Proceedings Article

A Combinatorial Prediction Market for the U.S. Elections

MIROSLAV DUDÍK, Microsoft Research, New York City
SEBASTIEN LAHAIE, Microsoft Research, New York City
DAVID M. PENNOCK, Microsoft Research, New York City
DAVID ROTHSCHILD, Microsoft Research, New York City

We report on a large-scale case study of a combinatorial prediction market. We implemented a back-end pricing engine based on Dudík et al.'s [2012] combinatorial market maker, together with a wizard-like front end to guide users to constructing any of millions of predictions about the presidential, senatorial, and gubernatorial elections in the United States in 2012. Users could create complex combinations of predictions and, as a result, we obtained detailed information about the joint distribution and conditional estimates of election results. We describe our market, how users behaved, and how well our predictions compared with benchmark forecasts. We conduct a series of counterfactual simulations to investigate how our market might be improved in the future.

Categories and Subject Descriptors: J.4 [**Social and Behavioral Sciences**]: Economics

General Terms: Economics

Additional Key Words and Phrases: prediction market, market maker, combinatorial security, elections

1. INTRODUCTION

Political prediction markets like Intrade and Betfair offer financial derivatives linked to election outcomes. Historically, prediction markets and other forms of expectation polling have a record of outperforming traditional voter-intention polls when it comes to projecting election winners [Rothschild 2009]. In 2012, these markets offered an impressive array of predictions about the U.S. elections, including the national presidential winner, the number of electoral votes for each candidate, and state-by-state outcomes in presidential, senatorial, and gubernatorial races.

From September 16, 2012 until election day on November 6, 2012, we ran our own prediction market game called WiseQ that went well beyond predicting individual elections. Our market allowed traders to predict *combinations* of outcomes, for example,

— The Republican party candidate Mitt Romney would win Ohio yet lose the election,
— The same party would win both Ohio and Pennsylvania, or
— Less than 100,000 jobs would be created in October 2012 and Romney would win.

In WiseQ, as in other recent prediction market designs [Abernethy et al. 2011; Hanson 2007], traders interact with a central market maker that sets prices on any and all possible predictions. Our market was expressive enough to allow traders to construct millions of such predictions on the fly, obtain a nearly-instantaneous price quote reflecting the current odds, purchase the contract, and watch as the odds for that outcome and thousands of related outcomes changed in response. The market was a fully working implementation of Dudík et al.'s [2012] automated combinatorial market maker with linear constraints that

Author contact: {mdudik,slahaie,dpennock,davidmr}@microsoft.com

EC'13, June 16–20, 2013, Philadelphia, USA.

operationalized every aspect of an interactive market website, including direct login, third-party login, portfolio display, map interface, league creation, leaderboard, detailed FAQs, forum, popular trades, ability to browse other people's trades, one-click selling, and more. The game was featured on Yahoo! News, the Huffington Post, RealClearPolitics, and the New York Times, drawing hundreds of traders from around the world.

In theory, combinatorial prediction markets can provide orders of magnitude more information than ordinary prediction markets. In practice, many questions remain. With so many choices, will traders become overwhelmed? What is a good interface to guide traders through the endless sea of possible predictions? Can a combinatorial exchange that simply matches willing traders ever work, or is a central market maker necessary to provide liquidity for the long tail of predictions that users will inevitably generate? Can the empirical accuracy we see in ordinary prediction markets generalize to combinatorial markets with the same number of "mental cycles" spread across a vastly larger space of outcomes? Can the market maker compute prices fast enough to support and satisfy many simultaneous traders? Do linear constraints induce a significant and noticeable amount of information propagation? Does offering users extreme flexibility to compose their own complex predictions engage or discourage them?

In this paper, we begin to address some of these questions. We report for the first time on a large-scale case study of a combinatorial prediction market. We describe our user interface, how people behaved, what types of predictions people made, the extent to which traders took advantage of the combinatorial expressiveness, how our market incorporated information, and how accurately our market predicted outcomes. We consider accuracy on outcomes that users directly traded as well as outcomes that were never explicitly touched, on which our market was still able to form inferences.

For the probabilities of simple (non-combinatorial) outcomes, we compare the accuracy of our market to PredictWise, a prediction aggregator that combines data from Intrade, Betfair, and other sources. We also demonstrate the power of our market to elicit predictions that go well beyond simple outcomes. We generate histograms of the entire distribution of Electoral College votes for each candidate and the distribution of jobs created in September, showing how they compare to actual outcomes. We are able to compute conditional probabilities, for example the likelihood of Romney winning given that the October jobs report is bleak, or the likelihood that whoever wins Ohio wins the election.

Finally, we conduct a series of counterfactual simulations to determine what design decisions we could have improved upon. We examine how decreases or increases in the liquidity of our market maker would have hurt or improved our forecasts.

On the whole, our market performed well, both on the front end—hundreds of users placed thousands of trades including many intricate combinatorial trades—and the back end—quoting prices in real time and quickly propagating information via linear constraints—yielding accurate and sensible results.

2. MARKET DESIGN

2.1. Preliminaries

We begin by describing the market maker used in our election market. Dudík et al. [2012] and Abernethy et al. [2011] provide more details, background, and generalizations of this market design.

Outcomes and securities. Let Ω be a finite set of *outcomes*, corresponding to mutually exclusive and exhaustive states of the world. We are interested in eliciting expectations of binary random variables $\phi_i : \Omega \to \{0, 1\}$, $i \in \mathcal{I}$, which model the occurrence of various events taking indices in the finite set $\mathcal{I}$. For example, we could consider a variable that equals 1 if Romney wins Iowa in the Presidential Elections 2012, and 0 otherwise. Another example is a variable that equals 1 if the September 2012 jobs numbers (the September change in

nonfarm payroll) is more than 100,000 and Obama wins the Presidential Elections 2012, and 0 otherwise.

Each variable ϕ_i is associated with a *security*. A security is a contract that pays out $\phi_i(\omega)$ dollars when the outcome ω occurs; thus the random variable ϕ_i is also called the *payoff function*. Binary securities, such as the ones we described above, pay out \$1 if the specified event occurs and \$0 otherwise. The vector $(\phi_i)_{i\in\mathcal{I}}$ is denoted $\boldsymbol{\phi}$.

Cost function. The state of the market is specified by a vector $\boldsymbol{\theta} \in \mathbb{R}^{\mathcal{I}}$ listing the number of shares of each security sold so far by the market maker. The market starts at $\boldsymbol{\theta} = \boldsymbol{\theta}_0$. Security prices are defined using a convex and differentiable *cost function* $C : \mathbb{R}^{\mathcal{I}} \to \mathbb{R}$, following the cost function paradigm of market making [Chen and Pennock 2007]. Traders buy *bundles* $\boldsymbol{\delta} \in \mathbb{R}^{\mathcal{I}}$ of security shares issued by a central market maker. A trader holding a bundle $\boldsymbol{\delta}$ receives the payoff $\boldsymbol{\delta} \cdot \boldsymbol{\phi}(\omega)$; negative entries in $\boldsymbol{\delta}$ correspond to short-selling.

A trader wishing to buy a bundle $\boldsymbol{\delta}$ in the market state $\boldsymbol{\theta}$ must pay $C(\boldsymbol{\theta}+\boldsymbol{\delta})-C(\boldsymbol{\theta})$ to the market maker, after which the new state becomes $\boldsymbol{\theta}+\boldsymbol{\delta}$. Thus, the vector of instantaneous prices in the state $\boldsymbol{\theta}$ is $p(\boldsymbol{\theta}) \coloneqq \nabla C(\boldsymbol{\theta})$. A risk-neutral trader is incentivized to buy the security i if she believes that the expectation of ϕ_i is larger than $p_i(\boldsymbol{\theta})$, and sell if she believes it is lower.

Market desiderata. Let conv denote the convex hull, cl the closure of a set, and im an image of a mapping. We say that $\boldsymbol{\mu} \in \mathbb{R}^{\mathcal{I}}$ is a *coherent* belief if there exists a probability distribution over Ω such that $\boldsymbol{\mu} = \mathbb{E}[\boldsymbol{\phi}]$. The set of all coherent beliefs is denoted $\mathcal{M}$ and is called a marginal polytope (or a realizable polytope). It is not too difficult to show that $\mathcal{M} = \operatorname{conv}\{\boldsymbol{\phi}(\omega) : \omega \in \Omega\}$.

There are several standard properties that we may wish our market to satisfy [Abernethy et al. 2011]:

— *Expressiveness.* For any coherent belief, there exists a state $\boldsymbol{\theta}$ that achieves the corresponding prices at least in a limit: $\mathcal{M} \subseteq \operatorname{cl}\operatorname{im}\boldsymbol{p}$.
— *No arbitrage.* There is no market state $\boldsymbol{\theta}$ that would allow trades with a guaranteed profit regardless of the outcome: for all $\boldsymbol{\theta}$, $\boldsymbol{\delta} \in \mathbb{R}^{\mathcal{I}}$, there is $\omega \in \Omega$ such that $C(\boldsymbol{\theta}+\boldsymbol{\delta}) - C(\boldsymbol{\theta}) \geq \boldsymbol{\delta} \cdot \boldsymbol{\phi}(\omega)$. Abernethy et al. [2011] show that there is no arbitrage opportunity if and only if $\operatorname{im}\boldsymbol{p} \subseteq \mathcal{M}$.
— *Bounded loss.* There exists a constant M which bounds the loss of the market maker regardless of the outcome and how many shares of each security are sold.

Linearly constrained market making. In many cases of interest, including the combinatorial market in this paper, market making that is simultaneously expressive and arbitrage-free is computationally hard [Chen et al. 2008b]. To facilitate efficient pricing, we use Dudík et al.'s [2012] market maker design, which is expressive and has a bounded loss, but allows arbitrage. Arbitrage is partly resolved by maintaining a set of linear constraints that must be satisfied by coherent beliefs.

The construction of a *linearly constrained market maker* (LCMM) begins with a tuple $(\Omega, \mathcal{I}, \boldsymbol{\phi}, C)$ describing an expressive bounded-loss market whose prices can be efficiently calculated. Because of expressivity, we have $\mathcal{M} \subseteq \operatorname{cl}\operatorname{im}\boldsymbol{p}$. For all but the simplest problems, we will also have $\mathcal{M} \subsetneq \operatorname{cl}\operatorname{im}\boldsymbol{p}$, that is, there will be some arbitrage opportunities. An LCMM defines a relaxation $\tilde{\mathcal{M}} \supseteq \mathcal{M}$ described by linear constraints:

$$\tilde{\mathcal{M}} = \{\boldsymbol{\mu} \in \mathbb{R}^{\mathcal{I}} : \mathbf{A}^\top \boldsymbol{\mu} \geq \mathbf{b}\} \ .$$

When a constraint is violated, there is an arbitrage opportunity in the market, and LCMM acts as an arbitrager by executing appropriate trades until none of the constraints are violated. Dudík et al. [2012] provide a procedure that allows independent pricing and arbitrage removal, while maintaining bounded loss. Some of the constraints are explicitly listed and some are generated during the run of the market.

2.2. Compositional market design

While the abstractions used in the previous section are useful for reasoning about pricing, arbitrage and loss, in order to implement a combinatorial market we need to develop a more structured view that allows easy definitions of the relationships that must hold among securities.

Securities in our market correspond to indicators of discrete random variables. The variables are denoted $X_j : \Omega \to \mathcal{X}_j$ and their values are denoted x_j. Given a set of variables X_j, $j \in \mathcal{J}$, we construct a set of securities $\mathcal{I} = \{(j, x_j) : j \in \mathcal{J},\ x_j \in \mathcal{X}_j\}$ with payoff functions corresponding to indicators

$$\phi_j(x_j; \omega) = 1[X_j(\omega) = x_j] \ ,$$

where we use notation $\phi_j(x_j; \omega)$ instead of a more awkward $\phi_{(j,x_j)}(\omega)$. Our cost function is the sum of Hanson's [2003; 2007] logarithmic market scoring rules (LMSR) across the variables:

$$C(\boldsymbol{\theta}) = b \sum_{j \in \mathcal{J}} \ln \left(\sum_{x_j \in \mathcal{X}_j} e^{\theta_j(x_j)/b} \right) \ ,$$

where $\theta_j(x_j)$ denotes $\theta_{(j,x_j)}$ and $b > 0$ is the *liquidity parameter* controlling how fast the prices change in response to the purchase of shares. A smaller value of b (lower liquidity) means prices rise faster as shares are purchased; a larger value of b (higher liquidity) yields slower changes.

We allow traders to buy bundles of the form $1[X_j \in A]$ where $A \subseteq \mathcal{X}_j$. By this we mean vectors $\boldsymbol{\delta} \in \mathbb{R}^{\mathcal{I}}$ such that

$$\delta_{j'}(x_{j'}) = \begin{cases} \delta & \text{if } j' = j \text{ and } x_{j'} \in A \\ 0 & \text{otherwise} \end{cases}$$

where $\delta > 0$ is the (scalar) number of shares bought. Note that $\delta > 0$ is not a restriction since users can buy the complementary bundle $1[X_j \notin A]$ to achieve the same effect as short-selling.

The cost of δ shares of bundle $1[X_j \in A]$ is

$$C(\boldsymbol{\theta} + \boldsymbol{\delta}) - C(\boldsymbol{\theta}) = b \ln \left(\mu[X_j \in A] e^{\delta/b} + \mu[X_j \notin A] \right)$$

where we use the shorthand

$$\mu[X_j \in A] = \sum_{x_j \in A} p_j(x_j; \boldsymbol{\theta}) = \frac{\sum_{x_j \in A} e^{\theta_j(x_j)/b}}{\sum_{x_j \in \mathcal{X}_j} e^{\theta_j(x_j)/b}} \ .$$

(Recall that by definition $p(\boldsymbol{\theta}) = \nabla C(\boldsymbol{\theta})$.)

Initially, our market contains no variables and hence no securities. The market operator can create new variables and specify their relationship to any existing variables. At the time of creation of a new variable X_j, we need to specify its domain $\mathcal{X}_j$, the mapping $X_j(\omega)$, initial prices $\mu[X_j = x_j]$ for $x_j \in \mathcal{X}_j$ (these prices determine an initial state $\theta_j(x_j)$ for $x_j \in \mathcal{X}_j$), and possibly linear constraints that tie it to the previous variables. The initial prices can be chosen based on the prices of previously existing variables.

We distinguish the following types of variables:

— *Atomic variables.* The market operator explicitly specifies $\mathcal{X}_j$, $X_j(\omega)$, and initial prices. For instance, the presidential outcome in Florida is represented by an atomic variable FL with values in $\{\text{Dem}, \text{Rep}\}$ whose prices were initialized at $\mu[\text{FL} = \text{Dem}] = 0.51$ and $\mu[\text{FL} = \text{Rep}] = 0.49$ on 9-16-2012.

— *Pairs.* An example of a pair variable is the combined presidential outcome in states FL and OH. The new variable X_j is derived from a pair of existing variables X_k, X_ℓ by setting $\mathcal{X}_j = \mathcal{X}_k \times \mathcal{X}_\ell$ and $X_j(\omega) = (X_k(\omega), X_\ell(\omega))$. We create marginal constraints:

$$\mu[X_k = x_k] = \textstyle\sum_{x_\ell \in \mathcal{X}_\ell} \mu[X_j = (x_k, x_\ell)] \qquad \text{for all } x_k \in \mathcal{X}_k$$
$$\mu[X_\ell = x_\ell] = \textstyle\sum_{x_k \in \mathcal{X}_k} \mu[X_j = (x_k, x_\ell)] \qquad \text{for all } x_\ell \in \mathcal{X}_\ell \ .$$

We use two initialization strategies. If $\mathcal{X}_k = \mathcal{X}_\ell = \{\text{Dem}, \text{Rep}\}$, namely, we are creating a pair of party-valued variables, we let $\mu_1 \coloneqq \mu[\mathcal{X}_k = \text{Rep}]$ and $\mu_2 \coloneqq \mu[\mathcal{X}_\ell = \text{Rep}]$ and use an initialization parametrized by ρ that interpolates between independence (if $\rho = 0$) and maximum correlation (if $\rho = 1$):

$$\mu[\mathcal{X}_j = (\text{Rep}, \text{Rep})] = (1 - \rho)\mu_1\mu_2 + \rho \min\{\mu_1, \mu_2\}.$$

We fill in the remaining initialization prices uniquely to satisfy the marginal constraints. For non-party pairs, we used independent initialization

$$\mu[X_j = (x_k, x_\ell)] = \mu[X_k = x_k]\mu[X_\ell = x_\ell] \ .$$

— *Triples.* An example of a triple variable is the combined outcome in presidential races in FL, OH, and the national race. Triples are defined from three existing variables X_k, X_ℓ, X_m. We assume that all pair variables exist (and if not, we create them first), denoting them $X_{k\ell}$, X_{km}, $X_{\ell m}$. We create marginal constraints with these pair variables, and use the initialization of the form

$$\mu[X_j = (x_k, x_\ell, x_m)] \propto \frac{\mu[X_{k\ell} = (x_k, x_\ell)] \cdot \mu[X_{km} = (x_k, x_m)] \cdot \mu[X_{\ell m} = (x_\ell, x_m)]}{\mu[X_k = x_k] \cdot \mu[X_\ell = x_\ell] \cdot \mu[X_m = x_m]} \ .$$

This initialization guarantees prices are coherent with marginals as long as at least one pair of the variables is conditionally independent conditioned on the third variable. However, in general, the marginal constraints might be initially violated.

— *Counts.* Count variables are for example used to represent how many states in a given region (such as the Midwest or Northeast) will vote for a Democratic or Republican candidate. The user has the ability to specify any range in which the outcome count could fall. Formally, given a set of existing distinct variables $\{X_{j_1}, \ldots, X_{j_n}\}$ and the corresponding values $x_{j_1}, \ldots, x_{j_n}$, we create a new variable X_j monitoring how many of the variables attain the given value. We have $\mathcal{X}_j = \{0, \ldots, n\}$ and

$$X_j(\omega) = \sum_{k=1}^{n} 1[X_{j_k}(\omega) = x_{j_k}] \ .$$

We again assume that all of the pairwise variables $X_{j_k j_\ell}$ exist, and create the first and second moment constraints:

$$\sum_{k=1}^{n} k \cdot \mu[X_j = k] = \sum_{k=1}^{n} \mu[X_{j_k} = x_{j_k}]$$
$$\sum_{k=1}^{n} k^2 \cdot \mu[X_j = k] = \sum_{k=1}^{n} \Big(\mu[X_{j_k} = x_{j_k}] + \sum_{\ell \neq k} \mu[X_{j_k j_\ell} = x_{j_k j_\ell}] \Big) \ .$$

The initial prices are set proportional to a discretized Gaussian distribution with the mean and variance as determined by the right-hand side of the two constraints. Again, note that the initialization does not guarantee that the constraints will hold exactly (but they should not be too violated). In addition, note that $1[X_j = n]$ is equivalent to the conjunction $1[X_{j_1} = x_{j_1} \wedge \cdots \wedge X_{j_n} = x_{j_n}]$ and analogously $1[X_j = 0]$ is equivalent to $1[X_{j_1} \neq x_{j_1} \wedge \cdots \wedge X_{j_n} \neq x_{j_n}]$. For these extreme values, we introduce *clique* and *tree* constraint generators [Dudík et al. 2012].

— *Weighted sums.* An example of a weighted sum variable is the number of electoral votes obtained by a candidate (this is a weighted sum of state-by-state presidential outcomes). We allow sums of the form

$$X_j(\omega) = \sum_{k=1}^{n} a_k 1[X_{j_k}(\omega) = x_{j_k}]$$

where a_k are positive integers and j_k are not necessarily distinct. We set $\mathcal{X} = \{0, \ldots, n\}$ where $n = \sum_k a_k$ if j_k's are distinct, or the sum only includes the largest weight assigned to values of each distinct variable. We create the first and second moment constraints and initialize prices similarly to the count variables.

— *Paths.* As a fun example, we implemented variables modeling whether, in the presidential election, Democrats (or Republicans) would win a path of neighboring states connecting Mexico with Canada, or the Atlantic with the Pacific. We created constraint generators for these variables as follows. Consider the event $E =$ "There is a path of Rep from Mexico to Canada". Then $\mathbb{P}[E]$ is bounded from below by the probability that all the states on a specific path $\mathcal{P}$ vote Republican. Let $A_1, \ldots, A_n$ be the events corresponding to each individual state on the path $\mathcal{P}$ voting Republican, and let $\bar{A}_1, \ldots, \bar{A}_n$ be the complementary events that the states vote Democratic. Then

$$\mathbb{P}[E] \geq \mathbb{P}[\mathcal{P} \text{ votes Rep}] \geq \mathbb{P}[A_1] - \mathbb{P}[A_1 \cap \bar{A}_2] - \mathbb{P}[A_2 \cap \bar{A}_3] - \cdots - \mathbb{P}[A_{n-1} \cap \bar{A}_n]$$

where the last inequality is a particular instance of a tree inequality [Dudík et al. 2012; Galambos and Simonelli 1996]. Finding the tightest lower bound then corresponds to the problem of finding shortest paths where edge weights are of the form $\mu[X_j = (\text{Dem}, \text{Rep})]$ for variables modeling pair outcomes. Since these only provide lower bounds, we use a specific property of our design and also create the inequalities

$$\mu[\text{Dem path Atlantic-Pacific}] + \mu[\text{Rep path Mexico-Canada}] \leq 1$$
$$\mu[\text{Rep path Atlantic-Pacific}] + \mu[\text{Dem path Mexico-Canada}] \leq 1$$

which follow because Mexico-Canada paths act as cuts for Atlantic-Pacific paths and vice-versa (we do not treat states that border by corners as neighbors).

2.3. Implementation

User Interface. When a user logged into the WiseQ game, she interacted with several sections of the interface. First, there was a portfolio that showed what securities the user owned. Second, there were three U.S. maps displaying the current presidential, senatorial and gubernatorial predictions across states. Third, there was both an overall leaderboard and leaderboards for user-created "leagues". Leagues allowed users to compete against their social network with overall scores translated into within-league scores. On leaderboard pages, a user could access the full portfolio of all other users for comparison and research.

A trading wizard was always visible on all pages for users to buy securities (i.e., make predictions). The wizard asked the user to "make a prediction on" and provided a drop-down menu of 23 prediction types including presidential, senatorial and gubernatorial outcomes in states, national outcomes for the Senate, House and presidential elections, numerical variables such as the number of electoral votes won by a candidate, "jobs numbers" for September and October (change in nonfarm payroll) and various combinations. After choosing the prediction type (e.g., a presidential race in a pair of states), the exact contest (e.g., two particular states) and outcomes (e.g., a winning party), the user could see a full sentence describing her prediction, such as: "Democrats will win the presidential election in both Ohio and Virginia". Three values were displayed underneath: the current odds, a fill-in text box for the investment amount with a default initial value, which the user could adjust,

and the return if correct. The user executed the trade by pressing a large "buy prediction" button underneath.

There were three additional ways to execute trades. First, the user could sell anything in her portfolio by clicking the "sell" button that was next to each security owned by the user. Second, the user could click on a state in any of the three U.S. maps to obtain a pre-filled wizard. At that point, the user would just have to click the "buy prediction" button or could adjust any aspect of the prediction in the wizard. Third, WiseQ had a running list of trending predictions and editorial picks, and the user could just click "buy prediction" to execute any of the predictions on that list.

Pricing Engine. The back-end pricing engine for our market consisted of a fully functional Java implementation of Dudík et al.'s [2012] linearly constrained market maker, with the specific price initialization and constraints described in Section 2.2. A goal of their market maker design was the separation of price computations and arbitrage removal to enable near-instantaneous price queries. In our implementation, pricing and arbitrage removal ran in separate threads. If an operation (pricing or trading) was performed for a security that was not yet present in the market, the security was first created and initialized. Price queries on existing securities did not trigger new constraint violations. On the other hand, the execution of trades resulted in price changes, which were followed by price updates for potentially all securities in the market.

3. RESULTS AND ANALYSIS

Our market was launched on 09-16-2012 and was closed at 7pm EST on 11-06, which is when media coverage of the election results began. In our market 680 users signed up for an account, out of which 437 made at least one trade. We restrict our analysis to these active users. The users made 3,137 trades in total, using all 23 possible prediction types, with the first trade placed on 09-16 at 16:37 EDT and the last trade placed on 11-06 at 17:33 EST. Overall, 514 distinct bundles were bought. Several of them were bought by many users (such as the national presidency), but 261 were traded by a unique user, which underscores the fact that a market maker is essential for a combinatorial prediction market.

In our back-end pricing engine the market consisted of 2,840 variables corresponding to 17,222 securities. During the run of the market 20,983 unique constraints were created, and of these 261 constraints were created by constraint generators; the remaining were created at initialization (such as marginal constraints). Of the generated constraints, 92 (35%) were generated before the election night. The remaining 169 (65%) were created on the election night, as the election results in individual races were reported and the market prices were finalized to 0-1 values.

In the remainder of this section we undertake a detailed analysis of the trading data generated by WiseQ. As part of our analysis we aim to address several concrete questions: (1) Was user behavior consistent with risk-neutral profit maximization, or was it distorted because the market used play money rather than real money? (2) Did combinatorial bets contribute to market performance? (3) How much of the market accuracy can be ascribed to user input (trades) rather than well-initialized prices? (4) How sensitive is market accuracy to the choice of liquidity parameter?

Section 3.1 studies the behavior of users to make sure it is consistent with profit maximization, and to understand what distinguishes the top performers. Section 3.2 assesses the accuracy of our market forecasts against several benchmarks. Section 3.3 performs counterfactual experiments to understand how market forecasting performance depends on the liquidity parameter.

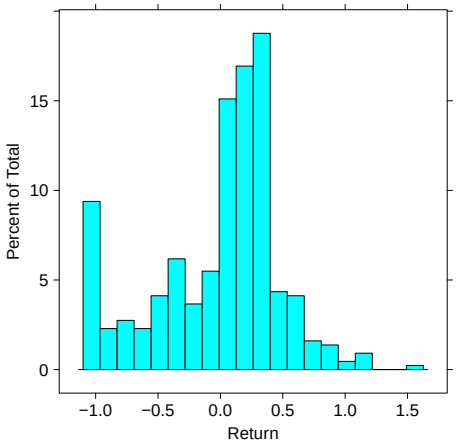

Fig. 1. Distribution of return across user population. The spike at -100% return corresponds to users whose shares all failed to pay out.

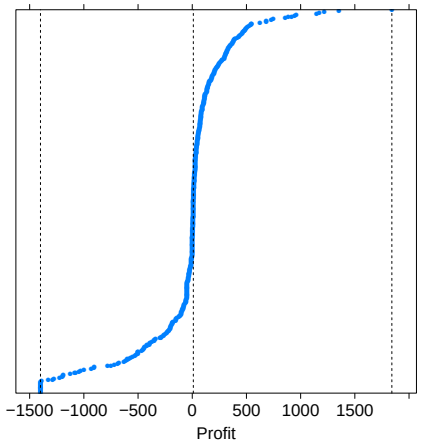

Fig. 2. The dots represent users, who are sorted by increasing profit. Reference lines indicate the minimum, median, and maximum profits.

3.1. User Behavior

User Performance. Each user received 1400 points to buy securities in the market. At any time a user could also liquidate its shares of a particular security. In the following, the *revenue* of a user is the total amount of points received from selling securities, and from 0-1 payouts from securities that were held until market close. The *spend* of a user is the total amount of points spent buying securities. The *profit* of a user is revenue minus spend, while its *return* is profit over spend.

Table I summarizes these metrics across the population of users, listing for each the minimum, maximum, and quartiles. Note that the spend of a user can exceed 1400 points if the user repeatedly liquidates held shares and reinvests the proceeds. The profit and return of a user are lower bounded by 1400 points and -100% respectively, but in principle they have no upper bound. Profit and return quantify user success. Their medians are (slightly) above zero, indicating that the median user improved the accuracy of market prices via her trades. We also note that over three quarters of users never sold any securities, meaning that their revenues are based on the quality of their outcome predictions rather than predictions of price movements.

Table I. Summary statistics on user activity and performance.

Percentile	Revenue	Spend	Profit	Return	Buys	Sells
0%	0.00	1.97	-1400.00	-1.00	1	0
25%	60.83	100.00	-50.07	-0.34	2	0
50%	238.47	350.02	9.93	0.12	4	0
75%	1006.14	1199.97	95.29	0.29	9	0
100%	5041.55	4505.96	1841.55	1.53	64	37

Figure 1 provides a histogram of the user returns. We see a large mass of users to the right of the 0% return mark; these users were all able to improve the market's forecast accuracy. The reason the median return is nevertheless close to 0% is that this mass is offset by a spike at the -100% return mark, corresponding to users whose securities all paid out 0 points (i.e., the corresponding outcomes did not occur). The shape of the return distribution shows

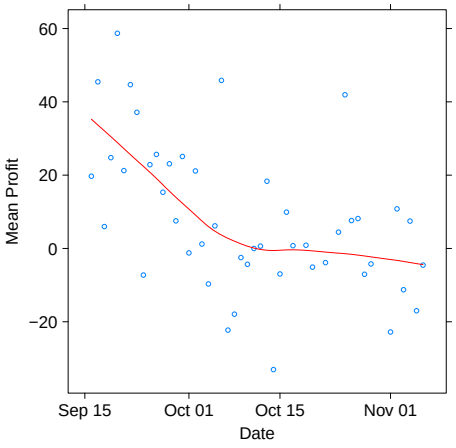

Fig. 3. Mean daily profit of users from market opening to closure, together with LOESS curve showing a downward trend, which slows down once the estimate falls just below zero.

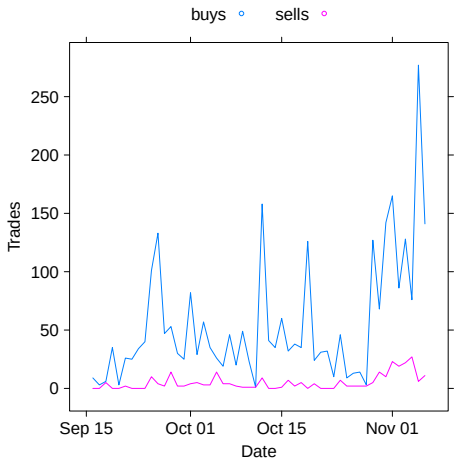

Fig. 4. Time series of the number of 'buy' and 'sell' transactions in the market. The ratio of total buy to sell trades is almost 11 to 1.

that users were not adopting the strategy of spending their budgets on high-risk securities, on the off chance that enough of them would pay out to propel them to the top of the leaderboard. This would have been a plausible strategy—after all, points are worthless in real life so there is no benefit to a positive profit if one still ends up low on the leaderboard—but it would have lead to a return distribution with spikes at the two extremes, which is not the case here. Figure 2 plots user profits in increasing order, which can be interpreted as a cumulative distribution function (CDF) of profit across the user population. The profit distribution is symmetric, with most mass concentrated between -500 and 500 points, and the median just slightly exceeding 0 points.

Figure 3 shows a time series of daily mean profit across users. The local regression curve (LOESS) depicts a trend where profit is steadily decreasing across time until it reaches the 0 mark around October 10th, at which point it only decreases slightly below 0. The interpretation is that agents are steadily incorporating information into the market by taking advantage of profit opportunities until the latter are exhausted.

Table II. Prediction types ranked by user interest.

Prediction Type	**Unique Users**	**Trades per User**
Presidential—Singleton	413	3.87
Senate and House—Singleton	176	3.12
Presidential—Combinatorial	150	3.12
Electoral Votes	63	1.79
Governor	47	1.49
Economic Indicators	37	1.54
Senate, Governor and Presidential—Combinatorial	12	1.50

User Activity. We next examine user activity in terms of the number of trades performed and the kinds of securities transacted. Figure 4 presents a time series of the buy and sell orders in the market. There are 3–4 spikes in the buy series, but they do not correspond to election events like the presidential debates; we suspect they are due to new user sign-ups after advertising our market on social media. Trade activity picks up sharply around 10

Table III. Top singleton securities by spend.

Prediction	Spend	Outcome	% Spend
Federal	43663	Dem	76.3
		Rep	23.6
FL	26034	Dem	48.4
		Rep	51.5
OH	16715	Dem	75.3
		Rep	24.6
VA	14753	Dem	48.9
		Rep	51.0
CO	12143	Dem	58.3
		Rep	41.6
Elect. Votes	10419		
NC	9911	Dem	67.3
		Rep	32.6
Senate	8319	Dem	82.9
		Rep	13.5
		Neither	3.4
House	7500	Dem	16.3
		Rep	83.6
Senate-MA	7221	Dem	94.3
		Rep	5.6
NH	4803	Dem	36.2
		Rep	63.7
IA	4350	Dem	67.1
		Rep	32.8

Table IV. Top combinatorial securities by spend.

Prediction	Spend
VA,WI=Dem	1900
CO,IA,NH,OH,VA=Dem	1650
Fed,OH=Rep	1551
OH,VA=Dem	1393
FL=Rep,Fed=Dem	1349
FL,NC=Rep	1160
FL,Fed,OH=Dem	1010
NH,NV,OH,VA,WI=Dem	980
Path Canada-Mexico=Dem	835
CO,Fed=Dem	800
FL,NC=Rep,Fed=Dem	765
Fed,OH=Dem	584

days prior to the elections, with the highest volume coming on the eve of election day. Sell volume is much lighter, and the ratio of securities held until close to securities eventually liquidated in the market was almost 11 to 1.

As mentioned, users made trades on all 23 possible prediction types in the trading wizard. Table II lists the prediction types ranked by user interest, as measured by unique users and number of trades per user in each category. The prediction types are grouped into seven categories for clarity. The rankings are almost the same under both metrics (the only difference is the rank of the Governor category). Notably, combinatorial securities attracted interest with over a third of users making a combinatorial trade. It does not appear that the order of prediction types in the WiseQ pull-down menu had much influence on the kinds of trades selected. While singleton presidential and senatorial elections appeared at the top of the WiseQ menu and are also most popular in Table II, the menu listed the presidential electoral votes much below the Governor outcomes and combinatorial Senate-Governor-Presidential elections, yet the electoral votes were a more popular prediction type.

Table III shows that when users were buying singleton securities, they were focusing on the most meaningful and uncertain outcomes. Besides the national presidential outcome, the four top predictions by spend were the presidential outcomes in Florida, Ohio, Virginia, and Colorado. The most efficient proxy for outcome uncertainty is the state-by-state advertising spending of official campaigns and their aligned outside groups. Ohio and Florida were nearly tied in spending with just shy of $200 million each, and were the top two individual race securities in our market. Virginia at $150 million was both the third highest spending and third most traded, while Colorado at $80 million was both the fourth highest spending

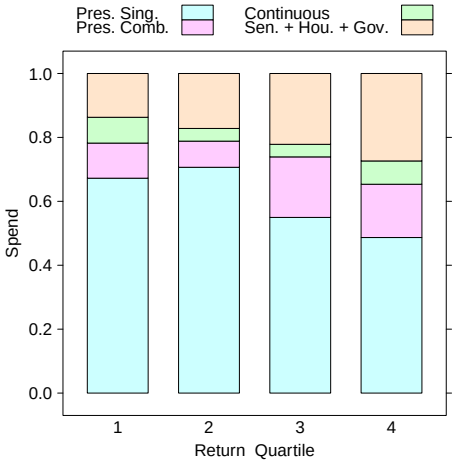

Fig. 5. Allocation of user spend among different prediction categories, where users are segmented by return quartile. A higher-numbered quartile corresponds to higher return. Within each quartile the category spends have been normalized so that they sum to one.

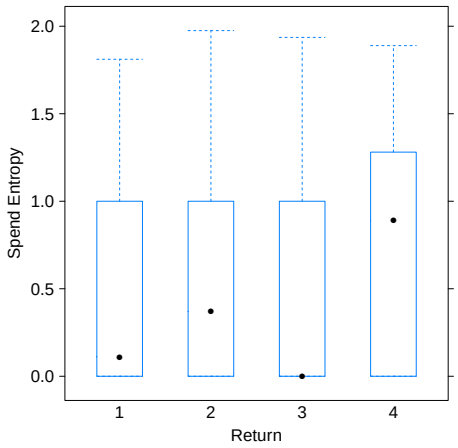

Fig. 6. Boxplots of the entropy of user spend allocation, with users segmented according to return quartiles. A dot represents the median, box edges represent the the 25th and 75th percentiles, and whisker endpoints the minimum and maximum.

and fourth most traded.[1] The next 5 most popular state-by-state presidential races fill out 5 of the 6 next most expensive states in terms of advertising dollars. The top Senate race on the list, Massachusetts, was also the most expensive Senate race by over 25% more than the second most expensive race.[2] Table IV shows that users were focusing their combinatorial spend on how these key states interacted with each other. Of the top 12 combinatorial securities, all but two of them include Florida, Ohio, or Virginia.

User Strategies. We now take a closer look at trading patterns to understand what distinguishes high-performing users from the rest. In order to do this we segment the users into four groups according to return quartiles, and examine their spend allocation among the different prediction categories. We use the prediction categories from Table II, with the Senate-House-Governor securities (singleton and combinatorial) combined into the same category, and the Electoral Vote and Economic Indicator securities also combined into a "Continuous" category.

Figure 5 shows the allocation of spend among the four securities categories just defined, for each quartile of user return. (The breakpoints for the segments can be read from the 'Return' column in Table I.) A clear pattern from this plot is that higher return is associated with greater spend allocation to Senate-House-Governor securities. The users who performed best were those who were most informed on the likely outcomes of the non-presidential races. As we will see in Section 3.3, our priors (i.e., initial prices) were the least accurate for these races, so these races provided the most profitable prediction opportunities. The spend allocations also show that the two highest return quartiles had a higher proportion of spend on combinatorial bets, which is evidence that such bets contribute to market accuracy, although this pattern is less pronounced.

Figure 5 also suggests that the high-return users diversified their spend allocation more among prediction categories than other segments, at least in aggregate. To examine "di-

[1] http://www.nationaljournal.com/hotline/ad-spending-in-presidential-battleground-states-20120620
[2] http://www.opensecrets.org/overview/topraces.php

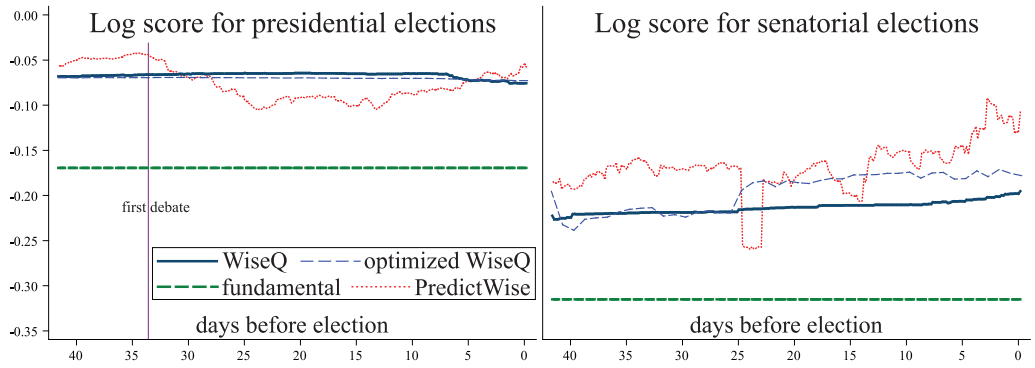

Fig. 7. Average log score for presidential (left) and senatorial (right) races, compiled every four hours. All 51 presidential and 33 senatorial elections are included. Optimized WiseQ performance corresponds to the optimal choice of the liquidity parameter determined in Section 3.3.

versification" at the user level, we consider the distributions of user spend entropy within each return quartile. The entropy of a user's spend is defined as $-\sum_c s_c \log_2 s_c$, where s_c is the normalized user spend in category c (so that normalized spends sum to 1), and c ranges across the four prediction categories. Higher spend entropy corresponds to a more uniform allocation of spend across the categories; the lowest possible entropy is 0 bits (all trades are in the same category) and the highest possible here is 2 bits (since there are four categories). Figure 6 summarizes the distributions of spend entropy within the four user return segments. Spend entropy clearly distinguishes the highest-return segment from the rest: the median is much higher, and combined with the higher 75th percentile this shows that the distribution is more skewed towards uniform spend allocation.

3.2. Market Accuracy

We compare the accuracy of WiseQ with two external benchmarks. The comparison is limited to simple variables representing presidential and senatorial outcomes since these are the only outcomes where standard forecasts are available. The first benchmark is the *fundamental model* of Hummel and Rothschild [2013], which is a statistical model based on presidential approval, incumbency, past election results, economic indicators, measures of ideology, and biographical information. The second benchmark is the PredictWise[3] aggregator [Rothschild 2013], which combines signals from polling, external prediction markets such as Betfair and Intrade, and the fundamental model. Note that we do not compare with pure polling and pure external prediction markets, because both have higher errors than PredictWise, and neither of them can generate a complete set of forecasts during our time frame. The fundamental forecasts were determined on February 15 for the presidential elections and June 15 for the senatorial elections, whereas PredictWise forecasts were generated and published in real-time. PredictWise is similar to the popular New York Times blog FiveThirtyEight[4] by Nate Silver, but FiveThirtyEight is only updated sporadically and does not include prediction market data, which is key to achieving accuracy earlier in the cycle and to obtaining real-time updates for major events.

Figure 7 illustrates how the log score evolves over the evaluation period (higher values correspond to better forecasts). WiseQ is more accurate than the benchmarks during most of the time frame for the presidential races and slightly less accurate for the senatorial races. We also plot the "optimized" WiseQ performance which corresponds to the optimal

[3] `http://www.predictwise.com/`
[4] `http://fivethirtyeight.blogs.nytimes.com/`

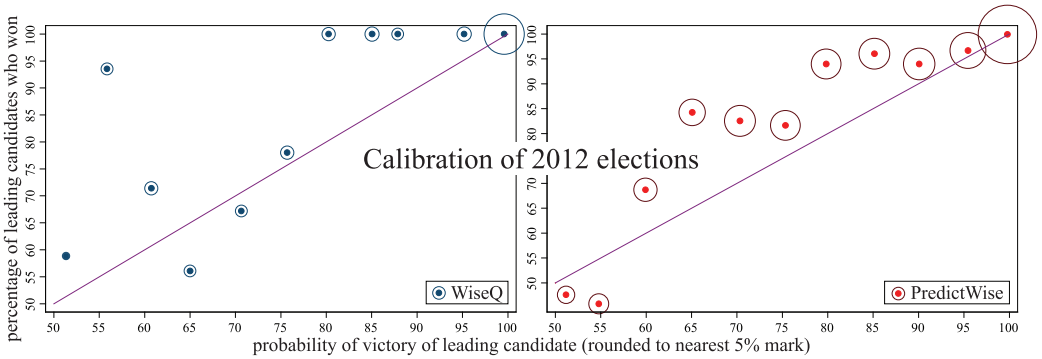

Fig. 8. Calibration for WiseQ (left) and PredictWise (right). We compile the calibrations from the forecasts obtained every four hours. All 51 presidential and 33 senatorial races are included. The size of the outside circles corresponds to the number of distinct races included in the bin.

choice of the liquidity parameter as determined by counterfactual experiments in Section 3.3. Figure 7 shows that optimizing the liquidity parameter would have made WiseQ a much tighter competitor in the Senate.

A second test of a forecast is its calibration and again WiseQ compares reasonably with PredictWise. In Figure 8 we first bin forecasts to the nearest 5% mark according to the probability they ascribe to the leading candidate (i.e, between 50% and 100%). For each bin, we plot the fraction of outcomes that come true. A perfect calibration would correspond to the average forecast equal to the average probability of occurrence in each bin. To provide enough data, we include all 84 presidential and senatorial races and use forecasts from every four-hour interval. The size of the outside circles reflects the number of distinct races included in the bin. Since PredictWise has more variation during the time frame, there are many more distinct races included in each bin, yielding a lower variance in our calibration estimates. Both WiseQ and PredictWise are under-confident for the most certain races, as too high a fraction of forecasts in the 70s and 80s comes true. Yet, while PredictWise continues to show under-confidence, WiseQ is relatively well calibrated for more uncertain races where the leading candidate is between 50 and 75 percent; this portion of the curve corresponds to the outcomes that are most important for stakeholders.

The next step beyond individual outcomes is to consider the pairwise combinations of all 51 state-by-state presidential outcomes. While WiseQ forms these meaningful forecasts directly, standard forecasts do not provide these probabilities without additional assumptions. To obtain pairwise forecasts from the singleton PredictWise forecasts we consider two standard heuristics: the independence heuristic and the ranking heuristic [Chen et al. 2008a]. The independence heuristic assumes that outcomes in the two states are statistically independent. The ranking heuristic first orders states according to forecasted victory probabilities of one candidate, say Obama. It then assumes that in the final outcome, a state can be a victory only if all the higher ranked states are victories. For example, consider two states, X and Y, where X is 55% likely to go for Obama and Y is 40% likely to go for Obama. The likelihood of X going for Romney and Y going for Obama is then 0% by the ranking assumption. The likelihood of both states going for Obama is the likelihood of the less likely state, or 40%, and the likelihood of both states going for Romney is similarly determined as 45%. Thus, the likelihood of X going for Obama and Y going for Romney is the remainder, namely 15%. With four possible outcomes for any state pair, there is a total of 5,100 pairwise outcomes. Figure 9 shows the average log score of WiseQ predictions and the two heuristics applied to PredictWise. The ranking heuristic is consistently more accurate

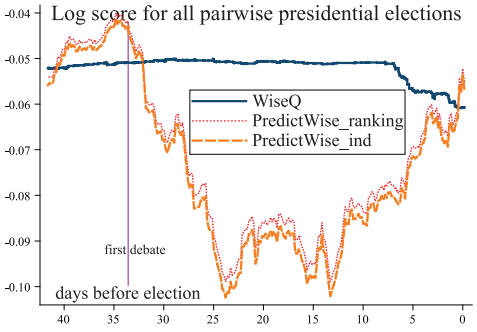

Fig. 9. Average log score for presidential outcomes across all pairs of states, compiled every four hours. There are four possible outcomes for 1,275 pairwise state combinations for a total of 5,100 outcomes for each time period.

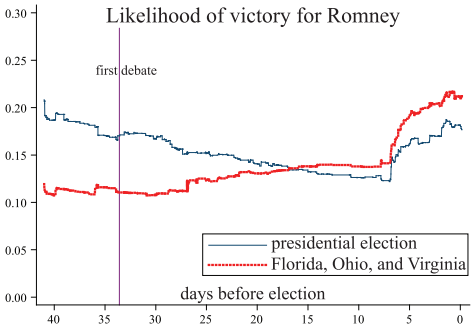

Fig. 10. Likelihood of Romney's victory in national elections and the trifecta of Florida, Ohio, and Virginia. We compile the forecasts every hour.

than independence. WiseQ's performance gain over PredictWise is even more pronounced here than for individual races in Figure 7.

WiseQ uniquely answers critical questions for political science and economics that standard forecasts cannot address. Obama and Romney were two firms spending billions of dollars in pursuit of a single outcome—winning the presidency—and WiseQ data provides meaningful conditional likelihoods that could help them determine efficient resource allocation. That same type of data also answers critical research questions about the nature of campaigns and elections.

One example that pundits and campaigns prominently examined during the 2012 election, generally without data, were the conditional likelihoods of victory in the national election and the key swing states of Ohio, Florida, and Virginia. On the morning after the first debate, October 4, Obama was over 90% likely to win the presidency if he won Ohio, and similarly for Florida (95%) and Virginia (92%). Yet, Romney was well less than 50% likely to win the presidency conditional on winning Ohio (41%), Florida (32%), or Virginia (31%). Romney's chances of taking all of these states were slim: WiseQ had that possibility at just over 11%. Our full breakdown had Obama at 21% to carry exactly one of the states, 32% to carry two of the states, and 36% to carry all three states. So, despite making gains in the national polls, these conditional numbers and three-state probabilities provide a clear picture explaining why Romney was still so unlikely to win the election, at just 17% that morning. Figure 10 highlights this relationship during the 2012 elections by showing how the likelihood of Romney winning all three states (Ohio, Florida, and Virginia) closely tracked the likelihood of winning the national elections; they are never more than a few percentage points on either side of each other. This relationship becomes even tighter as Election Day approaches.

For many questions there is an added value to the stakeholders in probability distributions versus just point estimates or single probabilities. WiseQ tracks these for economic indicators and the number of electoral votes. That same morning after the first debate, the pundits quickly turned to the key economic indicator that was due out the following day, the September change in nonfarm payroll. Briefing released a point estimate earlier that week of 165,000, an alternate expectation of 120,000, and the prior month was 142,000.[5] At 8:30 AM on October 5 the Department of Labor announced just 114,000 new jobs for September. WiseQ was able to show a full probability distribution of the likely jobs numbers, shown in

[5] `http://biz.yahoo.com/c/ec/201240.html`

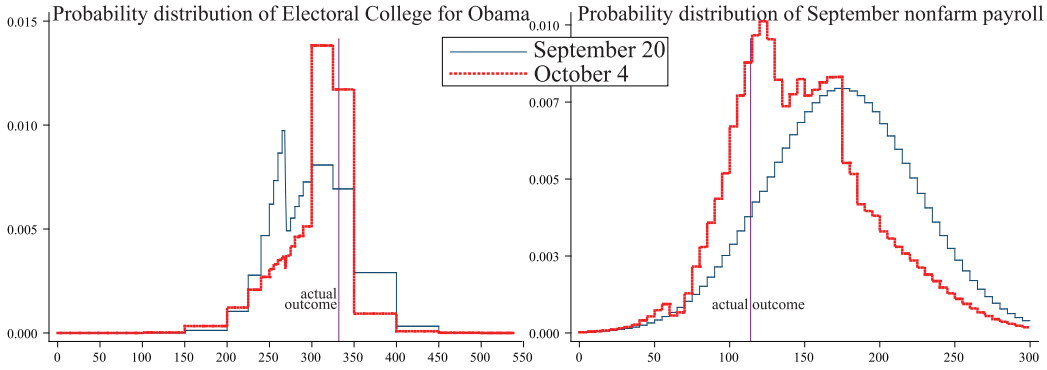

Fig. 11. Histograms for electoral votes and September Jobs numbers on October 4 and September 20 (initialization). We bin the histograms in the exact manner as our user interface.

Figure 11 on the right-hand side. The comparison of the October 4 distribution with the initial distribution from September 20 reveals how the market absorbed new information and moved away from a smooth distribution centered just above the prior month towards the actual outcome.

The left-hand side of Figure 11 shows how the distribution for the number of electoral votes for Obama changed from its initialization on September 20. The final answer was located on the peak in the 300 to 350 range. The users created this single-peaked distribution from the initialization that had two separate and smaller peaks; one peak was narrow and on the Romney side of victory, and the other peak was wider and on the Obama side of victory.

3.3. Counterfactual Analysis

A key difficulty in running automated market makers is the choice of the liquidity parameter. In this section, we evaluate how far we deviated from the optimal choice and how it affected our predictive performance. We also compare the linearly constrained market maker with an approach using independent markets for each variable.

Both styles of comparisons are *counterfactual* since we cannot actually rerun history with different market parameters and witness how real traders would react. Our approach to a counterfactual evaluation is to take the realized set of buy/sell transactions and transform them into a sequence of limit orders, and then repeatedly execute the sequence of limit orders under different conditions. Limit orders are triples, specifying a bundle to buy, the maximum price per share, and the maximum amount to spend (maximum cost). We execute them by spending the money until either the maximum price or the maximum cost is reached, whichever comes first.

The collected user data consists of market orders, so it only contains the number of shares bought/sold and their total cost/revenue. Based on this information we construct limit orders as follows:

— Each sell transaction is first transformed into a complementary buy transaction: a sale of q shares of a bundle $1[X_j \in A]$ with the revenue r is transformed into a purchase of q shares of a bundle $1[X_j \notin A]$ with the cost $q - r$.
— For a buy transaction of q shares of a bundle $1[X_j \in A]$ for the cost c, we first determine the revealed share price $p = c/q$. We impute the unobserved value v by sampling uniformly from the interval $[p, 1]$, and then create a limit order for the bundle $1[X_j \in A]$ with the maximum price v and the maximum cost c.

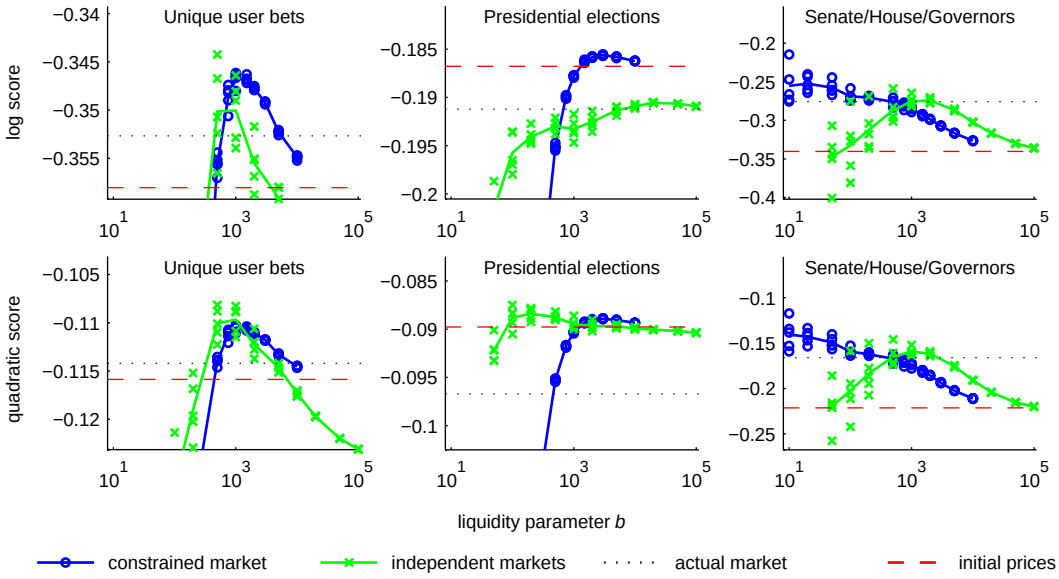

Fig. 12. Market's predictive performance for different liquidity parameters across different security types.

We create five replicates using this protocol (obtaining transactions with different values v). In all experiments below we evaluate markets on all five replicates and report results of the individual five runs as well as the average performance.

To evaluate the performance of various markets we use two scoring rules:

— *log score (log likelihood).* A bundle forecast $f = \mu[X_j \in A]$ is scored as a function of the actual outcome ω, which translates into a binary outcome $f^* = 1[X_j(\omega) \in A]$, as

$$f^* \log f + (1 - f^*) \log(1 - f) \ .$$

A set of forecasts for a variable X_j, when the final realization is $x^* = X_j(\omega)$, is scored according to the log likelihood as

$$\log \mu[X_j = x^*] \ .$$

— *quadratic score.* A forecast f for a binary outcome f^* is scored as

$$-(f - f^*)^2 \ .$$

A set of forecasts for a variable X_j, when the final realization is x^*, is scored as

$$-\sum_{x \in \mathcal{X}_j} \left(\mu[X_j = x] - 1[x^* = x]\right)^2 \ .$$

Values of both scores are negative, and larger score values correspond to better forecasts. Perfect forecasts, where $f = f^*$ (for bundles) or $\mu[X_j = x] = 1[x^* = x]$ (for variables), receive the maximum score of zero in both cases.

In Figure 12, we evaluate the performance of the linearly constrained market maker as well as the independent markets for varying liquidity parameters. We show the average performance across three groups of bets. First, we show the average score across all unique bundles created by users. Second, we show the average performance across all variables derived for presidential variables. Third, we show the performance across variables derived for the Senate-House-Governor races (abbreviated as SHG).

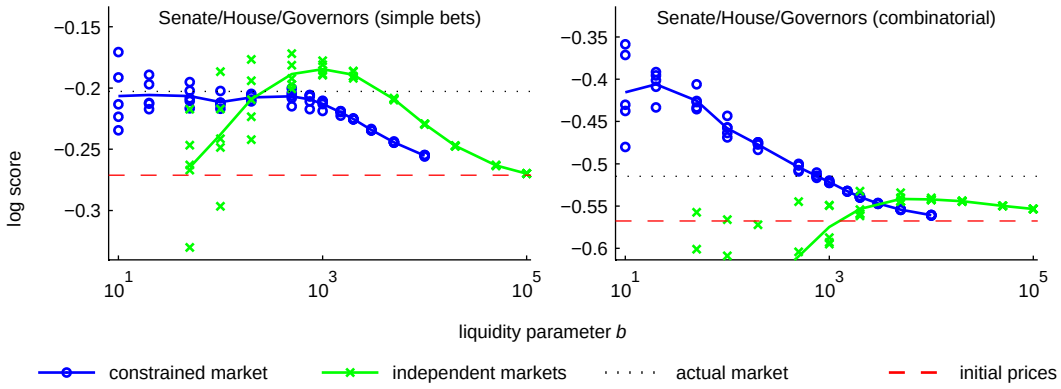

Fig. 13. Counterfactual evaluation of market's predictive performance on a subset of securities corresponding to simple and combinatorial bets on Senate, House and governors.

According to the log score, the optimized linearly constrained market maker outperforms the optimized independent markets in all three groups. This means that propagating information across related securities indeed improves the predictive performance of the market. Also note that the constrained market is more robust to variations in user beliefs than independent markets.

As we saw in Section 3.2, in the presidential elections, the initialization prices were very accurate and optimizing the liquidity parameter does not dramatically improve over the initialization. On the other hand, in the SHG races, a more dramatic improvement over the initialization is achievable and our actual market came close to achieving it. Across user bets, our actual market went about half-way from the initialization towards the optimal performance. The optimal liquidity for presidential outcomes was around 3,000, while the optimal liquidity for SHG was about 20, reflecting lower user activity (see Section 3.1). The optimal liquidity for user bundles was about 1,000. The broad range of optimal liquidity values suggests that for the best market accuracy, we should differentiate the liquidity across different bet types.

The counterfactual performance of the linearly constrained market maker crosses the actual performance in the range of 500–750. This is broadly consistent with the actual values that we used: 1,000 for time period 9/16–9/27; 500 for time period 9/27–10/29; and 350 for time period 10/29–11/14.

Compared with log score results, quadratic score results are less sharp—the improvement of linearly constrained market over independent markets remains strong for the SHG outcomes, but there is no improvement on the other two groups, and in fact, independent markets appear to have a small edge in presidential outcomes. Note that this is not too surprising since our market maker is designed to optimize the log score rather than quadratic score.

In Figure 13, we take a closer look at the performance on SHG variables. These are the variables where the market most significantly outperformed the initialization. We separate the bets into simple bets and combinatorial bets. Simple bets are bets on Senate and governor race winners in individual states. Combinatorial bets are bets on the number of races that each party will win, among Senate, governor and presidential races. In both cases, our market significantly outperformed the initialization, and in the case of simple bets, our market was quite close to the optimal performance. For combinatorial bets, however, we could have improved predictive performance much further. In the case of combinatorial bets, we also clearly see the benefits of a linearly constrained market. The optimal performance

of independent markets lags behind the constrained market, even if independent markets are optimized.

4. DISCUSSION AND CONCLUSION

The large-scale case study described in this paper enabled us to test a new market maker, a new front-end trading wizard, and probe critical questions in political science. Our play-money combinatorial prediction market was able to provide additional accuracy to established forecast categories (e.g., Electoral College outcomes), but more importantly, provide accurate and real-time predictions for a host of relevant categories that are challenging for statistical models and polls: combinatorial outcomes, and less prominent categories such as gubernatorial outcomes.

We evaluated several aspects of our market including user behavior, forecast accuracy, and sensitivity to liquidity settings, but the case study itself yielded some further questions and potential answers. As we saw in our counterfactual experiments, different groups of variables may benefit from using different liquidity parameters. Other considerations include the default investment size, trading wizard design, and additional variables. The default settings have a serious impact on investment choices; this includes both the size of the investment and the actual securities purchased.

We see the self-selection of qualified participants (users with the most valuable information) as an advantage of prediction markets that should hold true in domains beyond politics. We plan to run and evaluate our market maker in other domains such as sports, business, or entertainment.

REFERENCES

Abernethy, J., Chen, Y., and Wortman, J. 2011. An optimization-based framework for automated market-making. In *ACM Conference on Electronic Commerce*.

Chen, K., Ingersoll, J., and Kaplan, E. 2008a. Modeling a presidential prediction market. *Management Science 54,* 8, 1381–1394.

Chen, Y., Fortnow, L., Lambert, N., Pennock, D. M., and Wortman, J. 2008b. Complexity of combinatorial market makers. In *ACM Conference on Electronic Commerce*. ACM, New York, NY, USA, 190–199.

Chen, Y. and Pennock, D. M. 2007. A utility framework for bounded-loss market makers. In *Conference on Uncertainty in Artificial Intelligence*. 49–56.

Dudík, M., Lahaie, S., and Pennock, D. M. 2012. A tractable combinatorial market maker using constraint generation. In *ACM Conference on Electronic Commerce*.

Galambos, J. and Simonelli, I. 1996. *Bonferroni-Type Inequalities with Applications*. Springer, New York.

Hanson, R. D. 2003. Combinatorial information market design. *Information Systems Frontiers 5,* 1, 107–119.

Hanson, R. D. 2007. Logarithmic market scoring rules for modular combinatorial information aggregation. *Journal of Prediction Markets 1,* 1, 1–15.

Hummel, P. and Rothschild, D. 2013. Fundamental models for forecasting elections. `ResearchDMR.com/HummelRothschild_FundamentalModel`.

Rothschild, D. 2009. Forecasting elections: Comparing prediction markets, polls, and their biases. *Public Opinion Quarterly 73,* 5, 895–916.

Rothschild, D. 2013. Combining forecasts: Accuracy and timeliness. `ResearchDMR.com/RothschildForecast12`.