. For example, one component needed is the sum of the exponent of the linear term for each race:

$$\sum_{h \in r} e^{x_{rh}\beta + \omega_{rh}} \tag{3.17}$$

A non-parallelized solution would be to create a loop that computes the exponent of each item, then a second loop that sums these exponents by race. An intermediate solution would be to use the OpenMP[Boa11] library to take advantage of running multiple threads on the same CPU. However this is still limited in terms of parallelization benefit. Parallelization on the GPU is fairly simple with a few strategic calls to the Thrust library. As the exponent steps are independent, they may all be performed in parallel. The subsequent sums by race are also independent, so they may also be computed in parallel.

3.7 Results

Data from 3681 races in Hong Kong were collected from the Hong Jockey Club's official race records. 2944 races were used to learn the model through cyclical coordinate descent as described in algorithm 1. The data were 186 covariates describing each horse's past performance. (i.e. days of rest since last race, change in weight carried, average speed in previous races, and if the horse had gained or lost weight.) 737 races were withheld to use for testing the predictive strength of the model.

First, a two stage conditional logistic model, as described by Benter [Ben94] was fit to establish a base level of performance. Bets were placed on any horse

with a positive expectation, defined as: $Pr_{rh}(win) \cdot \text{dividend}_{rh} > 1.0$.

Return on investment (ROI) was calculated using equation 3.16. The Benter two-stage model produced a return on investment of 14.2%. Next, we fit our proposed modified frailty model with the same data. λ was varied between 1 and 100 to find the best resulting profit. With a λ of 7, our model produced a maximum ROI of 36.73% which is a significant improvement over the Benter model. As third version was tried, by adding the square of each covariate to the data set x^2 in an attempt to capture some non-linearity. However, the ROI produced was not as good as the main model. A plot of ROI as a function of λ for both models is included as plot 3.1

Parallelization of the model fitting algorithm via the GPU was compared to using the OpenMP library across a range of operating threads. The algorithm took 494 milliseconds to fit using the GPU solution. The openMP solution speed ranged from a high of 3995 milliseconds with a single thread to 1324 milliseconds using 8 threads. The GPU solution was faster than the best openMP solution by a factor of 2.68. A plot demonstrating the model fitting speeds is included as plot 3.2

3.8 Conclusion

The conditional logistic regression model has been a standard tool in horse race modeling. We have found that this continues to hold true. However, much of the past work revolves around how to best estimate a "strength" score that is fed into this form of model, and then maximizing accuracy probability estimates. By both combining a calculated inverse-frailty score, and changing the goal to directly maximizing profit, we are able to repeatedly generate a significantly higher return on investment.

Fitting a model with number of coefficients repeatedly as we iterate over a

range of tuning parameters can be prohibitively slow. By using a CCD algorithm parallelized to take advantage of a GPU, we achieve a massive speedup, allowing us to quickly experiment with different forms of the model and tuning parameters.

Ultimately, our model innovations and tuning procedure produce a return on investment over 36% over repeated cross-validation trials.

Algorithm 1 Single CCD Step with Bounding Box

Initialize: $\beta_j \leftarrow 0, \Delta_j \leftarrow 1$ for $j = 1, \ldots, d$;

while iteration i not converged **do**

for $j = 1, 2, p$ **do**

Calculate tentative step Δ_{vj}

$$\Delta_{j+} = -\frac{\frac{\partial L}{\partial \beta_j} + \lambda}{\frac{\partial^2 L}{\partial \beta_j^2}}$$

$$\Delta_{j-} = -\frac{\frac{\partial L}{\partial \beta_j} - \lambda}{\frac{\partial^2 L}{\partial \beta_j^2}}$$

$$\Delta_{vj} = \begin{cases} \Delta_{j-} & \text{if } \Delta_{j-} < 0 \\ \Delta_{j+} & \text{if } \Delta_{j+} > 0 \\ 0 & \text{otherwise} \end{cases}$$

$\beta_j \leftarrow \beta_j^{i-1} + \min(\max(\Delta_{vj}, -\Delta_{vj}, \gamma_j))$ Bound on step size

$\gamma_j \leftarrow \max(2|\beta_j|, \gamma_j/2)$ Update upper bound limit

end for

end while

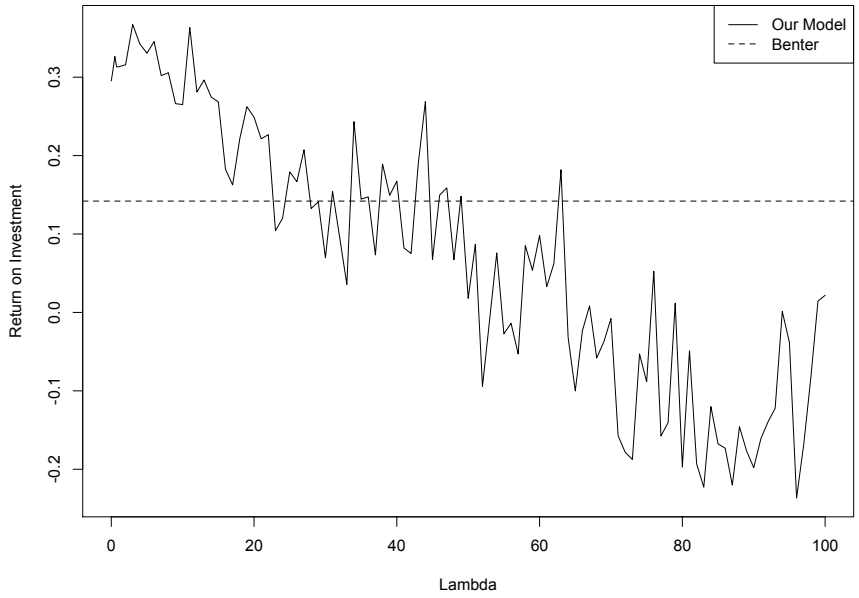

Figure 3.1: Mean ROI Over 10 Fold Cross Validation

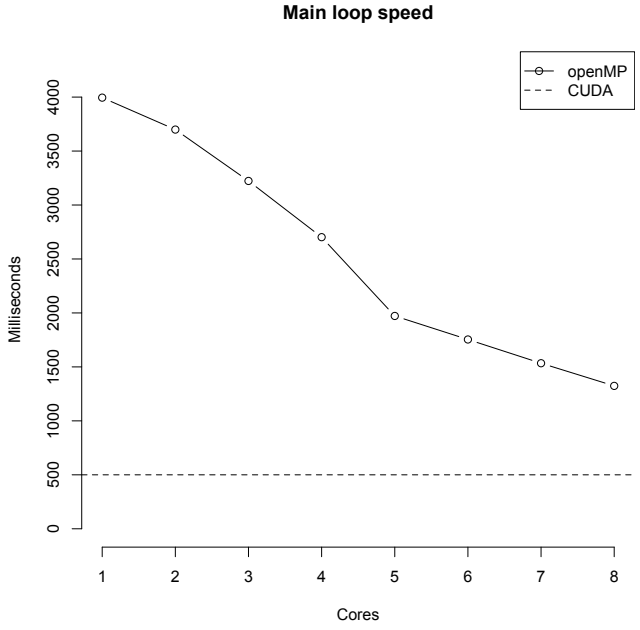

Figure 3.2: Speed of parallelization

CHAPTER 4

An Exploration of using LMS Adaptive Filters in Predicting Movement of a Stock Market Index

4.1 Introduction

The price of a stock index is determined by a weighted average or sum of its component stocks. We explore the application of adaptive filtering techniques derived from an LMS filter to predict the price or direction of the index one day in the future. A two stage algorithm incorporating diffusion and a feed-forward design is developed and then tuned for optimal hyper-parameters.

In finance, the goal of most trading firms or "agents" is to seek an *"edge"* to generate profit. An edge is defined as either a market inefficiency in pricing an asset, or a recognized pattern in asset pricing that allows future prediction. In this paper, I explore the application of adaptive filtering to see if the future price or direction of a stock index can be predicted accurately enough to generate profit. We take the approach of a "data detective", attempting multiple configurations of filtering, diffusion, weighting, and data analysis to find *edge* by comparing a standardized loss measure. Additionally, since the goal of *"edge"* doesn't have a specific definition beyond profit, multiple types of edges are tested for the various configurations explored.

The objective of this paper is to determine if a diffusion network of adaptive filters can reliably predict either future price or direction of a stock index. Various implementation of the LMS filter and NLMS filter are attempted in an exploratory

manner, along with hyper-parameter tuning to seek the optimal filtering and diffusion strategy. Multiple scoring metrics are explored to find a possible financial "edge"

This paper is structured as follows: section 2 gives a brief overview of technical analysis and adaptive filtering. section 3 explains basic concepts, data sources, and scoring metrics. section 4 describes the details of the different filter constructions and analysis. section 5 lists the results. section 6 is a discussion of the findings and suggestions for future exploration section 7 is the conclusion.

4.2 Overview and Related Work

The study of predicting stock prices from their past behavior is generally known as "technical analysis". The focus of this paper is on exploration of filtering techniques, so the the discussion of technical trading will be brief. The reader is referred to a very thorough discussion of technical trading on Wikipedia[wik] if they wish more details on that discipline.

The earliest discussion of this technical trading was made by Joseph de la Vega when looking at Dutch markets in the 17th Century [Veg88]. Since that time, there have been an endless number of theories and models about predicting stock performance based on history. Some of the more traditional tools are moving average, Bollinger bands, "charting", and trend analysis. Despite the popularity of technical trading, the academic community appears to be divided as the relative merit of these techniques.

The Dow Jones company publishes a variety of stock indexes. An index is comprised of a group of stocks related by industry. Some indexes use the weighted average prices of their member stocks, others use the sum of prices. Two of the more well know indexes are the S&P 500 and the Dow Jones Industrial Average, both are designed to measure the direction of the overall U.S. stock market. For

this paper, I chose to look at The Dow Jones Utility stock index, which is comprised of 15 companies in The United States. This index was formed in February of 2000. Shares of the index fund are actively traded on the NYSE while futures and options are actively traded by the Chicago Board Options Exchange.

We could find no previously published references to predicting index performance from filtering on its composite members.

In 2010, X. Zheng and B. Chen[ZC10] did present a paper where they applied filtering to predicting future performance of the S&P 500 index, but the input parameters were a collection of global financial indicators including Interest Rate, Petroleum Price, Baltic Dry Index, CBOE DJIA Volatility, and Exchange Rate. They calculate the exponential weighted moving average of each of these 5 input data, then use a Kalman filter to provide the recursive prediction and updating process.

Lopes and Sayed[LS08] published a paper describing the combination of individual filters through diffusion in order to obtain a more accurate "group" estimate.

Takane, Young, and DeLeeuw[TYL77] explain the concept of Alternating Least Squares. The technique is used to impute missing values by alternating between two least squares regressions. The output of each regression is used as the input of the other. This alternation continues until the resultant values converge.

In this paper, I combine concepts from all three of the mentioned papers in a variety of ways looking for maximum *edge* A simple LMS filter for the stock index, using just it's own historical prices does not track well at all, as evidenced by Figure 4.1 on Page 37

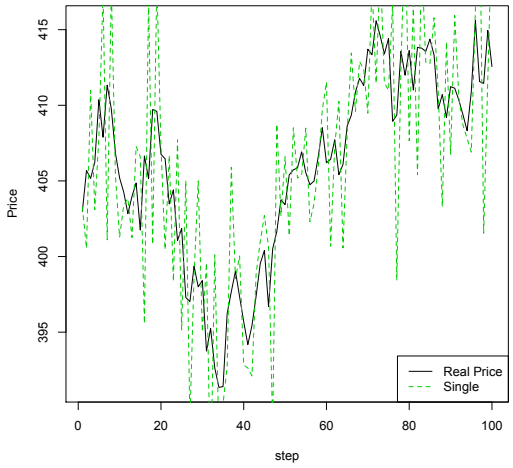

Figure 4.1: "Simple LMS on the stock index"

4.3 Basic Concepts

The Dow Jones Utility index (DJU) was chosen as it is comprised of a relatively small number of stocks making initial exploration easier. The index is composed of 15 companies and is a weighted sum of their prices. Yahoo.com was used as a source for the end-of-day prices of the index and its component stocks from 01/03/2007 until 03/04/2011 giving 1,051 trading days of information. The R programming environment was used for data management, script development, and all testing.

An important step in any financial modeling application is the measurement of success. For this paper, I chose to measure performance of the different algorithms over the last 100 days of trading the index. The initial measure is the sum of squared errors between predicted and actual price.

Borrowing loosely from Takane, Young, and DeLeeuw, I create a two stage process where the output of each stage is the input to the other, thus "alternating" between filters. We call this technique *"Alternating LMS Adaptive Filtering"*. We use the general framework of Diffusion filtering described by Lopes and Sayed as

the first stage filter, and then a Normalized LMS filter on the index for our second stage filter. At each time step, each stage is iterated once. Ultimately, I found that feedback from the second stage to the first reduced accuracy, and the final algorithm was modified so that the first stage only feeds the second stage.

Lopes and Sayed describe the formulation of a network of adaptive filters. They explain how each filter absorbs the weighted coefficients from its neighbors. In particular, their equation 19 is the starting place for the filtering algorithm developed in this paper.

$$\phi_k^{(i-1)} = \sum_{\ell \in N_{k,i-1}} c_{k\ell} \psi_\ell^{(i-1)}, \quad \phi_k^{(-1)} = 0 \tag{4.1}$$

$$\psi_k^{(i)} = \phi_k^{(i-1)} + \mu_k u_{k,i}^* (d_k^{(i)} - u_{k_I} \phi_k^{(i-1)}) \tag{4.2}$$

Their algorithm is for predicting individual "agents" by diffusing values over the group. Our goal, in this paper, is to predict the index which is a weighted collection of the agent's values. It follows that I should be able to derive some weighted collection of the individual stocks to predict the value of the filter.

Our initial two stage algorithm has the following structure:

Algorithm 2 Initial two stage filter algorithm

for t = 1 to Time **do**

  for s = 1 to stocks **do**

    $w_{st} \leftarrow \phi_{t-1} + \mu \cdot u_{st}^* \left[d_{st} - (wi_{t-1} \cdot u_{st}) \right]$

  end for

$\phi_t \leftarrow c \cdot w_{st}$

$wi_t \leftarrow wi_{t-1} + \mu \cdot ui_t^* \left[di_t - (\phi_t \cdot ui_t) \right]$

end for

Where w_{st} is the weight for stock s at time t, μ is the step size, u_{st} are the regressors, d_{st} is the true value for stock s at time t, c is a weighting function for the stock weights used to combine them into ϕ. In the second stage, wi_t is weight

of the index at time t and ui_t are the regressors at time t.

We chose to score the algorithm by the root mean squared error over the last 100 time steps (RMSE)

$$SSQ = \sqrt{\frac{1}{100}\sum_{i=952}^{1051}(y_i - \hat{y}_i)^2} \tag{4.3}$$

Where y_i represents the true index price at time i and $\hat{y}$ is the predicted value. As stock market data is extremely noisy, a second performance metric was developed that ignored price and instead focused on direction. Predicting the index moving up when the index actually moved up, or predicting the index moving down when the index actually moved down would be a valuable result that might be used to generate possible profit.

$$SSD = \frac{1}{100}\sum_{i=952}^{1051}\begin{cases}1 & \text{if } d_i = \hat{d}_i \\ 0 & \text{if } d_i \neq \hat{d}_i\end{cases}$$

Where

$$d_i = \begin{cases}\text{up} & \text{if } d_i > d_{i-1} \\ \text{down} & \text{if } d_i < d_{i-1}\end{cases}$$

So that d represents the *direction* of the price change. (Did the index move up or down since yesterday.) The score SSD represents the percentage of time a given algorithm predicted the correct direction.

A common practice in finance is to look at at the percentage change in price for each time step as opposed the actual dollar amount change. Ultimately an investor doesn't care of an asset moves by \$1, but cares what percentage change it made. (i.e. A \$1 move of a stock priced at \$5 is much more significant than a \$1 move of a stock priced at \$100. The industry term for this percentage change is "Daily Return", often just abbreviated as "Return". The Return is calculated

by:

$$R_i = \frac{p_i - p_{i-1}}{p_{i-1}} \tag{4.4}$$

Where R represents the Return at period i and p represents the price.

This gives us two different performance measures over two different index price formats, represented by the Table 4.1 on Page 40.

	Error Measure		
		SSQ	SSD
Price Meas.	Dollar	SSQ(Dollar)	SSD(Dollar)
	Return	SSQ(Return)	SSD(Return)

Table 4.1: Matrix of performance measurements considered

Each system design was tested against each of the 4 measurement methods to determine if there was a significant advantage. Since the SSQ measures are on different scales, I standardized them by looking at the z-test of the SSQ. The score can be interpreted as, "how many standard deviations of the data is our error."

$$z = RMSE/\sqrt{var(dju)} \tag{4.5}$$

Where $RMSE$ is the Root Mean Squared Error as calculated above and $var(dju)$ is the variance of the prices (or returns) for the 100 day testing period.

4.4 Experimental Design

The general filtering technique chosen for this paper was Least Mean Squares (LMS) along with a few of its variants as detailed below. The standard formula for an LMS adaptive filter is:

Least Means Squares

$$w_t = w_{t-1} + \mu u^* \left[d_i - u_i w_{i-1} \right] \tag{4.6}$$

where w_i indicates the weight at time t, d represents the known or correct value at time t, u represents the predictive data, and μ is a positive step size. For each individual stock, d is the price (or Return) at time t and u represents a vector of prior prices of length l so that $u = \{u_{i-1}, u_{i-2}, ...u_{i-l}\}$. This leads to two hyper-parameters to be tuned: The step size μ and the length of history to consider at each step l.

Other algorithms explored were:

Normalized Least Means Squares

$$f = \mu / (\epsilon + |u|^2) \tag{4.7}$$

$$w_t = w_{t-1} + f \cdot u^* \left[d_t - u_t w_{t-1} \right] \tag{4.8}$$

Normalized Least Means Squares with Diffusion

$$f = \mu / (\epsilon + |u|^2) \tag{4.9}$$

$$\phi = \sum_{s=1}^{S} c_s \cdot w_s \tag{4.10}$$

$$w_t = \phi_{t-1} + f \cdot u^* \left[d_t - u_t \phi_{t-1} \right] \tag{4.11}$$

Sign Error Least Means Squares

$$w_t = w_{t-1} + \mu u^* csgn([d_t - u_t w_{t-1}]) \tag{4.12}$$

The Algorithm 1 from section 3 is treated as a general base framework for

testing different filtering strategies. For example the ui_t n the second stage filter may be the mean of the w_{st}, or the error weighted sum of w_{st}. The u_{st} may be phi, wi_t, or some other variable. This general structure allows us to quickly and easily test a wide variety of filtering strategies looking for optimal results. Additionally, the hyper-parameters μ and $size$ may be tuned for optimal results.

A total of 96 filters variations were developed and tested using multiple combinations of:

1. LMS, Normalized LMS, Diffusion LMS, or Sign Error LMS for the first stage filter.
2. LMS, Normalized LMS, or Sign Error LMS for the second stage filter.
3. Four different input variables for the second stage filter.
4. All of the above methods repeated, but with feedback from the second stage filter fed back to the first stage filter.

Finally, 10 values for μ and 10 values for $size$ were tested for each of the 96 filters. Resulting in evaluation of over 960 filters.

Using our initial two-stage algorithm, I found the optimal filter was a Diffusion Normalized LMS filter for stage one, with a Normalized filter for stage two, the phi from the diffusion used as the input data for the second stage filter, and the weights from the second stage filter fed back in to the first.

4.5 Results

Ultimately, I found that while our initial algorithm worked reasonably well, I discovered that results could be improved. The initial algorithm was modified to eliminate the feedback from the second stage to the first, thus eliminating the "alternating" step. Now, the output from the diffusion filter on the stocks was

used as the input to the index filter, as before, but the index filter was not fed back to the stock filters.

Algorithm 3 Final two stage filter algorithm

for t = 1 to Time **do**

for s = 1 to stocks **do**

$w_{st} \leftarrow \phi_t + \mu \cdot u^*_{st} \left[d_{st} - (\phi_t \cdot u_{st}) \right]$

end for

$\phi_t \leftarrow c \cdot w_{st}$

$wi_t \leftarrow wi_{t-1} + \mu \cdot ui^*_t \left[di_t - (\phi_t \cdot ui_t) \right]$

end for

Both algorithms were then tuned for ideal hyper-parameter values.

		RMSE	Z
Algo 1	Dollar	1.9786	.2276
	Return	0.00080	0.1826
Algo 2	Dollar	2.7911	0.2703
	Return	0.000795	0.5107

Table 4.2: Results when predicting prices

The best result, as defined by the lowest Z score is using Algorithm 2 for predicting prices with $\mu = 5$ and $size = 4$ as documented in Table 4.2 on Page 43. A chart of the final filtering process shows that it is tracking well. See Figure 4.2 on Page 44. The errors also appear to converge and are stable as evidenced by Figure 4.3 on Page 45

The second metric I discussed was the accuracy in predicting the direction of the index. The results are documented in Table 4.3 on Page 44. The best result was an accuracy of 76% found using algorithm 1 with $\mu = 0.1$ and $size = 2$. Being

able to predict the future direction of the stock index with 76% accuracy would appear to be a useful and potentially profitable result.

	Algo 1	Algo 2
Price	0.55	0.55
Return	0.76	0.70

Table 4.3: Results when predicting direction

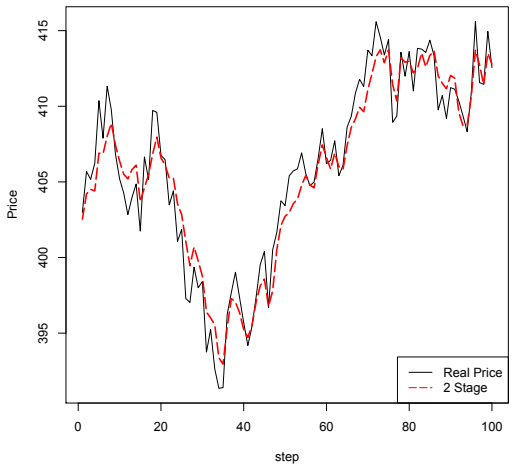

Figure 4.2: "Final algorithm predictions"

4.6 Discussion

It appears as if an industry "signal" can be learned from the stocks underlying an index that can help predict the future price or direction of the index better than just relying on the historical prices of the index alone. This could lead to a significant financial modeling system that doesn't appear to have been used before.

Given the limited scope of this paper, only variations of the LMS algorithm

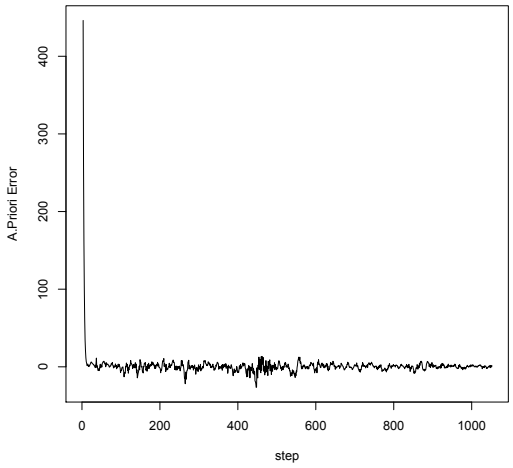

Figure 4.3: "Final Error Convergence"

were explored. However, the final filter derived from the second algorithm appears to be tracking well. It would be a useful test to simulate trading the actual index asset to determine if the best z-score or best direction score generated the highest profit, if any.

Furthermore, it would be worthwhile to develop and test adaptive filtering algorithms based on RLS or Lattice filters to see if improvements in stability, reduced error, or profit could be found. Additionally, it might be interesting to include input beyond the component stocks such as the price of oil, index rates, or even the temperature.

4.7 Conclusion

A two stage filtering algorithm was developed and coded to predict time forward prices or direction of The Dow Jones Utility index. We developed and tested 960 unique combinations of filters and hyper-parameters looking for the optimal combination as defined by two different error methods. The resulting algorithm produces a filter that tracks well and shows a good degree of accuracy in predicting

both future price and direction one day into the future.

CHAPTER 5

Estimating Adwords Clicks and Conversions using a Beta Distribution Regression

5.1 Introduction

In this paper, I propose a novel method for modeling both the click through rate and the conversion rate for ads in the AdWords system. Clicks are considered as arising from a Binomial distribution where the probability is a function of a Beta distribution. Conversions (sales) are also modeling with a similar structure. While prior papers have traditionally employed a logistic regression to directly model click through rate, I [use a re-paramaterized Beta distribution that allows for both a location and a precision parameter to be estimated separately, better capturing the variance of the inferred values. A full conditional likelihood model is described and fit for the probability of a click given an impression and a series of covariates and a hierarchical indicator of specific sub-brands. Separate models for both Click Through Rate and Conversion Per Click rate are then fit using a Metropolis within Gibbs algorithm to estimate the coefficients of interest.

5.2 Adwords Overview

Advertising is one of the largest generators of revenue stemming from The Internet. Most "free" services that exist online, are simply designed around a business model that realizes revenue through advertising. One could reasonably argue that

much of the World Wide Web is simply a vehicle for serving advertising, not unlike the typical free newspaper found in most major cities. Facebook, Myspace, Yahoo, Google, etc. are all examples of firms that generate most, if not all, of their revenue through advertising. Generally, online advertising can be divided into two distinct groups: Display and Search. In this paper, I will focus on Search advertising as that is currently the larger of the two. (Both in size and revenue.)

Google reported $43.68 billion dollars of advertising revenue for 2012. Of that revenue, 68%, or $31.22 Billion, came directly from Search Advertising. Google's main advertising product is known as AdWords and is composed of 11 ads present on any page listing the results of a search query. The pricing model is one of "pay-per-click" (PPC) where advertisers only pay if a browser clicks on their ad. Advertisers offer a bid of the maximum price they are willing to pay along with a daily budget of the maximum amount they are willing to spend. Generally, there are multiple advertisers bidding on a given search in Google. Google has an internal algorithm, that conducts an instant auction, designed to maximize their profit by selecting the order in which ads are displayed. (Ads in positions below 11 are placed on later pages of the search results, a very undesirable position.) A screenshot of a typical Google results page, illustrating AdWords is in Figure 5.1

One great advantage of Internet advertising vs. traditional media is that advertisers have no cost to modify ad creatives, and receive feedback on resulting performance very quickly. Advertisers bid to show their ad for a given search query, but they are unsure of the the optimal querys to bid on, the optimal creative, and the optimal maximum price to pay for a click. Furthermore, advertisers may intentionally have different ad variants or query targets in order to capture a broader demographic. The resulting scenario is that most advertisers have a large portfolio of many ads aimed at many querys, with only limited to no knowledge

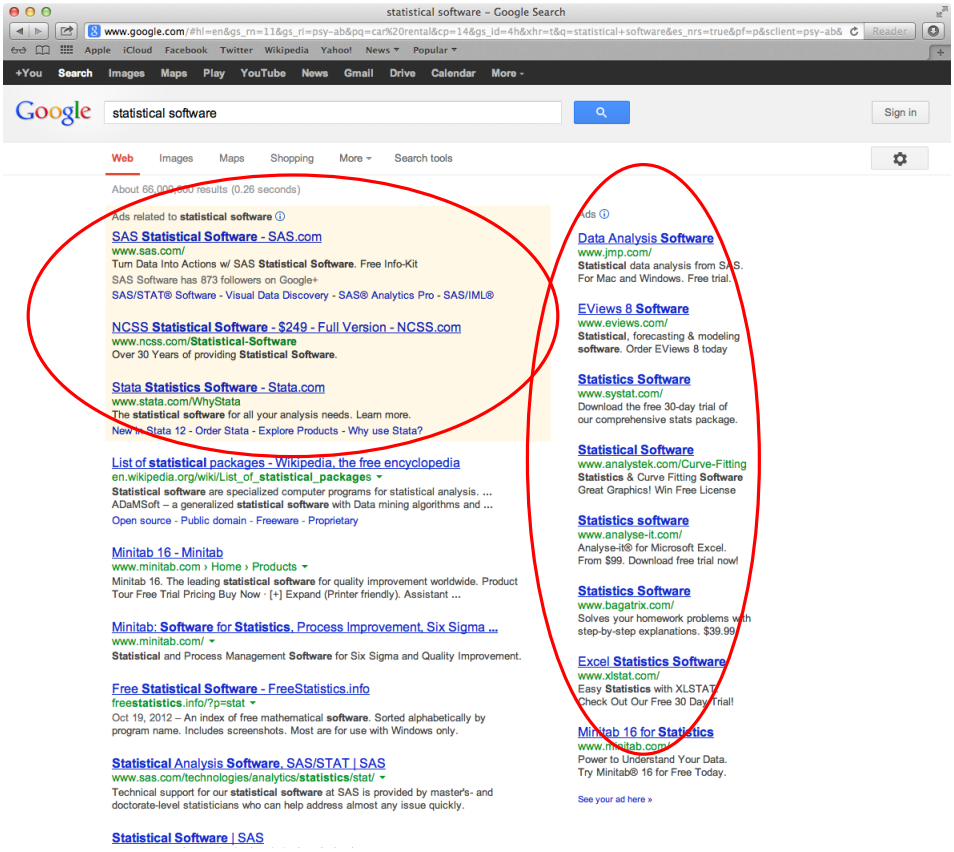

Figure 5.1: Adwords System

about how to optimize that complex scenario for maximum profit.

An advertiser faces three questions when developing an online advertising campaign using AdWords.

1. What query do I want to show ads for?

2. What ad should I run for a given query?

3. What is the optimal amount to bid for a click from a specific query?

In traditional advertising, massive effort has been spent on defining demographic segments, and then studying how to best target each specific segment -

where to advertise and what message to deliver. A majority of offline advertising is sold as belonging to a specific demographic group. Firms are often developing new ways to sub-divide audiences into better focused demographics. With pay-per-click advertising on search engines, there is no a priori definition of a demographic. Subsequently, we may use the query entered into the search engine as a proxy for identifying groups of similar interest or demand. People are giving us clues as to their interest by what they search on.

An advertiser may identify thousands of queries that may come from people interested in their product. However, determining the true value of people entering a specific query is not a clearcut process. People click on an ad, costing the advertiser money, and then hopefully complete a purchase, bringing the advertiser revenue. Understanding how cost and revenue are affected by different components of search queries is key to maximizing profit. In this paper, I focus on novel methods for estimating the true click through rate and conversion rate of ads as a function of various query features.

One key goal of any advertiser is to estimate both Clicks and Sales for a given search query. For if these two values were known with perfect information, then an optimal advertising strategy would be a trivial calculation. Each different query represents a distinct sub-group of people searching for information. Subsequently, we can use the query itself as an indication of the probability of a person clicking on an ad, and making a purchase. Furthermore, the Click Through Rate (CTR) and Conversion Per Click (CVC) will most likely respond differently to different query parameters. (i.e. some groups may click often, but purchase little while other groups may click rarely but purchase more often.) Figure 5.2 is a cartoon diagram of the system just described.

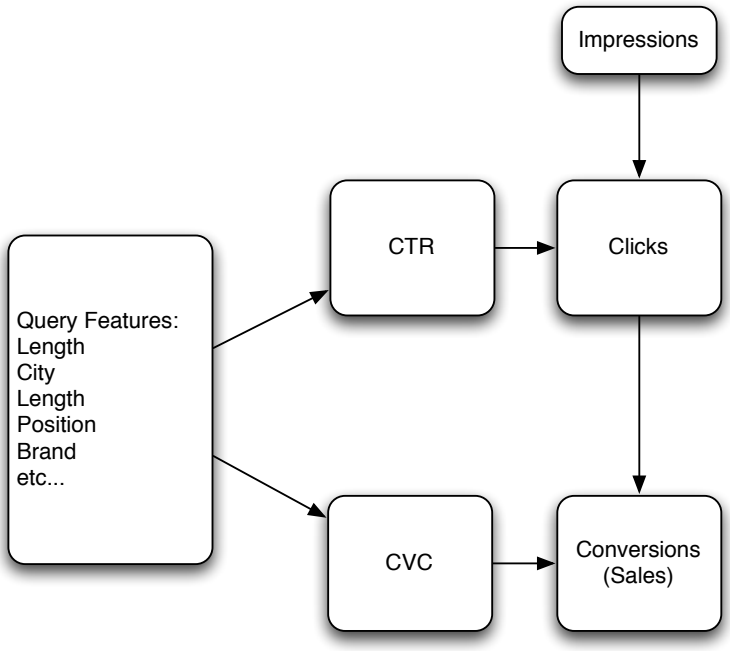

Figure 5.2: Adwords System

Due to the complex nature of the topic, and the diversity of published research, I will institute a standard notation for all mathematical formulae in this paper, described in Table 5.1.

5.3 Literature Review

5.3.1 Budget Optimization in Search-Based Advertising Auctions

[FMP06] Feldman et. al. , in one of the earlier papers on search ad optimization, approach the subject from the perspective an advertiser. The authors investigate optimal bid and budget choices given that the true CTR and POS are known for every bid level. They propose a randomized bidding strategy that consists of two bids at the endpoints of the line on a convex hull. The hull is defined on a graph of CTR and cost. However, they skip the important issues of estimating CTR as a function of either position or content and simply state that CTR can be measured

Q	query
I	Impression
C	Click
S	Sale (conversion)
R	Revenue from a conversion
CPC	Cost Per Click
CPV	Cost Per Conversion
POS	Position
BID	Bid amount
CTR	Click Through Rate: $Clicks/Impressions$
CVC	Conversion Per Click: $Conversions/Clicks$
CVI	Conversion per Impression: $Conversions/Impressions$

Table 5.1: Summary of Notation

empirically using past history. Another interesting contribution of the authors is the discussion of treating a budget composed of multiple queries as a "Knapsack Problem".

5.3.2 A Model of Individual Keyword Performance in Paid Search Advertising

[RB07] Rutz and Bucklin published the first paper that considered the profitability of an advertisement as a function of keywords (query input by user). The study focuses on discriminating between attractive vs. unattractive keyword without changing the existing Bid or ad text. Traditionally, ad campaign managers will use a pre-defined CTR as the threshold for keeping or discarding a keyword in their advertising campaign. (i.e. Only keep keywords that have a CTR of at least 0.05.) The authors point out that the flaws of this "traditional" method

of advertising in that it fails to account for the resultant CVC and CPV of each keyword. They then develop a model to evaluate the performance of individual keywords based on CVC. Conversion is described as a binary choice conditional on a click, represented by a logit model. A linear combination of weighted covariates plus an error term is considered the "value" of a particular keyword and is then used in a logistic function to determine the expected CVC of that keyword.

$$CVC_{k,t} = \frac{1}{1 + e^{-\nu}} \tag{5.1}$$

Two categories of covariates were used to fit this model. The first being meta-features of the keyword: POS, CTR, and CPC.

$$\nu_{k,t} = \beta_0 + \beta_1 POS_{k,t} + \beta_2 CTR_{k,t} + \beta_3 CPC_{k,t} + \epsilon_{k,t} \tag{5.2}$$

The second being actual words present in the keyword: City, State, Brand, etc.

$$\nu_{k,t} = \beta_0 + \beta_1 CITY_{k,t} + \beta_2 STATE_{k,t} + \beta_3 BRAND_{k,t} + \epsilon_{k,t} \tag{5.3}$$

Lastly, a Bayesian versions of the two models were developed where the β were unique to each keyword. (Note: In the Appendix, the authors suggest that the β are drawn from a prior Normal distribution.

$$\nu_{k,t} = \beta_{0,w} + \beta_{1,w} POS_{k,t} + \beta_{2,w} CTR_{k,t} + \beta_{3,w} CPC_{k,t} + \epsilon_{k,t} \tag{5.4}$$

$$\nu_{k,t} = \beta_{0,w} + \beta_{1,w} CITY_{k,t} + \beta_{2,w} STATE_{k,t} + \beta_{3,w} BRAND_{k,t} + \epsilon_{k,t} \tag{5.5}$$

Their results, from analysis of a large empirical dataset, produce several important findings:

- Keywords are predictive of conversion rates
- Position is not the only factor that drives conversion rate

- Covariates (CTR, Position, etc.) are important in estimating conversion rate
- Keywords with higher CTR tend to have higher CPC
- Wordgraphics (brand in query, etc.) have an effect in some of their models.

5.3.3 Predicting Clicks: Estimating the Click-Through Rate for New Ads"

[RDR07] Richardson, Dominowska, and Ragno write about the problem of estimating CTR for new ads from the perspective of a search engine. Since ad position is a function of both bid price and historical CTR, new ads are hard to position given their unknown CTR. No discussion of bid price or advertising budget is made. The authors look at several different feature sets as input to a logistic regression model that estimates CTR:

- The CTR of other ads that have the same bid term
- Related term CTR
- Unigrams from ad text
- keyword specificity
- External data sources. (web directories, search engine result count, frequency of query.)

The authors employ a logistic regression model and use the KL-divergence as the measure of goodness-of-fit. Their first feature is the CTR of "related" ads to estimate the CTR for a new ad.

$$f_{0,new} = \frac{\alpha\overline{CTR} + N(ad_{term})CTR(ad_{term})}{a + N(ad_{term})} \tag{5.6}$$

where $C\bar{T}R$ is the mean CTR for all ads, $N(term)$ is the count of ads with a given keyword, $CTR(ad_{term})$ is the CTR for ads with the same term and $alpha$ is a "prior" .

Another interesting feature they derive is the weighted average of the CTR of ads with related terms. For example, an ad for "red shoes" and an ad for "buy red shoes" are related with only one word difference. The authors describe a method of tuning across different lengths of similarity. Next, they looked at several features relating to subjective analysis of ad quality, such as appearance, reputation, landing page quality, etc. Finally, they looked at unigram features for the most common 10,000 words to see which had a positive or negative impact on CTR. For example, the word "official" in the ad title had the highest impact on CTR.

5.3.4 Budget constrained bidding in keyword auctions and online knapsack problems

[ZCL08] Chakrabarty, Zhou, and Lukose look at bidding on a portfolio of keywords as an online knapsack problem. Instead of focusing on keyword choice, or ad text, they look at how to allocate a budget with everything else known. They do not discuss how change in bid impacts both position and CTR.

In this paper, the authors assume that the correct CPC and CTR are known for all ads, and then describe an algorithm for allocating a budget amongst ads. Keeping with the terminology of the Knapsack Problem, the weight of an ad is defined as $w_t = CPC * C_t$ and the value of an ad is defined as $\pi = (R - CPC_t) * C$. The value to weight ratio, constrained by an lower bound L and an upper bound U is then $L < \frac{\pi_t}{w_t} < U$.

With the upper bound defined as $U = \frac{R}{CPC_{min}} - 1$, the lower bound set as some small constant $L > 0$, e as the base of the natural logarithm, and z as the percentage of budget used at time t, then their online knapsack algorithm is:

---

while Budget not exhausted **do**

$\psi_t = \frac{Ue^z}{L}\frac{L}{e}$

if $\frac{\pi_t}{w_t} > \psi_t$ **then**

Pick Element t

end if

end while

5.3.5 On Best-Response Bidding in GSP Auctions

[CDE08] The authors propose an equation they name "Balanced Bidding" that computes the upper bound an advertiser should be willing to pay for an improved position on online advertising. They then continue on to demonstrate that if all advertisers follow the Balanced Bidding procedure, that the system will converge to a Nash Equilibrium with probability 1.

The Balanced Bidding equation be derived as follows. Let:

- CTR_1 = click through rate for slot 1
- CTR_2 = click through rate for slot 2
- PPC_1 = amount paid per click in position 1
- PPC_2 = amount paid per click in position 2
- R = Revenue generated from a click

The equation is based on an equilibrium where the advertiser is indifferent as to the outcome of being at position 1 or position 2. (Note: Here, position one is preferred, being higher on the page and having a higher CTR, but costing more.)

$$CTR_1 \cdot R - CTR_1 \cdot CPC_1 = CTR_2 \cdot R - CTR_2 \cdot CPC_2 \tag{5.7}$$

$$R - CPC_1 = \frac{CTR_2 \cdot R - CTR_2 \cdot CPC_2}{CTR_1} \tag{5.8}$$

$$CPC_1 = \frac{CTR_2}{CTR_1} \cdot (R - CPC_2) \tag{5.9}$$

5.3.6 The value of location in keyword auctions[NDI10]

The authors make a case that the current auction structure for advertising, established by Google, is unfair to advertisers as it encourages them to pay more for an ad than it is actually worth to them. They describe modeling click through rate for an ad in position j as a probability θ_j with a Zipf distribution $\theta_j = \frac{1}{J^\alpha}$ where the probabilities sum to one. α is a parameter fit by estimation. Intuitively, this is a flawed representation, as the very strong possibility of no ad being clicked is completely ignored. Furthermore, given their formulation, the click through rate for position one is 100%.

The authors also explicitly use the balanced bidding strategy proposed in Carey et al.[CDE08], which is a very elegant solution to estimating a bid upper bound, but has some inherent flaws which will be discussed later in this paper.

Naldi, et al. then attempt to describe individual advertiser valuations of different advertising positions as a function of several different distributions. (Uniform, Normal, Exponential, and Pareto) The Zipf parameter is varied to produce a range of results, but no rationale is provided as to the choice of this parameter, or how to estimate the best fit of this distribution to the scenarios described.

5.3.7 Location, Location, Location: An Analysis of Profitability of Position in Online Advertising Markets

[AHS11] Agarwal et al. model both the CTR and CVI as functions of position, month, and keyword length. The inclusion of keyword length is a novel idea that the authors theorize is indicative of ad specificity and subsequent consumer interest. (i.e."Hotel Los Angeles Free Wifi" is more specific than "Hotel Los Angeles".) They explain that shorter keywords are more "general" and that while attracting clicks, may not generate as many sales. Longer keywords are more specific so that people clicking are more likely to find what they are looking for, and make a purchase. Their computations, from empirical data, show both CTR and CVI are non-monotonic; and that the most profitable ad position is most often not position one, but hovers around positions 3 to 4. It is explained that the CPC of ads in lower positions is significantly less than ads in higher positions, yet the CTR and CVI decrease at a lower rate. For example, an ad in position three might have a CPC half that of an ad in position 1, but will have a CTR only a little less, therefore making position three more profitable.

The authors also describe the problem of data sparsity, as many ads do not receive clicks or generate conversion each day. In order to more accurately fit their model with this sparsity, they adopt a hierarchical Bayesian approach where each ad's coefficients are drawn from a shared normal distribution with similar ads.

$$CVI_{k,t} = \frac{1}{exp^{-U_{k,t}}} \tag{5.10}$$

$$U_{k,t} = Y_{k,t} + \epsilon_{k,t} \tag{5.11}$$

$$Y_{k,t} = \beta_{k,t} X_{k,t} \tag{5.12}$$

$$\beta_k = \Delta_b Z_k + \mu_k \tag{5.13}$$

$$\mu_k \sim N(0, V) \tag{5.14}$$

With X_k representing covariates relating to position, Z_k representing the keyword specific characteristics - length in this case, Δ_b representing the relationship between keyword characteristics and the mean values of coefficients, and μ_k representing keyword specific heterogeneity. Consumer choice of selecting an ad can also be modeled using the same model, but substituting CTR for CVI.

5.3.8 Beta Regression for Modeling Rates and Proportions

[FC04] Ferrari and Cribari-Neto propose a re-parameterization of the Beta distribution that allows for relatively straightforward regression using a linear combination of weighted covariates through a logit link. The Beta probability density function takes two parameters $B(p, q)$. The authors re-parameterize the distribution with a mean $\mu = \frac{p}{p+q}$ and a precisions Φ such that $p = \mu\Phi$ and $q = (1-\mu)\Phi$. We can then define μ as a linear combination of covariates and coefficients passed through a logistic link function.

$$P(y; p, q) = \frac{\Gamma(p+q)}{\Gamma(p)\Gamma(q)} y^{p-1}(1-y)^{q-1}, 0 < y < 1 \tag{5.15}$$

$$P(y; \mu, \Phi) = \frac{\Gamma(\Phi)}{\Gamma(\mu\Phi)\Gamma((1-\mu)\Phi)} y^{\mu\Phi-1}(1-y)^{(1-\mu)\phi-1}, 0 < y < 1 \tag{5.16}$$

$$g(\mu) = \eta = \sum_{i=1}^{Q} x_i\beta_i \tag{5.17}$$

$$\mu = \frac{e^{-x^T\beta}}{1+e^{-xT\beta}} \tag{5.18}$$

5.4 A Brief Discussion of Ad Position

All of the papers discussed in the above literature review, along with all other papers on the subject read, describe ad position as a discrete integer. While this is absolutely true for a single ad display, the dynamic nature of the Adwords system means that for any repeated measures, ad position is stochastic, ceteris paribus

(bid, ad copy, etc.), Google reports AdWords statistics as a 24 hour summary with only *mean position* provided. In six months of data, for two very different companies, and a total of 56,441 distinct ads, not once was the daily average position reported as an integer value. Furthermore, there were 54,672 ads where the bid did not change during the study period, and an expectation of an integer daily position would have seemed plausible.

Trimming the data set to ads with variance below the 90th percentile, to minimize outliers, results in 49,204 ads of which 31,432 had a non-zero variance of reported position over the study period. The median reported position variance was 1.333. The distribution of reported ad position variance may be reasonably represented by a Gamma distribution with parameters shape=1.0238 and rate=0.6046. A density plot of the reported position variance, with representative Gamma distribution is in Figure 5.3.

One thought would be to use the Bid offered by the advertiser as a proxy for ad position or "ad competitiveness". However, this is difficult to do in actuality. An ad position may bounce around stochastically throughout the day due to numerous endogenous factors that we are unable to see. For example, both Google's internal algorithms and other advertisers changing bids will cause our ad position to change. However, we are never able to see that data. Given that we are only reported an average daily position, it isn't possible to derive the values of these other endogenous inputs.

Subsequently, for this paper, I will treat daily average ad position as a simple, continuous value, covariate to be used in the regressions, with full acknowledgement that it is an imperfect summary statistic. More complete modeling of the position can be considered an open problem.

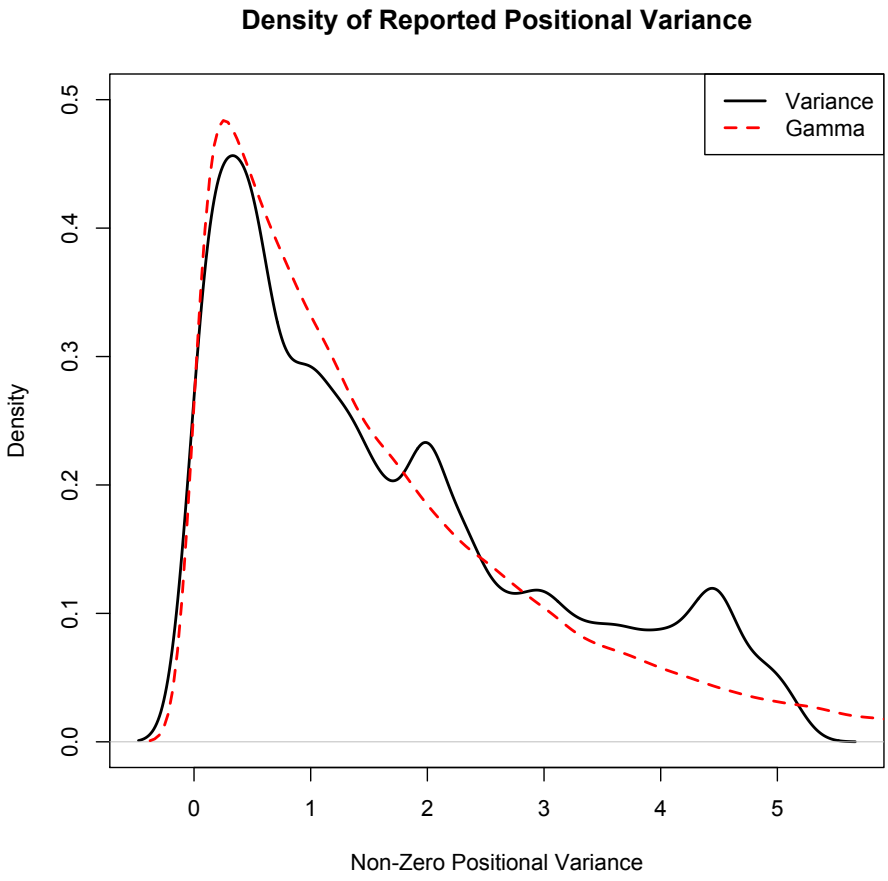

Figure 5.3: Variance of Reported Daily Average Ad Position

5.5 Understanding Variability of Ad Impressions

For a given ad, ceteris paribus, The number of reported daily impressions can vary significantly. The stochasticity of this key element can cause many naive advertisers to misunderstand the true performance of a given ad. Every day different people search for different things, different competitors may be changing ads and bids, and Google is constantly optimizing ad display to maximize *their* profit. An ad click is a fairly rare event, with most advertisers experiencing a CTR of 1% - 5%. However, the occasional event happens where an ad is only shown a few times for a day, and still receives a click. For example, an ad that receives an

average of 200 impressions per day may experience several days with only 3-4 impressions. Subsequently, clicks during those days will significantly bias the CTR and possible CVC rate much higher than actuality. Simply estimating CTR or CVC from a direct observation of impressions and clicks (or conversions) will lead to erroneous estimates. Furthermore, when an ad is relatively new, without many days of reporting, it is especially sensitive to this bias, and may lead an advertiser to incorrectly favor a new ad due to over estimation of its performance. Subsequently, any inference of CTR or CVC needs to take into account the likelihood of the number or impressions for each measurement.

The number of daily impressions, for the 56,441 ads in this study, with outliers above the 99th percentile removed, may be reasonably represented by a log-normal distribution with mean=2.1811 and standard deviation=1.1354. A density plot of reported daily impressions with representative log-normal distribution is in Figure 5.4

5.6 Modeling Clicks and Conversions as a Beta Distribution

Cary's Balanced Bid formula may be solved to give the equilibrium or "indifference" price for any ad position.[CDE08] In real world applications, where a price is not chosen by the advertiser, but a bid is made in an auction, the actual CPC paid is a function of a second-price or "Vickery" auction. The indifference price described by Cary, et al. may be considered an upper bound on the price an advertiser should rationally be willing to pay for a given position.

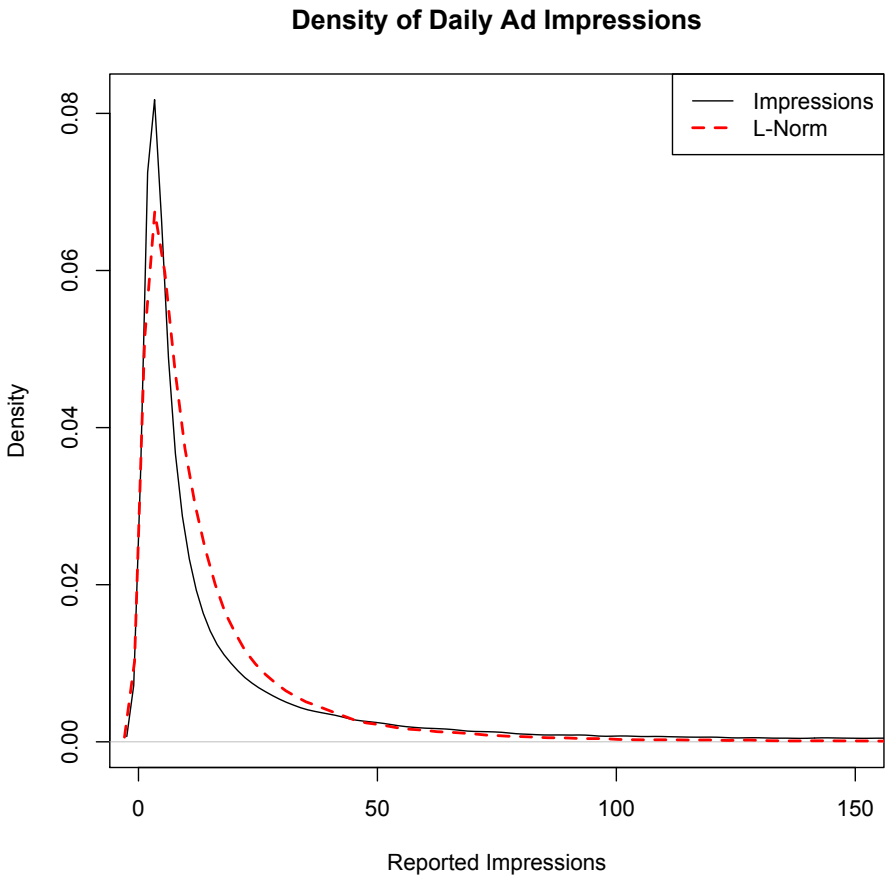

Figure 5.4: Variance of Reported Daily Average Ad Position

$$CTR_1 \cdot R - CTR_1 \cdot CPC_1 = CTR_2 \cdot R - CTR_2 \cdot CPC_2 \tag{5.19}$$

$$R - CPC_1 = \frac{CTR_2 \cdot R - CTR_2 \cdot P2}{CTR_1} \tag{5.20}$$

$$CPC_1 = \frac{CTR_2}{CTR_1} \cdot (R - CPC_2) \tag{5.21}$$

However, Cary neglects to consider that not all clicks convert to a revenue event, and that different ad positions may have different rates of CVC. This is a very important distinction, as a change in ad position that increases CTR might actually have a lower CVC, thereby reducing profit. Agarwal et al. have demon-

strated that position definitely has an impact on CVC [AHS11]. Including a position dependent CVC into Cary's equation then gives:

$$CPC_1 = CVC_1 \cdot R - \frac{CTR_2}{CTR_1}(CVC_2 \cdot R - CPC_2) \tag{5.22}$$

Once a query is chosen, and ad copy written, the only real "control" left to the advertiser is choice of bid. Cary's approximation is useful, if discrete position information is available, CTR for all positions is known, and CVC for all positions is known. However, we may still use it as a tool for thinking about advertising strategy. As previously discussed in this paper, both CTR and CVC are random variables $\in [0, 1]$, that it may be approximated by a Beta distribution, and position is a random variable with a daily average being reported.

If we are then able to model CTR and CVC using regression, then the fitted coefficients may then be used to determine the changes to cost, revenue, and expected profit as a function of changing the bid. Rutz and Bucklin [RB07] have shown that ad features and words have an effect on CTR, so a natural place to start, would then be to use regression to fit a Beta distribution using ad features and word grams to estimate both CTR and CVC.

Ferrari and Cribari-Neto [FC04] propose a re-parameterization of the Beta distribution that facilitates regression using a linear combination of weighted covariates transformed through a logit link.

$$P(y;p,q) = \frac{\Gamma(p+q)}{\Gamma(p)\Gamma(q)} y^{p-1}(1-y)^{q-1}, 0 < y < 1 \tag{5.23}$$

$$P(CTR;\mu,\phi) = \frac{\Gamma(\phi)}{\Gamma(\mu\phi)\Gamma((1-\mu)\phi)} y^{\mu\phi-1}(1-y)^{(1-\mu)\phi-1}, 0 < y < 1 \tag{5.24}$$

$$\mu = \frac{p}{p+q} \tag{5.25}$$

$$\phi = p+q \tag{5.26}$$

We can then represent μ as a transformed linear combination of covariates and coefficients. Given that $\mu \in [0,1]$ Ferrari and Cribari-Neto suggest using a logic link transformation of μ

$$\eta = X^t\beta \tag{5.27}$$

$$\mu = \frac{1}{1+e^{-\eta}} \tag{5.28}$$

One distinct advantage of use a Beta distribution over a Logistic is that the Beta allows for independent estimate of variance. ϕ is the precision parameter that may also be estimated through regression with different covariates and coefficients from μ. ϕ may then be considered a measure of difference between using a Beta distribution over a Logistic. As ϕ approaches 0, the variance of the Beta distribution approaches the variance of the Logistic, and the means of the distributions will also converge. Given the re-parameterization of the PDF described above, the variance then becomes:

$$VAR = \frac{\mu(1-\mu)}{1+\phi} \tag{5.29}$$

$$\phi = X^t\gamma \tag{5.30}$$

While Rutz and Bucklin [RB07] proposed a logistic regression model, we can compare the difference between my proposed Beta distribution with their logistic through the parameter ϕ.

5.7 Full Model for Estimating CTR and CVC

We can consider the number of clicks an ad receives as represented by a binomial distribution, $P(C) \sim Binom(I, \pi)$ with I representing the number of impressions an ad received for the day, and π representing the probability of a click (CTR). Treating π as a random variable arising from a Beta distribution allows for regression using query specific covariates and a separate parameter ϕ for the variance of π

Let θ represent the coefficients of the model and X be an $[N, M]$ vector of covariates representing query length, brand, program, and location for each ad on each day.

The full conditional model for a sale, given a click and ad features, is:

$$P(S|I, C, X) = P(C|I, CTR) \cdot P(CTR|X, \theta_{ctr}) \cdot P(\theta_{ctr}) \tag{5.31}$$

$$\cdot P(S|C, CVC) \cdot P(CVC|X, \theta_{cvc}) \cdot P(\theta_{cvc}) \tag{5.32}$$

$$\tag{5.33}$$

As CTR and CVC are independent, each may be modeled separately, using the same covariates and model. Company A owns seven distinct brands. While a simple indicator covariate for "brand in query" could be used, I chose to treat coefficients for each distinct brand as coming from a nomad distribution with mean and variance as coefficients to be estimated. That hierarchy provides an additional level of understanding as to how each brand affects both CTR and CVC.

For Clicks:

$$P(C) \sim Binom(I, \pi^{cl}) \tag{5.34}$$

$$P(\pi^{cl}|X, \theta^{cl}) \sim Beta(\mu^{cl}\phi^{cl}, (1-\mu^{cl})\phi^{cl}) \tag{5.35}$$

$$\mu^{cl} = \frac{1}{1+e^{-X^t\beta^{cl}+X^t_{brands}\cdot\beta^{cl}_{brands}}} \tag{5.36}$$

$$P(\beta^{cl}) \sim \mathcal{MVN}(0, \Sigma^{cl}_{beta}) \tag{5.37}$$

$$P(\beta^{cl}_{brands}) \sim N(\mu^{cl}_{brands}, \tau^{cl}_{brands}) \tag{5.38}$$

$$P(\mu^{cl}_{brands}) \sim N(0, \gamma^{cl}) \tag{5.39}$$

$$P(\tau^{cl}_{brands}) \sim EXP(\delta^{cl}) \tag{5.40}$$

$$P(\phi^{cl}) \sim Exp(\lambda^{cl}) \tag{5.41}$$

$$\tag{5.42}$$

With γ^{cl}, δ^{cl}, λ^{cl}, Σ^{cl}_{beta}, and α^{cl} as hyper-parameters for this hierarchical model.

Sales, given a click:

$$P(S) \sim Binom(C, \pi^{cv}) \tag{5.43}$$

$$P(\pi^{cv}|X, \theta^{cv}) \sim Beta(\mu^{cv}\phi^{cv}, (1-\mu^{cv})\phi^{cv}) \tag{5.44}$$

$$\mu^{cv} = \frac{1}{1+e^{-X^t\beta^{cv}+X^t_{brands}\cdot\beta^{cv}_{brands}}} \tag{5.45}$$

$$P(\beta^{cv}) \sim \mathcal{MVN}(0, \Sigma^{cv}_{beta}) \tag{5.46}$$

$$P(\beta^{cv}_{brands}) \sim N(\mu^{cv}_{brands}, \tau^{cv}_{brands}) \tag{5.47}$$

$$P(\mu^{cv}_{brands}) \sim N(0, \gamma^{cv}) \tag{5.48}$$

$$P(\tau^{cv}_{brands}) \sim EXP(\delta^{cv}) \tag{5.49}$$

$$P(\phi^{cv}) \sim Exp(\lambda^{cv}) \tag{5.50}$$

$$\tag{5.51}$$

With γ^{cv}, δ^{cv}, λ^{cv}, Σ^{cv}_{beta}, and α^{cv} as hyper-parameters for this hierarchical model.

5.8 Estimating Model Coefficients

The complex nature of the proposed model presents a rather difficult challenge for fitting coefficients. A Metropolis within Gibbs algorithm was written in the R statistical programming language and used to fit all coefficients. Separate processes were then run for both CTR and CVC. The algorithm is descried in Algorith 4

Algorithm 4 Metropolis within Gibbs

for 1:S **do**

Sample $P(\tau_{brand}|\mu_{brand}, \beta, \phi, X, Clicks, Impr)$ with small MH loop.

for j in 1:T **do**

    $\tau^*_{brand} \leftarrow \tau_{brand} + N(0,1)$ # Proposal

    Compute L^* using τ^*_{brand}

    $r = min(1, \frac{L_{prop}}{L})$ # Acceptance ratio

    if $r > U(0,1)$ **then**

        $\tau^i_{brand} \leftarrow \tau^*_{brand}$

        $L \leftarrow L^*$

    else

        $\tau^i_{brand} \leftarrow \tau^i_{brand}$

    end if

end for

Sample $P(\mu_{brand}|\tau_{brand}, \beta, \phi, X, Clicks, Impr)$ with small MH loop.

for j in 1:T **do**

    $\mu^*_{brand} \leftarrow \mu_{brand} + N(0,1)$ # Proposal

    Compute L^* using μ^*_{brand}

    $r = min(1, \frac{L_{prop}}{L})$ # Acceptance ratio

    if $r > U(0,1)$ **then**

        $\mu^i_{brand} \leftarrow \mu^*_{brand}$

        $L \leftarrow L^*$

    else

        $\mu^i_{brand} \leftarrow \mu^i_{brand}$

    end if

end for

Algorithm 5 Metropolis within Gibbs - Part 2

Loop over all brands

for b in Brands **do**

    # Sample $P(\beta_{brand_b}|\tau_{brand}, \mu_{brand}, \beta, \phi, X, Clicks, Impr)$ with small MH loop.

    for j in 1:T **do**

        $\beta^*_{brand_b} \leftarrow \beta^i_{brand_b} + N(0,1)$ # Proposal

        Compute L^* using $\beta^*_{brand_b}$

        $r = min(1, \frac{L_{prop}}{L})$ # Acceptance ratio

        if $r > U(0,1)$ **then**

            $\beta^i_{brand_b} \leftarrow \beta^*_{brand_b}$

            $L \leftarrow L^*$

        else

            $\beta^i_{brand_b} \leftarrow \beta^i_{brand_b}$

        end if

    end for

end for

Sample $P(\beta|\tau_{brand}, \mu_{brand}, \phi, X, Clicks, Impr)$ with small MH loop.

for j in 1:T **do**

    $\beta^* \leftarrow \beta^i + MVN(0, \Sigma)$ # Proposal

    Compute L^* using β^*

    $r = min(1, \frac{L_{prop}}{L})$ # Acceptance ratio

    if $r > U(0,1)$ **then**

        $\beta^i \leftarrow \beta^*$

        $L \leftarrow L^*$

    else

        $\beta^i \leftarrow \beta^i$

    end if

end for

Algorithm 6 Metropolis within Gibbs - Part 3

53: # Sample $P(\phi|bid, pos, clicks, impr, \beta_{position}, \beta)$ with small MH loop.

54: **for** j in 1:T **do**

55: $\phi^* \leftarrow \phi^i + N(0,1)$ # Proposal

56: Compute L^* using β^*

57: $r = min(1, \frac{L_{prop}}{L})$ # Acceptance ratio

58: **if** $r > U(0,1)$ **then**

59: $\phi^i \leftarrow \phi^*$

60: $L \leftarrow L^*$

61: **else**

62: $\phi^i \leftarrow \phi^i$

63: **end if**

64: **end for**

5.9 Data Collection

Google provides a convenient, but complicated API for directly accessing data from an AdWords account. Code was written in Python to interface directly with Google's AdWords API and query all available data. The collected data was then stored in a MySQL database.

164 days of full advertising data (impressions, clicks, conversions, position, bids, cost-per-click, queries, and grouping.) were collected for two distinct companies. Company A owns seven brands consisting of many trade schools around the country that specialize in technical training for fields such as nursing, welding, auto repair, etc. Company B is a dating site for a narrowly focused demographic. To protect company privacy, the names are withheld and actual ad text has been anonymized.

The dataset for Company A was used for empirical fit and analysis of the proposed model. There were a total of 19,257 ads of which 6,863 had more than two changes in bid price over the study period. 3,590 ads had more than two bid changes and at least one click. Of those ads, 742 produced at least one conversion.

	Company A
Ads	19,257
Ad Groups	9,459
Impressions	21,470,029
Clicks	199,649
Conversions	4,286

Table 5.2: Data Collected

5.10 Empirical Analysis and Results

To be determined.

References

[AHS11] Ashish Agarwal, Kartik Hosanagar, and Michael D. Smith. "Location, Location, Location: An Analysis of Profitability of Position in Online Advertising Markets." *Journal of Marketing Research (JMR)*, **48**(6):1057–1073, 2011.

[Bay07] Andrew Bayer. *Bayer on Speed; New Strategies for Racetrack Betting*. Mariner Books, 2007.

[BC86] Ruth N. Bolton and Randall G. Chapman. "Searching for Positive Returns at the Track: A Multinomial Logit Model for Handicapping Horse Races." *Management Science*, **32**(8):1040–1060, 1986.

[Ben94] Bill Benter. *Computer based horse race handicapping and wagering systems: a report*, pp. 183–198. London: Academic Press, 1994.

[Boa11] OpenMP Architecture Review Board. "OpenMP Application Program Interface Version 3.1.", May 2011.

[CDE08] Matthew Cary, Aparna Das, Benjamin Edelman, Ioannis Giotis, Kurtis Heimerl, Anna R. Karlin, Claire Mathieu, and Michael Schwarz. "On Best-Response Bidding in GSP Auctions." *National Bureau of Economic Research Working Paper Series*, pp. 13788+, February 2008.

[Cha94] R.G. Chapman. *Still Searching for Positive Returns at the Track: Empirical Results from 2,000 Hong Kong Races*, pp. 173–81. London: Academic Press, 1994.

[ED07] Edelman and David. "Adapting support vector machine methods for horserace odds prediction." *Annals of Operations Research*, **151**(1):325–336, April 2007.

[FC04] Silvia Ferrari and Francisco Cribari-Neto. "Beta Regression for Modelling Rates and Proportions." *Journal of Applied Statistics*, **31**(7):799–815, August 2004.

[FMP06] Jon Feldman, S. Muthukrishnan, Martin Pal, and Cliff Stein. "Budget Optimization in Search-Based Advertising Auctions." Technical report, December 2006.

[Gil01] Muriel Gillick. "Guest Editorial: Pinning Down Frailty." *The Journals of Gerontology Series A: Biological Sciences and Medical Sciences*, **56**(3):M134–M135, 2001.

[Hau08] Donald B. Hausch, editor. *Efficiency of Racetrack Betting Markets*. World Scientific Publishing Company, 2008.

[HB10] J. Hoberock and N. Bell. "Thrust: A parallel template library." *Online at http://thrust. googlecode. com*, 2010.

[Hof09] Peter Hoff. *A First Course in Bayesian Statistical Models*. Springer Science and Business Media, 2009.

[Kap03] Michael Kaplan. "The High Tech Trifecta.", 2003.

[Kel56] J. Kelly. "A new interpretation of information rate." *Information Theory, IRE Transactions on*, **2**(3):185–189, September 1956.

[LS08] C.G. Lopes and A.H. Sayed. "Diffusion Least-Mean Squares Over Adaptive Networks: Formulation and Performance Analysis." *Signal Processing, IEEE Transactions on*, **56**(7):3122–3136, 2008.

[LSJ07] Stefan Lessmann, M. Sung, and Johnnie E.V. Johnson. "Adapting least-square support vector regression models to forecast the outcome of horseraces." *Journal of Prediction Markets*, **1**(3):169–187, December 2007.

[LSJ09] Stefan Lessmann, Ming-Chien Sung, and Johnnie E. V. Johnson. "Identifying winners of competitive events: A SVM-based classification model for horserace prediction." *European Journal of Operational Research*, **196**(2):569–577, 2009.

[LSJ10] Stefan Lessmann, Ming-Chien Sung, and Johnnie E.V. Johnson. "Alternative methods of predicting competitive events: An application in horserace betting markets." *International Journal of Forecasting*, **26**(3):518 – 536, 2010. ¡ce:title¿Sports Forecasting¡/ce:title¿.

[MN89] Peter McCullagh and J. A. Nelder. *Generalized Linear Models, Second Edition (Chapman & Hall/CRC Monographs on Statistics & Applied Probability)*. Chapman & Hall/CRC, London, United Kingdom, 2 edition, August 1989.

[NDI10] Maurizio Naldi, Giuseppe D'Acquisto, and Giuseppe F. Italiano. "The value of location in keyword auctions." *Electronic Commerce Research and Applications*, **9**(2):160–170, March 2010.

[RB07] Oliver J. Rutz and Randolph E. Bucklin. "A Model of Individual Keyword Performance in Paid Search Advertising.", 2007.

[RDR07] Matthew Richardson, Ewa Dominowska, and Robert Ragno. "Predicting clicks: estimating the click-through rate for new ads." In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pp. 521–530, New York, NY, USA, 2007. ACM.

[SSZ12] M.A. Suchard, S.E. Simpson, I. Zorych, P. Ryan, and D. Madigan. "Massive parallelization of serial inference algorithms for a complex generalized linear model." *arXiv preprint arXiv:1208.0945*, 2012.

[Tib96] Robert Tibshirani. "Regression Shrinkage and Selection via the Lasso." *Journal of the Royal Statistical Society. Series B (Methodological)*, **58**(1):267–288, 1996.

[Tib97] Robert Tibshirani. "The LASSO Method for Variable Selection In The Cox Model." *Statist. Med.*, **16**(4):385–395, 1997.

[TYL77] Yoshio Takane, ForrestW Young, and Jan Leeuw. "Nonmetric individual differences multidimensional scaling: An alternating least squares method with optimal scaling features." **42**(1):7–67, 1977.

[Veg88] Joseph de la Vega. "Confusión de Confusiones." 1688.

[wik] "Technical Analysis.".

[ZC10] Xiaolian Zheng and B.M. Chen. "Identification of market forces in the financial system adaptation framework." In *Control and Automation (ICCA), 2010 8th IEEE International Conference on*, pp. 103–108, 2010.

[ZCL08] Yunhong Zhou, Deeparnab Chakrabarty, and Rajan Lukose. "Budget constrained bidding in keyword auctions and online knapsack problems." In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pp. 1243–1244, New York, NY, USA, 2008. ACM.

[ZLS10] H. Zhou, K. Lange, and M.A. Suchard. "Graphics processing units and high-dimensional optimization." *Statistical science: a review journal of the Institute of Mathematical Statistics*, **25**(3):311, 2010.