An Introduction to Artificial Prediction Markets for Classification

Adrian Barbu   ABARBU@FSU.EDU
Department of Statistics
Florida State University
Tallahassee, FL 32306, USA

Nathan Lay   NLAY@FSU.EDU
Department of Scientific Computing
Florida State University
Tallahassee, FL 32306, USA

Abstract

Prediction markets are used in real life to predict outcomes of interest such as presidential elections. This paper presents a mathematical theory of artificial prediction markets for supervised learning of conditional probability estimators. The artificial prediction market is a novel method for fusing the prediction information of features or trained classifiers, where the fusion result is the contract price on the possible outcomes. The market can be trained online by updating the participants' budgets using training examples. Inspired by the real prediction markets, the equations that govern the market are derived from simple and reasonable assumptions. Efficient numerical algorithms are presented for solving these equations. The obtained artificial prediction market is shown to be a maximum likelihood estimator. It generalizes linear aggregation, existent in boosting and random forest, as well as logistic regression and some kernel methods. Furthermore, the market mechanism allows the aggregation of specialized classifiers that participate only on specific instances. Experimental comparisons show that the artificial prediction markets often outperform random forest and implicit online learning on synthetic data and real UCI datasets. Moreover, an extensive evaluation for pelvic and abdominal lymph node detection in CT data shows that the prediction market improves adaboost's detection rate from 79.6% to 81.2% at 3 false positives/volume.

Keywords: online learning, ensemble methods, supervised learning, random forest, implicit online learning.

1. Introduction

Prediction markets, also known as information markets, are forums that trade contracts that yield payments dependent on the outcome of future events of interest. They have been used in the US Department of Defense (Polk et al., 2003), health care (Polgreen et al., 2006), to predict presidential elections (Wolfers and Zitzewitz, 2004) and in large corporations to make informed decisions (Cowgill et al., 2008). The prices of the contracts traded in these markets are good approximations for the probability of the outcome of interest (Manski, 2006; Gjerstad and Hall, 2005). prediction markets are capable of fusing the information

that the market participants possess through the contract price. For more details, see Arrow et al. (2008).

In this paper we introduce a mathematical theory for simulating prediction markets numerically for the purpose of supervised learning of probability estimators. We derive the mathematical equations that govern the market and show how can they be solved numerically or in some cases even analytically. An important part of the prediction market is the contract price, which will be shown to be an estimator of the class-conditional probability given the evidence presented through a feature vector $\mathbf{x}$. It is the result of the fusion of the information possessed by the market participants.

The obtained artificial prediction market turns out to have good modeling power. It will be shown in Section 3.1 that it generalizes linear aggregation of classifiers, the basis of boosting (Friedman et al., 2000; Schapire, 2003) and random forest (Breiman, 2001). It turns out that to obtain linear aggregation, each market participant purchases contracts for the class it predicts, regardless of the market price for that contract. Furthermore, in Sections 3.2 and 3.3 will be presented special *betting functions* that make the prediction market equivalent to a logistic regression and a kernel-based classifier respectively.

We introduce a new type of classifier that is *specialized* in modeling certain regions of the feature space. Such classifiers have good accuracy in their region of specialization and are not used in predicting outcomes for observations outside this region. This means that for each observation, a different subset of classifiers will be aggregated to obtain the estimated probability, making the whole approach become a sort of *ad-hoc* aggregation. This is contrast to the general trend in boosting where the same classifiers are aggregated for all observations.

We give examples of generic specialized classifiers as the leaves of random trees from a random forest. Experimental validation on thousands of synthetic datasets with Bayes errors ranging from 0 (very easy) to 0.5 (very difficult) as well as on real UCI data show that the prediction market using the specialized classifiers outperforms the random forest in prediction and in estimating the true underlying probability.

Moreover, we present experimental comparisons on many UCI datasets of the artificial prediction market with the recently introduced implicit online learning (Kulis and Bartlett, 2010) and observe that the market significantly outperforms the implicit online learning on some of the datasets and is never outperformed by it.

2. The Artificial Prediction Market for Classification

This work simulates the *Iowa electronic market* (Wolfers and Zitzewitz, 2004), which is a real prediction market that can be found online at http://www.biz.uiowa.edu/iem/.

2.1 The Iowa Electronic Market

The *Iowa electronic market* (Wolfers and Zitzewitz, 2004) is a forum where contracts for future outcomes of interest (e.g. presidential elections) are traded.

Contracts are sold for each of the possible outcomes of the event of interest. The contract price fluctuates based on supply and demand. In the Iowa electronic market, a winning contract (that predicted the correct outcome) pays \$1 after the outcome is known. Therefore, the contract price will always be between 0 and 1.

Our market will simulate this behavior, with contracts for all the possible outcomes, paying 1 if that outcome is realized.

2.2 Setup of the Artificial Prediction Market

If the possible classes (outcomes) are $1, ..., K$, we assume there exist contracts for each class, whose prices form a K-dimensional vector $\mathbf{c} = (c_1, ..., c_K) \in \Delta \subset [0,1]^K$, where Δ is the probability simplex $\Delta = \{\mathbf{c} \in [0,1]^K, \sum_{k=1}^K c_k = 1\}$.

Let $\Omega \subset \mathbb{R}^F$ be the instance or feature space containing all the available information that can be used in making outcome predictions $p(Y = k|\mathbf{x}), \mathbf{x} \in \Omega$.

The market consists of a number of market participants $(\beta_m, \phi_m(\mathbf{x}, \mathbf{c})), m = 1, ..., M$.

A *market participant* is a pair $(\beta, \phi(\mathbf{x}, \mathbf{c}))$ of a *budget* β and a *betting function* $\phi(\mathbf{x}, \mathbf{c})$: $\Omega \times \Delta \to [0,1]^K, \phi(\mathbf{x}, \mathbf{c}) = (\phi^1(\mathbf{x}, \mathbf{c}), ..., \phi^K(\mathbf{x}, \mathbf{c}))$. The budget β represents the weight or importance of the participant in the market. The betting function tells what percentage of its budget this participant will allocate to purchase contracts for each class, based on the instance $\mathbf{x} \in \Omega$ and the market price $\mathbf{c}$. As the market price $\mathbf{c}$ is not known in advance, the betting function describes what the participant plans to do for each possible price $\mathbf{c}$. The betting functions could be based on trained classifiers $h(\mathbf{x}) : \Omega \to \Delta, h(\mathbf{x}) = (h^1(\mathbf{x}), ..., h^K(\mathbf{x})), \sum_{k=1}^K h^k(\mathbf{x}) = 1$, but they can also be related to the feature space in other ways. We will show that logistic regression and kernel methods can also be represented using the artificial prediction market and specific types of betting functions. In order to bet at most the budget β, the betting functions must satisfy $\sum_{k=1}^K \phi^k(\mathbf{x}, \mathbf{c})) \leq 1$.
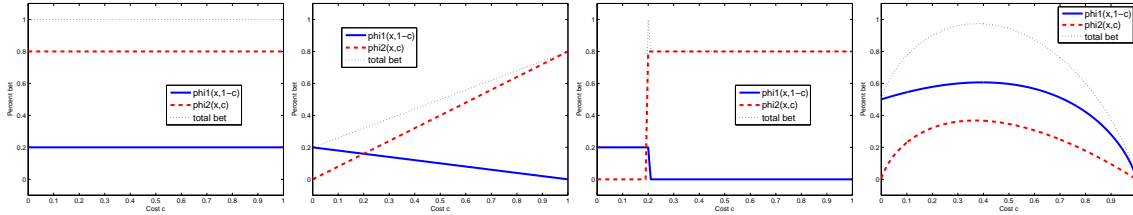

Figure 1: Betting function examples: a) Constant, b) Linear, c) Aggressive, d) Logistic. Shown are $\phi^1(\mathbf{x}, 1-c)$ (red), $\phi^2(\mathbf{x}, c)$ (blue), and the total amount bet $\phi^1(\mathbf{x}, 1-c) + \phi^2(\mathbf{x}, c)$ (black dotted). For a) through c), the classifier probability is $h^2(\mathbf{x}) = 0.2$.

Examples of betting functions include the following, also shown in Figure 1:

- Constant betting functions
$$\phi^k(\mathbf{x}, \mathbf{c}) = \phi^k(\mathbf{x})$$
for example based on trained classifiers $\phi^k(\mathbf{x}, \mathbf{c}) = \eta h^k(\mathbf{x})$, where $\eta \in (0,1]$ is constant.

- Linear betting functions
$$\phi^k(\mathbf{x}, \mathbf{c}) = (1 - c_k)h^k(\mathbf{x}) \tag{1}$$

- Aggressive betting functions
$$\phi^k(\mathbf{x}, \mathbf{c}) = h^k(\mathbf{x}) \begin{cases} 1 & \text{if } c_k \leq h^k(\mathbf{x}) \\ 0 & \text{if } c_k > h^k(\mathbf{x}) + \epsilon \\ \frac{h^k(\mathbf{x}) + \epsilon - c_k}{\epsilon} & \text{otherwise} \end{cases} \tag{2}$$

3

- Logistic betting functions:

$$\phi_m^1(\mathbf{x}, 1 - c) = (1 - c)(x_m^+ - \ln(1 - c)/B),$$
$$\phi_m^2(\mathbf{x}, c) = c(-x_m^- - \ln c/B)$$

where $x^+ = xI(x > 0), x^- = xI(x < 0)$ and $B = \sum_m \beta_m$.

The betting functions play a similar role to the *potential functions* from maximum entropy models (Berger et al., 1996; Ratnaparkhi et al., 1996; Zhu et al., 1998), in that they make a conversion from the feature output (or classifier output for some markets) to a common unit of measure (energy for the maximum entropy models and money for the market).

The contract price does not fluctuate in our setup, instead it is governed by Equation (4). This equation guarantees that at this price, the total amount obtained from selling contracts to the participants is equal to the total amount won by the winning contracts, independent of the outcome.
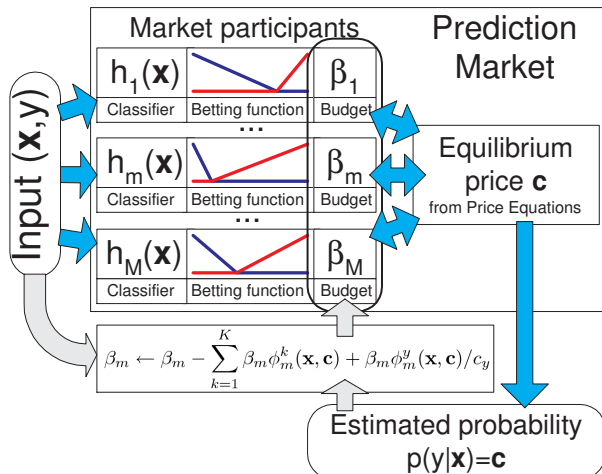

Figure 2: Online learning and aggregation using the artificial prediction market. Given feature vector $\mathbf{x}$, a set of market participants will establish the market equilibrium price $\mathbf{c}$, which is an estimator of $P(Y = k|\mathbf{x})$. The equilibrium price is governed by the Price Equations (4). Online training on an example $(\mathbf{x}, y)$ is achieved through **Budget Update** $(\mathbf{x}, y, \mathbf{c})$ shown with gray arrows.

2.3 Training the Artificial Prediction Market

Training the market involves initializing all participants with the same budget β_0 and presenting to the market a set of training examples $(\mathbf{x}_i, y_i), i = 1, ..., N$. For each example $(\mathbf{x}_i, y_i)$ the participants purchase contracts for the different classes based on the market price $\mathbf{c}$ (which is not known yet) and their budgets β_m are updated based on the contracts purchased and the true outcome y_i. After all training examples have been presented, the participants will have budgets that depend on how well they predicted the correct class y for each training example $\mathbf{x}$. This procedure is illustrated in Figure 2.

Algorithm 1 Budget Update $(\mathbf{x}, y, \mathbf{c})$

Input: Training example $(\mathbf{x}, y)$, price $\mathbf{c}$
for $m = 1$ **to** M **do**
   Update participant m's budget as

$$\beta_m \leftarrow \beta_m - \sum_{k=1}^{K} \beta_m \phi_m^k(\mathbf{x}, \mathbf{c}) + \frac{\beta_m}{c_y} \phi_m^y(\mathbf{x}, \mathbf{c}) \tag{3}$$

end for

Algorithm 2 Prediction Market Training

Input: Training examples $(\mathbf{x_i}, y_i), i = 1, ..., N$
Initialize all budgets $\beta_m = \beta_0, m = 1, ..., M$.
for each training example $(\mathbf{x}_i, y_i)$ **do**
   Compute equilibrium price $\mathbf{c}_i$ using Eq. 4
   Run **Budget Update** $(\mathbf{x}_i, y_i, \mathbf{c}_i)$
end for

The budget update procedure subtracts from the budget of each participant the amounts it bets for each class, then rewards each participant based on how many contracts it purchased for the correct class.

Participant m purchased $\beta_m \phi_m^k(\mathbf{x}, \mathbf{c})$ worth of contracts for class k, at price c_k. Thus the number of contracts purchased for class k is $\beta_m \phi_m^k(\mathbf{x}, \mathbf{c})/c_k$. Totally, participant m's budget is decreased by the amount $\sum_{k=1}^{K} \beta_m \phi_m^k(\mathbf{x}, \mathbf{c})$ invested in contracts. Since participant m bought $\beta_m \phi_m^y(\mathbf{x}, \mathbf{c})/c_y$ contracts for the correct class y, he is rewarded the amount $\beta_m \phi_m^y(\mathbf{x}, \mathbf{c})/c_y$.

2.4 The Market Price Equations

Since we are simulating a real market, we assume that the total amount of money collectively owned by the participants is conserved after each training example is presented. Thus the sum of all participants' budgets $\sum_{m=1}^{M} \beta_m$ should always be $M\beta_0$, the amount given at the beginning. Since any of the outcomes is theoretically possible for each instance, we have the following constraint:

Assumption 1 *The total budget $\sum_{m=1}^{M} \beta_m$ must be conserved independent of the outcome y.*

This condition transforms into a set of equations that constrain the market price, which we call the price equations. The market price $\mathbf{c}$ also obeys $\sum_{k=1}^{K} c_k = 1$.

Let $B(\mathbf{x}, \mathbf{c}) = \sum_{m=1}^{M} \sum_{k=1}^{K} \beta_m \phi_m^k(\mathbf{x}, \mathbf{c})$ be the total bet for observation $\mathbf{x}$ at price $\mathbf{c}$. We have

Theorem 1 Price Equations. *The total budget $\sum_{m=1}^{M} \beta_m$ is conserved after the* **Budget Update** $(\mathbf{x}, y, \mathbf{c})$, *independent of the outcome y, if and only if $c_k > 0, k = 1, ..., K$ and*

$$\sum_{m=1}^{M} \beta_m \phi_m^k(\mathbf{x}, \mathbf{c}) = c_k B(\mathbf{x}, \mathbf{c}), \quad \forall k = 1, ..., K \tag{4}$$

The proof is given in the Appendix.

2.5 Price Uniqueness

The price equations together with the equation $\sum_{k=1}^{K} c_k = 1$ are enough to uniquely determine the market price $\mathbf{c}$, under mild assumptions on the betting functions $\phi^k(\mathbf{x}, \mathbf{c})$.

Observe that if $c_k = 0$ for some k, then the contract costs 0 and pays 1, so there is everything to win. In this case, one should have $\phi^k(\mathbf{x}, \mathbf{c}) > 0$.

This suggests a class of betting functions $\phi^k(\mathbf{x}, c_k)$ depending only on the price c_k that are continuous and monotonically non-increasing in c_k. If all $\phi_m^k(\mathbf{x}, c_k), m = 1, ..., M$ are continuous and monotonically non-increasing in c_k with $\phi_m^k(\mathbf{x}, 0) > 0$ then $f_k(c_k) = \frac{1}{c_k} \sum_{m=1}^{M} \beta_m \phi_m^k(\mathbf{x}, c_k)$ is continuous and strictly decreasing in c_k as long as $f_k(c_k) > 0$.

To obtain conditions for price uniqueness, we use the following functions

$$f_k(c_k) = \frac{1}{c_k} \sum_{m=1}^{M} \beta_m \phi_m^k(\mathbf{x}, c_k), k = 1, ..., K \tag{5}$$

Remark 2 *If all $f_k(c_k)$ are continuous and strictly decreasing in c_k as long as $f_k(c_k) > 0$, then for every $n > 0$, $n \geq n_k = f_k(1)$ there is a unique $c_k = c_k(n)$ that satisfies $f_k(c_k) = n$.*

The proof is given in the Appendix.

To guarantee price uniqueness, we need at least one market participant to satisfy the following

Assumption 2 *The total bet of participant $(\beta_m, \phi_m(\mathbf{x}, \mathbf{c}))$ is positive inside the simplex Δ, i.e.*

$$\sum_{j=1}^{K} \phi_m^j(\mathbf{x}, c_j) > 0, \ \forall c \in (0, 1)^K, \sum_{j=1}^{K} c_j = 1. \tag{6}$$

Then we have the following result, also proved in the Appendix.

Theorem 3 *Assume all betting functions $\phi_m^k(\mathbf{x}, c_k), m = 1, ..., M, k = 1, ..., K$ are continuous, with $\phi^k(\mathbf{x}, 0) > 0$ and $\phi_m^k(\mathbf{x}, c)/c$ is strictly decreasing in c as long as $\phi_m^k(\mathbf{x}, c) > 0$. If the betting function $\phi_m(\mathbf{x}, \mathbf{c})$ of least one participant with $\beta_m > 0$ satisfies Assumption 2, then for the **Budget Update**$(\mathbf{x}, y, \mathbf{c})$ there is a unique price $\mathbf{c} = (c_1, ..., c_K) \in (0, 1)^K \cap \Delta$ such that the total budget $\sum_{m=1}^{M} \beta_m$ is conserved.*

Observe that all four betting functions defined in Section 2.2 (constant, linear, aggressive and logistic) satisfy the conditions of Theorem 3, so there is a unique price that conserves the budget.

2.6 Solving the Market Price Equations

In practice, a double bisection algorithm could be used to find the equilibrium price, computing each $c_k(n)$ by the bisection method, and employing another bisection algorithm to find n such that the price condition $\sum_{k=1}^{K} c_k(n) = 1$ holds. Observe that the n satisfying $\sum_{k=1}^{K} c_k(n) = 1$ can be bounded from above by

$$n = n \sum_{k=1}^{K} c_k(n) = \sum_{k=1}^{K} c_k(n) f_k(c_k(n)) = \sum_{k=1}^{K} \sum_{m=1}^{M} \beta_m \phi_m^k(\mathbf{x}, \mathbf{c}) \leq \sum_{m=1}^{M} \beta_m$$

because for each m, $\sum_{k=1}^{K} \phi_m^k(\mathbf{x}, \mathbf{c}) \leq 1$.

A potentially faster alternative to the double bisection method is the Mann Iteration (Mann, 1953) described in Algorithm 3. The price equations can be viewed as fixed point equation $F(\mathbf{c}) = \mathbf{c}$, where $F(\mathbf{c}) = \frac{1}{n}(f_1(\mathbf{c}), ..., f_K(\mathbf{c}))$ with $f_k(\mathbf{c}) = \sum_{m=1}^{m} \beta_m \phi_m^k(\mathbf{x}, c_k)$. The Mann iteration is a fixed point algorithm, which makes weighted update steps

$$\mathbf{c}^{t+1} = (1 - \frac{1}{t})\mathbf{c}^t + \frac{1}{t} F(\mathbf{c}^t)$$

The Mann iteration is guaranteed to converge for contractions or pseudo-contractions. However, we observed experimentally that it usually converges in only a few (up to 10) steps, making it about 100-1000 times faster than the double bisection algorithm. If, after a small number of steps, the Mann iteration has not converged, the double bisection algorithm is used on that instance to compute the equilibrium price. However, this happens on less than 0.1% of the instances.

Algorithm 3 Market Price by Mann Iteration

Initialize $i = 1$, $c_k = \frac{1}{K}, k = 1, ..., K$
repeat
$\quad f_k = \sum_m \beta_m \phi_m^k(x, \mathbf{c})$
$\quad n = \sum_k f_k$
$\quad$**if** $n \neq 0$ **then**
$\quad\quad f_k \leftarrow \frac{f_k}{n}$
$\quad\quad r_k = f_k - c_k$
$\quad\quad c_k \leftarrow \frac{(i-1)c_k + f_k}{i}$
$\quad$**end if**
$\quad i \leftarrow i + 1$
until $\sum_k |r_k| \leq \epsilon$ or $n = 0$ or $i > i_{\max}$

2.7 Two-class Formulation

For the two-class problem, i.e. $K = 2$, the budget equation can be simplified by writing $\mathbf{c} = (1 - c, c)$ and obtaining the *two-class market price equation*

$$(1 - c) \sum_{m=1}^{M} \beta_m \phi_m^2(\mathbf{x}, c) - c \sum_{m=1}^{M} \beta_m \phi_m^1(\mathbf{x}, 1 - c) = 0 \tag{7}$$

This can be solved numerically directly in c using the bisection method. Again, the solution is unique if $\phi_m^k(\mathbf{x}, c_k), m = 1, ..., M, k = 1, 2$ are continuous, monotonically non-increasing and obey condition (6). Moreover, the solution is guaranteed to exist if there exist m, m' with $\beta_m > 0, \beta_{m'} > 0$ and such that $\phi_m^2(\mathbf{x}, 0) > 0, \phi_{m'}^1(\mathbf{x}, 1) > 0$.

3. Relation to Existing Supervised Learning Methods

There is a large degree of flexibility in choosing the betting functions $\phi_m(\mathbf{x}, \mathbf{c})$. Different betting functions give different ways to fuse the market participants. In what follows we prove that by choosing specific betting functions, the artificial prediction market behaves like a linear aggregator or logistic regressor, or that it can be used as a kernel-based classifier.

3.1 Constant Betting and Linear Aggregation

For markets with constant betting functions, $\phi_m^k(\mathbf{x}, \mathbf{c}) = \phi_m^k(\mathbf{x})$ the market price has a simple analytic formula, proved in the Appendix.

Theorem 4 Constant Betting. *If all betting function are constant $\phi_m^k(\mathbf{x}, \mathbf{c}) = \phi_m^k(\mathbf{x})$, then the equilibrium price is*

$$\mathbf{c} = \frac{\sum_{m=1}^{M} \beta_m \phi_m(\mathbf{x})}{\sum_{m=1}^{M} \sum_{k=1}^{K} \beta_m \phi_m^k(\mathbf{x})} \tag{8}$$

Furthermore, if the betting functions are based on classifiers $\phi_m^k(\mathbf{x}, \mathbf{c}) = \eta h_m^k(\mathbf{x})$ then the equilibrium price is obtained by linear aggregation

$$\mathbf{c} = \frac{\sum_{m=1}^{M} \beta_m h_m(\mathbf{x})}{\sum_{m=1}^{M} \beta_m} = \sum_m \alpha_m h_m(\mathbf{x}) \tag{9}$$

This way the artificial prediction market can model linear aggregation of classifiers. Methods such as Adaboost (Freund and Schapire, 1996; Friedman et al., 2000; Schapire, 2003) and Random Forest (Breiman, 2001) also aggregate their constituents using linear aggregation. However, there is more to Adaboost and Random Forest than linear aggregation, since it is very important how to construct the constituents that are aggregated.

In particular, the random forest (Breiman, 2001) can be viewed as an artificial prediction market with constant betting (linear aggregation) where all participants are random trees with the same budget $\beta_m = 1, m = 1, ..., M$.

We also obtain an analytic form of the budget update:

$$\beta_m \leftarrow \beta_m - \beta_m \sum_{k=1}^{K} \phi_m^k(\mathbf{x}) + \beta_m \frac{\phi_m^y(\mathbf{x}) \sum_{j=1}^{M} \sum_{k=1}^{K} \beta_j \phi_j^k(\mathbf{x})}{\sum_{j=1}^{M} \beta_j \phi_j^y(\mathbf{x})}$$

which for classifier based betting functions $\phi_m^k(\mathbf{x}, \mathbf{c}) = \eta h_m^k(\mathbf{x})$ becomes:

$$\beta_m \leftarrow \beta_m(1 - \eta) + \eta \beta_m \frac{h_m^y(\mathbf{x}) \sum_{j=1}^{M} \beta_j}{\sum_{j=1}^{M} \beta_j h_j^y(\mathbf{x})}$$

This is a novel online update rule for linear aggregation.

3.2 Prediction Markets for Logistic Regression

A variant of logistic regression can also be modeled using prediction markets, with the following betting functions

$$\phi_m^1(\mathbf{x}, 1 - c) = (1 - c)(x_m^+ - \frac{1}{B} \ln(1 - c)),$$

$$\phi_m^2(\mathbf{x}, c) = c(-x_m^- - \frac{1}{B} \ln c)$$

where $x^+ = xI(x > 0), x^- = xI(x < 0)$ and $B = \sum_m \beta_m$. The two class equation (7) becomes: $\sum_{m=1}^{M} \beta_m c(1-c)(x_m - \ln(1-c)/B + \ln c/B) = 0$ so $\ln \frac{1-c}{c} = \sum_{m=1}^{M} \beta_m x_m$, which gives the logistic regression model

$$\hat{p}(Y = 1|\mathbf{x}) = c = \frac{1}{1 + \exp(\sum_{m=1}^{M} \beta_m \mathbf{x}_m)}$$

The budget update equation $\beta_m \leftarrow \beta_m - \eta \beta_m \left[(1-c)x_m^+ + cx_m^- - H(c)/B \right] + \eta \beta_m u_y(c)$ is obtained, where $u_1(c) = x_m^+ - \ln(1-c)/B, u_2(c) = -x_m^- - \ln(c)/B$.

Writing $\mathbf{x}\beta = \sum_{m=1}^{M} \beta_m \mathbf{x}_m$, the budget update can be rearranged to

$$\beta_m \leftarrow \beta_m - \eta \beta_m \left(x_m - \frac{\mathbf{x}\beta}{B} \right) \left(y - \frac{1}{1 + \exp(\mathbf{x}\beta)} \right). \tag{10}$$

This equation resembles the standard per-observation update equation for online logistic regression:

$$\beta_m \leftarrow \beta_m - \eta x_m \left(y - \frac{1}{1 + \exp(\mathbf{x}\beta)} \right), \tag{11}$$

with two differences. The term $\mathbf{x}\beta/B$ ensures the budgets always sum to B while the factor β_m makes sure that $\beta_m \geq 0$.

The update from eq. (10), like eq. (11) tries to increase $|\mathbf{x}\beta|$, but it does that subject to constraints that $\beta_m \geq 0, m = 1, ..., M$ and $\sum_{m=1}^{M} \beta_m = B$. Observe also that multiplying β by a constant does not change the decision line of the logistic regression.

3.3 Relation to Kernel Methods

Here we construct a market participant from each training example $(x_n, y_n), n = 1, ...N$, thus the number of participants M is the number N of training examples. We construct a participant from training example (x_m, y_m) by defining the following betting functions in terms of $u_m(\mathbf{x}) = \frac{\mathbf{x}_m^T \mathbf{x}}{\|\mathbf{x}_m\|\|\mathbf{x}\|}$:

$$\phi_m^{y_m}(\mathbf{x}) = u_m(\mathbf{x})^+ = \begin{cases} u_m(\mathbf{x}) \text{ if } u_m(\mathbf{x}) \geq 0 \\ 0 \text{ else} \end{cases},$$

$$\phi_m^{2-y_m}(\mathbf{x}) = -u_m(\mathbf{x})^- = \begin{cases} 0 \text{ if } u_m(\mathbf{x}) \geq 0 \\ -u_m(\mathbf{x}) \text{ else} \end{cases} \tag{12}$$

Observe that these betting functions do not depend on the contract price c, so it is a constant market but not one based on classifiers. The two-class price equation gives

$$c = \frac{\sum\limits_m \beta_m \phi_m^2(\mathbf{x})}{\sum\limits_m \beta_m (\phi_m^1(\mathbf{x}) + \phi_m^2(\mathbf{x}))} = \frac{\sum\limits_m \beta_m [y_m u_m(\mathbf{x}) - u_m(\mathbf{x})^-]}{\sum\limits_m \beta_m |u_m(\mathbf{x})|}$$

since it can be verified that $\phi_m^2(\mathbf{x}) = y_m u_m(\mathbf{x}) - u_m(\mathbf{x})^-$ and $\phi_m^1(\mathbf{x}) + \phi_m^2(\mathbf{x}) = |u_m(\mathbf{x})|$.

The decision rule $c > 0.5$ becomes $\sum_m \beta_m \phi_m^2(\mathbf{x}) > \sum_m \beta_m \phi_m^1(\mathbf{x})$ or $\sum_m \beta_m(\phi_m^2(\mathbf{x}) - \phi_m^1(\mathbf{x})) > 0$. Since $\phi_m^2(\mathbf{x}) - \phi_m^1(\mathbf{x}) = (2y_m - 2)u_m(\mathbf{x}) = (2y_m - 2)\frac{\mathbf{x}_m^T \mathbf{x}}{\|\mathbf{x}_m\| \|\mathbf{x}\|}$ (since in our setup $y_m \in \{1, 2\}$), we obtain the SVM type of decision rule with $\alpha_m = \beta_m / \|\mathbf{x}_m\|$:

$$h(\mathbf{x}) = \text{sgn}(\sum_{m=1}^{M} \alpha_m (2y_m - 3)\mathbf{x}_m^T \mathbf{x})$$

The budget update becomes in this case:

$$\beta_m \leftarrow \beta_m - \eta \beta_m |u_m(\mathbf{x})| + \eta \beta_m \frac{\phi_m^y(\mathbf{x})}{c_y}$$

The same reasoning carries out for $u_m(\mathbf{x}) = K(\mathbf{x}_m, \mathbf{x})$ with the RBF kernel $K(\mathbf{x}_m, \mathbf{x}) = \exp(-\|\mathbf{x}_m - \mathbf{x}\|^2 / \sigma^2)$. In Figure 3, left, is shown an example of the decision boundary of a market trained online with an RBF kernel with $\sigma = 0.2$ on 1000 examples uniformly sampled in the $[-1, 1]^2$ interval. In Figure 3, right is shown the estimated probability $\hat{p}(y = 1|\mathbf{x})$.
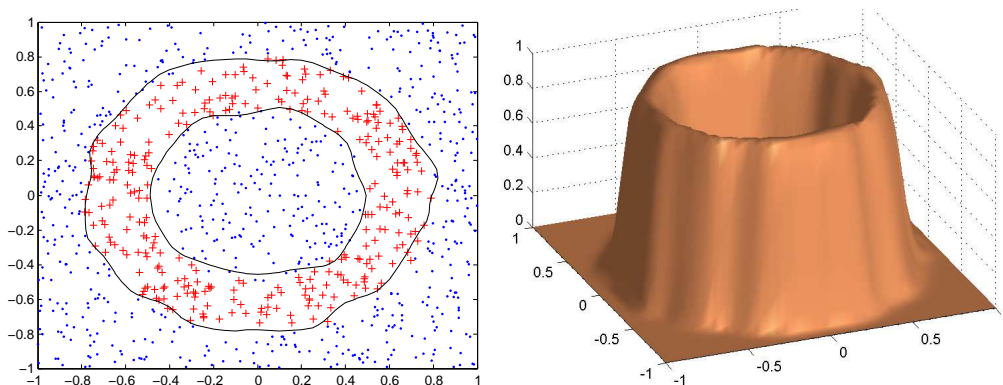

Figure 3: Left: 1000 training examples and learned decision boundary (right) for an RBF kernel-based market from eq. (12) with $\sigma = 0.1$. Right: estimated probability function.

This example shows that the artificial prediction market is an online method with enough modeling power to represent complex decision boundaries such as those given by RBF kernels through the betting functions of the participants. It will be shown in Theorem 5 that the constant market maximizes the likelihood, so it is not clear yet what can be done to obtain a small number of support vectors as in the online kernel-based methods (Bordes et al., 2005; Cauwenberghs and Poggio, 2001; Kivinen et al., 2004).

4. Prediction Markets and Maximum Likelihood

This section discusses what type of optimization is performed during the budget update from eq. (3). Specifically, we prove that the artificial prediction markets perform maximum likelihood learning of the parameters by a version of gradient ascent.

Consider the reparametrization $\gamma = (\gamma_1, ..., \gamma_M) = (\sqrt{\beta_1}, ..., \sqrt{\beta_M})$. The market price $\mathbf{c}(\mathbf{x}) = (c_1(\mathbf{x}), ..., c_K(\mathbf{x}))$ is an estimate of the class probability $p(y = k|\mathbf{x})$ for each instance

$\mathbf{x} \in \Omega$. Thus a set of training observations $(\mathbf{x}_i, y_i), i = 1, ..., N$, since $\hat{p}(y = y_i | \mathbf{x}_i) = c_{y_i}(\mathbf{x}_i)$, the (normalized) log-likelihood function is

$$L(\gamma) = \frac{1}{N} \sum_{i=1}^{N} \ln \hat{p}(y = y_i | \mathbf{x}_i) = \frac{1}{N} \sum_{i=1}^{N} \ln c_{y_i}(\mathbf{x}_i) \tag{13}$$

We will again use the total amount bet $B(\mathbf{x}, \mathbf{c}) = \sum_{m=1}^{M} \sum_{k=1}^{K} \beta_m \phi_m^k(\mathbf{x}, \mathbf{c})$ for observation $\mathbf{x}$ at market price $\mathbf{c}$.

We will first focus on the constant market $\phi_m^k(\mathbf{x}, \mathbf{c}) = \phi_m^k(\mathbf{x})$, in which case $B(\mathbf{x}, \mathbf{c}) = B(\mathbf{x}) = \sum_{m=1}^{M} \sum_{k=1}^{K} \beta_m \phi_m^k(\mathbf{x})$. We introduce a batch update on all the training examples $(\mathbf{x}_i, y_i), i = 1, ..., N$:

$$\beta_m \leftarrow \beta_m + \beta_m \frac{\eta}{N} \sum_{i=1}^{N} \frac{1}{B(\mathbf{x}_i)} \left(\frac{\phi_m^{y_i}(\mathbf{x}_i)}{c_{y_i}(\mathbf{x}_i)} - \sum_{k=1}^{K} \phi_m^k(\mathbf{x}_i) \right). \tag{14}$$

Equation (14) can be viewed as presenting all observations $(\mathbf{x}_i, y_i)$ to the market simultaneously instead of sequentially. The following statement is proved in the Appendix

Theorem 5 ML for constant market. *The update (14) for the constant market maximizes the likelihood (13) by gradient ascent on γ subject to the constraint $\sum_{m=1}^{M} \gamma_m^2 = 1$. The incremental update*

$$\beta_m \leftarrow \beta_m + \beta_m \frac{\eta}{B(\mathbf{x}_i)} \left(\frac{\phi_m^{y_i}(\mathbf{x}_i)}{c_{y_i}(\mathbf{x}_i)} - \sum_{k=1}^{K} \phi_m^k(\mathbf{x}_i) \right). \tag{15}$$

maximizes the likelihood (13) by constrained stochastic gradient ascent.

In the general case of non-constant betting functions, the log-likelihood is

$$L(\gamma) = \sum_{i=1}^{N} \log c_{y_i}(\mathbf{x}_i) = \sum_{i=1}^{N} \log \sum_{m=1}^{M} \gamma_m^2 \phi_m^{y_i}(\mathbf{x}_i, \mathbf{c}(\mathbf{x}_i)) - \sum_{i=1}^{N} \log \sum_{k=1}^{K} \sum_{m=1}^{M} \gamma_m^2 \phi_m^k(\mathbf{x}_i, \mathbf{c}(\mathbf{x}_i)) \tag{16}$$

If we ignore the dependence of $\phi_m^k(\mathbf{x}_i, \mathbf{c}(\mathbf{x}_i))$ on γ in (16), and approximate the gradient as:

$$\frac{\partial L(\gamma)}{\partial \gamma_j} \approx \sum_{i=1}^{N} \left(\frac{\gamma_j \phi_j^{y_i}(\mathbf{x}_i, \mathbf{c}(\mathbf{x}_i))}{\sum_{m=1}^{M} \gamma_m^2 \phi_m^{y_i}(\mathbf{x}_i, \mathbf{c}(\mathbf{x}_i))} - \frac{\gamma_j \sum_{k=1}^{K} \phi_j^k(\mathbf{x}_i, \mathbf{c}(\mathbf{x}_i))}{\sum_{k=1}^{K} \sum_{m=1}^{M} \gamma_m^2 \phi_m^k(\mathbf{x}_i, \mathbf{c}(\mathbf{x}_i))} \right)$$

then the proof of Theorem 5 follows through and we obtain the following market update

$$\beta_m \leftarrow \beta_m + \beta_m \frac{\eta}{B(\mathbf{x}, \mathbf{c})} \left[\frac{\phi_m^y(\mathbf{x}, \mathbf{c})}{c_y} - \sum_{k=1}^{K} \phi_m^k(\mathbf{x}, \mathbf{c}) \right], \quad m = 1, ..., M \tag{17}$$

This way we obtain only an approximate statement in the general case

Remark 6 Maximum Likelihood. *The prediction market update (17) finds an approximate maximum of the likelihood (13) subject to the constraint $\sum_{m=1}^{M} \gamma_m^2 = 1$ by an approximate constrained stochastic gradient ascent.*

11

Observe that the updates from (15) and (17) differ from the update (3) by using an adaptive step size $\eta/B(\mathbf{x}, \mathbf{c})$ instead of the fixed step size 1.

It is easy to check that maximizing the likelihood is equivalent to minimizing an approximation of the expected KL divergence to the true distribution

$$E_\Omega[KL\left(p(y|\mathbf{x}), c_y(\mathbf{x})\right)] = \int_\Omega p(\mathbf{x}) \int_Y p(y|\mathbf{x}) \log \frac{p(y|\mathbf{x})}{c_y(\mathbf{x})} dy d\mathbf{x}$$

obtained using the training set as Monte Carlo samples from $p(\mathbf{x}, y)$.

In many cases the number of negative examples is much larger than the positive examples, and is desired to maximize a weighted log-likelihood

$$L(\gamma) = \frac{1}{N} \sum_{i=1}^{N} w(\mathbf{x}_i) \ln c_{y_i}(\mathbf{x}_i)$$

This can be achieved (exactly for constant betting and approximately in general) using the weighted update rule

$$\beta_m \leftarrow \beta_m + \eta w(\mathbf{x}) \frac{\beta_m}{B(\mathbf{x}, \mathbf{c})} \left[\frac{\phi_m^y(\mathbf{x}, \mathbf{c})}{c_y} - \sum_{k=1}^{K} \phi_m^k(\mathbf{x}, \mathbf{c}) \right], \ m = 1, ..., M \tag{18}$$

The parameter η and the number of training epochs can be used to control how close the budgets β are to the ML optimum, and this way avoid overfitting the training data.

An important issue for the real prediction markets is the *efficient market hypothesis*, which states that the market price fuses in an optimal way the information available to the market participants (Fama, 1970; Basu, 1977; Malkiel, 2003). From Theorem 5 we can draw the following conclusions for the artificial prediction market with constant betting:

1. In general, an untrained market (in which the budgets have not been updated based on training data) will not satisfy the efficient market hypothesis.

2. The market trained with a large amount of representative training data and small η satisfies the efficient market hypothesis.

5. Specialized Classifiers

The prediction market is capable of fusing the information available to the market participants, which can be trained classifiers. These classifiers are usually suboptimal, due to computational or complexity constraints, to the way they are trained, or other reasons.

In boosting, all selected classifiers are aggregated for each instance $\mathbf{x} \in \Omega$. This can be detrimental since some classifiers could perform poorly on subregions of the instance space Ω, degrading the performance of the boosted classifier. In many situations there exist simple rules that hold on subsets of Ω but not on the entire Ω. Classifiers trained on such subsets $D_i \subset \Omega$, would have small misclassification error on D_i but unpredictable behavior outside of D_i. The artificial prediction market can aggregate such classifiers, transformed into participants that don't bet anything outside of their domain of expertise $D_i \subset \Omega$. This way, for different instances $\mathbf{x} \in \Omega$, different subsets of participants will contribute to the

resulting probability estimate. We call these *specialized classifiers* since they only give their opinion through betting on observations that fall inside their domain of specialization.

Thus a specialized classifier with a domain D would have a betting function of the form:

$$\phi^k(\mathbf{x}, \mathbf{c}) = \begin{cases} \varphi^k(\mathbf{x}, \mathbf{c}) \text{ if } \mathbf{x} \in D \\ 0 \text{ else} \end{cases} \tag{19}$$

This idea is illustrated on the following simple 2D example of a triangular region, shown in Figure 4, with positive examples inside the triangle and negatives outside. An accurate classifier for that region can be constructed using six market participants, one for each half-plane determined by each side of the triangle.
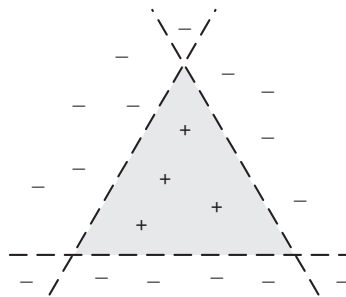

Figure 4: A perfect classifier can be constructed for the triangular region above from a market of six specialized classifiers that only bid on a half-plane determined by one side of the triangle. Three of these specialized classifiers have 100% accuracy while the other three have low accuracy. Nevertheless, the market is capable of obtaining 100% overall accuracy.

Three of these classifiers correspond to the three half planes that are outside the triangle. These participants have 100% accuracy in predicting the observations, all negatives, that fall in their half planes and don't bet anything outside of their half planes. The other three classifiers are not very good, and will have smaller budgets. On an observation that lies outside of the triangle, one or two of the high-budget classifiers will bet a large amount on the correct prediction and will drive the output probability. When an observation falls inside the triangle, only the small-budget classifiers will participate but will be in agreement and still output the correct probability. Evaluating this market on 1000 positives and 1000 negatives showed that the market obtained a prediction accuracy of 100%.

There are many ways to construct specialized classifiers, depending on the problem setup. In natural language processing for example, a specialized classifier could be based on grammar rules, which work very well in many cases, but not always.

We propose two generic sets of specialized classifiers. The first set are the leaves of the random trees of a random forest while the second set are the leaves of the decision trees trained by adaboost. Each leaf f is a rule that defines a domain $D_f = \{\mathbf{x} \in \Omega, f(\mathbf{x}) = 1\}$ of the instances that obey that rule. The betting function of this specialized classifier is given in eq. (19) where $\varphi_f^k(\mathbf{x}, \mathbf{c})$ is based on the associated classifier $h_f^k(\mathbf{x}) = n_{fk}/n_f$, obtaining constant, linear and aggressive versions. Here n_{fk} is the number of training instances of

class k that obey rule f and $n_f = \sum_k n_{fk}$. By the way the random trees are trained, usually $n_f = n_{fk}$ for some k.

In Friedman and Popescu (2008) these rules were combined using a linear aggregation method similar to boosting. One could also use other nodes of the random tree, not necessarily the leaves, for the same purpose.

It can be verified using eq. (8) that constant specialized betting is the linear aggregation of the participants that are currently betting. This is different than the linear aggregation of all the classifiers.

6. Related Work

This work borrows prediction market ideas from Economics and brings them to Machine Learning for supervised aggregation of classifiers or features in general.

Related work in Economics. Recent work in Economics (Manski, 2006; Perols et al., 2009; Plott et al., 2003) investigates the information fusion of the prediction markets. However, none of these works aims at using the prediction markets as a tool for learning class probability estimators in a supervised manner.

Some works (Perols et al., 2009; Plott et al., 2003) focus on parimutuel betting mechanisms for combining classifiers. In parimutuel betting contracts are sold for all possible outcomes (classes) and the entire budget (minus fees) is divided between the participants that purchased contracts for the winning outcome. Parimutuel betting has a different way of fusing information than the Iowa prediction market.

The information based decision fusion (Perols et al., 2009) is a first version of an artificial prediction market. It aggregates classifiers through the parimutuel betting mechanism, using a loop that updates the odds for each outcome and takes updated bets until convergence. This insures a stronger information fusion than without updating the odds. Our work is different in many ways. First our work uses the Iowa electronic market instead of parimutuel betting with odds-updating. Using the Iowa model allowed us to obtain a closed form equation for the market price in some important cases. It also allowed us to relate the market to some existing learning methods. Second, our work presents a multi-class formulation of the prediction markets as opposed to a two-class approach presented in (Perols et al., 2009). Third, the analytical market price formulation allowed us to prove that the constant market performs maximum likelihood learning. Finally, our work evaluates the prediction market not only in terms of classification accuracy but also in the accuracy of predicting the exact class conditional probability given the evidence.

Related work in Machine Learning. Implicit online learning (Kulis and Bartlett, 2010) presents a generic online learning method that balances between a "conservativeness" term that discourages large changes in the model and a "correctness" term that tries to adapt to the new observation. Instead of using a linear approximation as other online methods do, this approach solves an implicit equation for finding the new model. In this regard, the prediction market also solves an implicit equation at each step for finding the new model, but does not balance two criteria like the implicit online learning method. Instead it performs maximum likelihood estimation, which is consistent and asymptotically optimal. In experiments, we observed that the prediction market obtains significantly smaller misclassification errors on many datasets compared to implicit online learning.

Specialization can be viewed as a type of reject rule (Chow, 1970; Tortorella, 2004). However, instead of having a reject rule for the aggregated classifier, each market participant has his own reject rule to decide on what observations to contribute to the aggregation. ROC-based reject rules (Tortorella, 2004) could be found for each market participant and used for defining its domain of specialization. Moreover, the market can give an overall reject rule on hopeless instances that fall outside the specialization domain of all participants. No participant will bet for such an instance and this can be detected as an overall rejection of that instance.

If the overall reject option is not desired, one could avoid having instances for which no classifiers bet by including in the market a set of participants that are all the leaves of a number of random trees. This way, by the design of the random trees, it is guaranteed that each instance will fall into at least one leaf, i.e. participant, hence the instance will not be rejected.

A simplified specialization approach is taken in delegated classifiers (Ferri et al., 2004). A first classifier would decide on the relatively easy instances and would delegate more difficult examples to a second classifier. This approach can be seen as a market with two participants that are not overlapping. The specialization domain of the second participant is defined by the first participant. The market takes a more generic approach where each classifier decides independently on which instances to bet.

The same type of leaves of random trees (i.e. rules) were used by Friedman and Popescu (2008) for linear aggregation. However, our work presents a more generic aggregation method through the prediction market, with linear aggregation as a particular case, and we view the rules as one sort of specialized classifiers that only bid in a subdomain of the feature space.

Our earlier work (Lay and Barbu, 2010) focused only on aggregation of classifiers and did not discuss the connection between the artificial prediction markets and logistic regression, kernel methods and maximum likelihood learning. Moreover, it did not include an experimental comparison with implicit online learning and adaboost.

Two other prediction market mechanisms have been recently proposed in the literature. The first one (Chen and Vaughan, 2010; Chen et al., 2011) has the participants entering the market sequentially. Each participant is paid by an entity called the market maker according to a predefined scoring rule. The second prediction market mechanism is the machine learning market (Storkey, 2011; Storkey et al., 2012), dealing with all participants simultaneously. Each market participant purchases contracts for the possible outcomes to maximize its own utility function. The equilibrium price of the contracts is computed by an optimization procedure. Different utility functions result in different forms of the equilibrium price, such as the mean, median, or geometric mean of the participants' beliefs.

7. Experimental Validation

In this section we present experimental comparisons of the performance of different artificial prediction markets with random forest, adaboost and implicit online learning (Kulis and Bartlett, 2010).

Four artificial prediction markets are evaluated in this section. These markets have the same classifiers, namely the leaves of the trained random trees, but differ either in the betting functions or in the way the budgets are trained as follows:

1. The first market has constant betting and equal budgets for all participants. We proved in Section 3.1 that this is a random forest (Breiman, 2001).

2. The second market has constant betting based on specialized classifiers (the leaves of the random trees), with the budgets initialized with the same values like the market 1 above, but trained using the update equation (15). Thus after training it will be different from market 1.

3. The third market has linear betting functions (1), for which the market price can be computed analytically only for binary classification. The market is initialized with equal budgets and trained using eq. (17).

4. The fourth market has aggressive betting (2) with $\epsilon = 0.01$ and the market price computed using the Mann iteration Algorithm 3. The market is initialized with equal budgets and trained using eq. (17). The value $\epsilon = 0.01$ was chosen for simplicity; a better choice would be to obtain it by cross-validation.

For each dataset, 50 random trees are trained on bootstrap samples of the training data. These trained random trees are used to construct the random forest and the other three markets described above. This way only the aggregation capabilities of the different markets are compared.

The budgets in the markets 2-4 described above are trained on the same training data using the update equation (17) which simplifies to (15) for the constant market.

A C++ implementation of these markets can be found at the following address:
`http://stat.fsu.edu/~abarbu/Research/PredMarket.zip`

7.1 Case Study

We first investigate the behavior of three markets on a dataset in terms of training and test error as well as loss function. For that, we chose the `satimage` dataset from the UCI repository (Blake and Merz, 1998) since it has a supplied test set. The `satimage` dataset has a training set of size 4435 and a test set of size 2000.

The markets investigated are the constant market with both incremental and batch updates, given in eq. (15) and (14) respectively, the linear and aggressive markets with incremental updates given in (17). Observe that the η in eq. (15) is not divided by N (the number of observations) while the η in (14) is divided by N. Thus to obtain the same behavior the η in (15) should be the η from (14) divided by N. We used $\eta = 100/N$ for the incremental update and $\eta = 100$ for the batch update unless otherwise specified.

In Figure 5 are plotted the misclassification errors on the training and test sets and the negative log-likelihood function vs. the number of training epochs, averaged over 10 runs. From Figure 5 one could see that the incremental and batch updates perform similarly in terms of the likelihood function, training and test errors. However, the incremental update is preferred since it is requires less memory and can handle an arbitrarily large amount of training data. The aggressive and constant markets achieve similar values of the negative
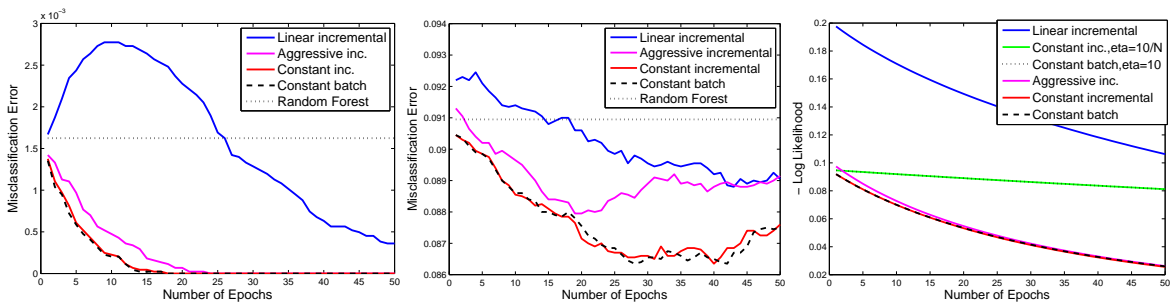
Figure 5: Experiments on the satimage dataset for the incremental and batch market up-
dates. Left: The training error vs. number of epochs. Middle: The test error
vs. number of epochs. Right: The negative log-likelihood function vs. number
of training epochs. The learning rates are $\eta = 100/N$ for the incremental update
and $\eta = 100$ for the batch update unless otherwise specified.

log likelihood and similar training errors, but the aggressive market seems to overfit more
since the test error is larger than the constant incremental (p-value< 0.05). The linear
market has worse values of the log-likelihood, training and test errors (p-value< 0.05).

7.2 Evaluation of the Probability Estimation and Classification Accuracy on Synthetic Data

We perform a series of experiments on synthetic datasets to evaluate the market's ability
to predict class conditional probabilities $P(Y|\mathbf{x})$. The experiments are performed on 5000
binary datasets with 50 levels of Bayes error

$$E = \int \min\{p(\mathbf{x}, Y = 0), p(\mathbf{x}, Y = 1)\}d\mathbf{x},$$

ranging from 0.01 to 0.5 with equal increments. For each dataset, the two classes have
equal frequency. Both $p(\mathbf{x}|Y = k), k = 0, 1$ are normal distributions $\mathcal{N}(\mu_k, \sigma^2 I)$, with
$\mu_0 = 0, \sigma^2 = 1$ and μ_1 chosen in some random direction at such a distance to obtain the
desired Bayes error.

For each of the 50 Bayes error levels, 100 datasets of size 200 were generated using
the bisection method to find an appropriate μ_1 in a random direction. Training of the
participant budgets is done with $\eta = 0.1$.

For each observation $\mathbf{x}$, the class conditional probability can be computed analytically
using the Bayes rule

$$p^*(Y = 1|\mathbf{x}) = \frac{p(\mathbf{x}|Y = 1)p(Y = 1)}{p(\mathbf{x}, Y = 0) + p(\mathbf{x}, Y = 1)}$$

An estimation $\hat{p}(y = 1|\mathbf{x})$ obtained with one of the markets is compared to the true
probability $p^*(Y = 1|\mathbf{x})$ using the L_2 norm

$$E(\hat{p}, p^*) = \int (\hat{p}(y = 1|\mathbf{x}) - p^*(y = 1|\mathbf{x}))^2 p(\mathbf{x})d\mathbf{x}$$

where $p(\mathbf{x}) = p(\mathbf{x}, Y = 0) + p(\mathbf{x}, Y = 1)$.
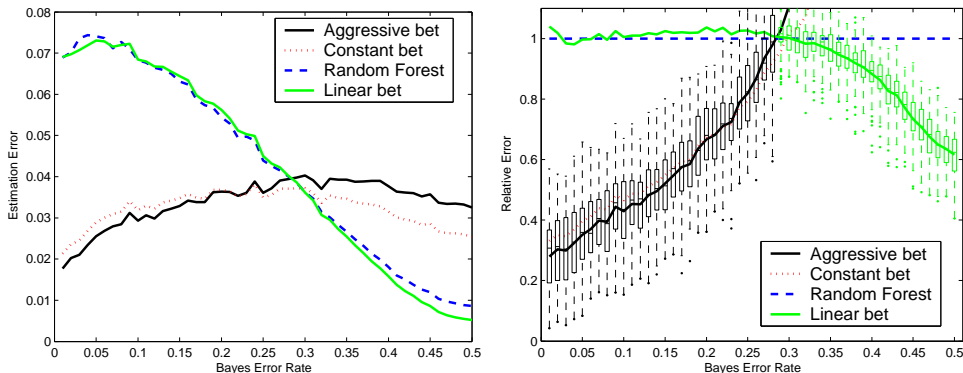
17

Figure 6: Left: Class probability estimation error vs problem difficulty for 5000 100D problems. Right: Probability estimation errors relative to random forest. The aggressive and linear betting are shown with box plots.
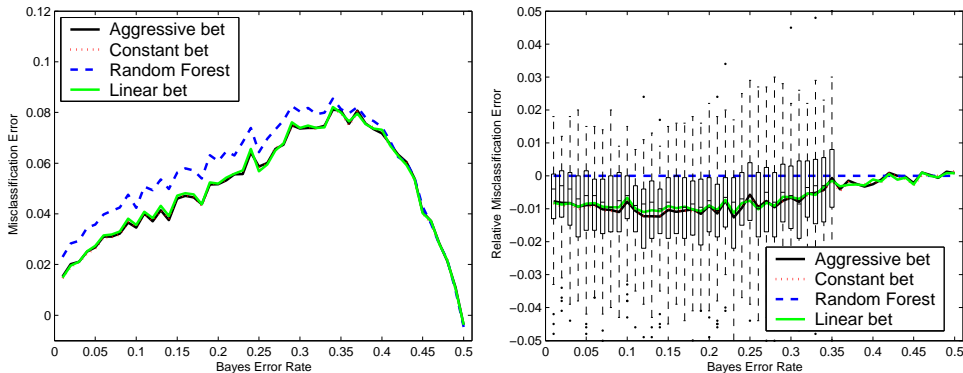

Figure 7: Left: Misclassification error minus Bayes error vs problem difficulty for 5000 100D problems. Right: Misclassification errors relative to random forest. The aggressive betting is shown with box plots.

In practice, this error is approximated using a sample of size 1000. The errors of the probability estimates obtained by the four markets are shown in Figure 6 for a 100D problem setup. Also shown on the right are the errors relative to the random forest, obtained by dividing each error to the corresponding random forest error. As one could see, the aggressive and constant betting markets obtain significantly better (p-value < 0.01) probability estimators than the random forest, for Bayes errors up to 0.28. On the other hand, the linear betting market obtains probability estimators significantly better (p-value < 0.01) than the random forest for Bayes error from 0.34 to 0.5.

We also evaluated the misclassification errors of the four markets in predicting the correct class, for the same 5000 datasets. The difference between these misclassification errors and the Bayes error are shown in Figure 7, left. The difference between these misclassification errors and the random forest error are shown in Figure 7, right. We see that all markets

with trained participants predict significantly better (p-value < 0.01) than random forest for Bayes errors up to 0.3, and behave similar to random forest for the remaining datasets.

7.3 Comparison with Random Forest on UCI Datasets

In this section we conduct an evaluation on 31 datasets from the UCI machine learning repository (Blake and Merz, 1998). The optimal number of training epochs and η are meta-parameters that need to be chosen appropriately for each dataset. We observed experimentally that η can take any value up to a maximum that depends on the dataset. In these experiments we took $\eta = 10/N_{train}$. The best number of epochs was chosen by ten fold cross-validation.

In order to compare with the results in (Breiman, 2001), the training and test sets were randomly subsampled from the available data, with 90% for training and 10% for testing. The exceptions are the `satimage`, `zipcode`, `hill-valley` and `poker`datasets with test sets of size $2000, 2007, 606, 10^6$ respectively. All results were averaged over 100 runs.

We present two random forest results. In the column named RFB are presented the random forest results from (Breiman, 2001)where each tree node is split based on a random feature. In the column named RF we present the results of our own RF implementation with splits based on random features. The leaf nodes of the random trees from our RF implementation are used as specialized participants for all the markets evaluated.

The CB, LB and AB columns are the performances of the constant, linear and respectively aggressive markets on these datasets.

Significant mean differences ($\alpha < 0.01$) from RFB are shown with $+, -$ for when RFB is worse respectively better. Significant paired t-tests (Demšar, 2006) ($\alpha < 0.01$) that compare the markets with our RF implementation are shown with $\bullet, \dagger$ for when RF is worse respectively better.

The constant, linear and aggressive markets significantly outperformed our RF implementation on 22, 19 respectively 22 datasets out of the 31 evaluated. They were not significantly outperformed by our RF implementation on any of the 31 datasets.

Compared to the RF results from Breiman (2001) (RFB), CB, LB and AB significantly outperformed RFB on 6,5,6 datasets respectively, and were not significantly outperformed on any dataset.

7.4 Comparison with Implicit Online Learning on UCI Datasets

We implemented the implicit online learning (Kulis and Bartlett, 2010) algorithm for classification with linear aggregation. The objective of implicit online learning is to minimize the loss $\ell(\beta)$ in a *conservative* way. The *conservativeness* of the update is determined by a Bregman divergence

$$D(\beta, \beta^t) = \phi(\beta) - \phi(\beta^t) - \langle \nabla \phi(\beta^t), \beta - \beta^t \rangle$$

where $\phi(\beta)$ are real-valued strictly convex functions. Rather than minimize the loss function itself, the function

$$f_t(\beta) = D(\beta, \beta^t) + \eta_t \ell(\beta)$$

Table 1: The misclassification errors for 31 datasets from the UC Irvine Repository are shown in percent (%).. The markets evaluated are our implementation of random forest (RF), and markets with Constant (CB), Linear (LB) and respectively Aggressive (AB) Betting. RFB contains the random forest results from (Breiman, 2001).

Data	N_{train}	N_{test}	F	K	RFB	RF	CB	LB	AB
breast-cancer	683	–	9	2	2.7	2.5	2.4	2.4	2.4
sonar	208	–	60	2	18.0	16.6	14.1 ●+	14.2 ●+	14.1 ●+
vowel	990	–	10	11	3.3	2.9	2.6 ●+	2.7 +	2.6 ●+
ecoli	336	–	7	8	13.0	12.9	12.9	12.8	12.9
german	1000	–	24	2	26.2	25.5	24.9 ●+	25.1	24.9 ●+
glass	214	–	9	6	21.2	23.5	22.2 ●	22.4	22.2 ●
image	2310	–	19	7	2.7	2.7	2.5 ●	2.5 ●	2.5 ●
ionosphere	351	–	34	2	7.5	7.4	6.7 ●	6.9 ●	6.7 ●
letter-recognition	20000	–	16	26	4.7	4.2 +	4.2 ●+	4.2 ●+	4.2 ●+
liver-disorders	345	–	6	2	24.7	26.5	26.3	26.2	26.2
pima-diabetes	768	–	8	2	24.3	24.1	23.8	23.7	23.8
satimage	4435	2000	36	6	10.5	10.1 +	10.0 ●+	10.1 ●+	10.0 ●+
vehicle	846	–	18	4	26.4	26.3	26.1	26.2	26.1
voting-records	232	–	16	2	4.6	5.3	4.2 ●	4.2 ●	4.2 ●
zipcode	7291	2007	256	10	7.8	7.7	7.6 ●+	7.7 ●+	7.6 ●+
abalone	4177	–	8	3	–	45.5	45.4	45.4	45.4
balance-scale	625	–	4	3	–	15.4	15.4	15.4	15.4
car	1728	–	6	4	–	2.8	2.0 ●	2.2 ●	2.0 ●
connect-4	67557	–	42	3	–	19.6	19.3 ●	19.4 ●	19.5 ●
cylinder-bands	277	–	33	2	–	22.7	20.9 ●	21.1 ●	20.9 ●
hill-valley	606	606	100	2	–	46.9	45.8 ●	46.3 ●	45.8 ●
isolet	1559	–	617	26	–	17.0	15.7 ●	15.8 ●	15.7 ●
king-rook-vs-king	28056	–	6	18	–	15.6	15.4 ●	15.4 ●	15.4 ●
king-rk-vs-k-pawn	3196	–	36	2	–	2.0	1.5 ●	1.6 ●	1.5 ●
madelon	2000	–	500	2	–	46.1	45.2 ●	45.3 ●	45.2 ●
magic	19020	–	10	2	–	12.0	11.9 ●	11.9 ●	11.9 ●
musk	6598	–	166	2	–	3.7	3.5 ●	3.6 ●	3.5 ●
poker	25010	10^6	10	10	–	43.2	43.1 ●	43.1 ●	43.1 ●
SAheart	462	–	9	2	–	30.8	30.8	30.7	30.8
splice-junction	3190	–	59	3	–	18.9	17.7 ●	18.2 ●	17.7 ●
yeast	1484	–	8	10	–	38.3	38.1	38.0	38.1

is minimized instead. Here η_t is the learning rate. The Bregman divergence ensures that the optimal β is not *too far* from β^t. The algorithm for implicit online learning is as follows

$$\tilde{\beta}^{t+1} = \underset{\beta \in \mathbb{R}^M}{\arg\min}\, f_t(\beta)$$

$$\beta^{t+1} = \underset{\beta \in S}{\arg\min}\, D(\beta, \tilde{\beta}^{t+1})$$

The first step solves the unconstrained version of the problem while the second step finds the *nearest* feasible solution to the unconstrained minimizer subject to the Bregman divergence.

For our problem we use

$$\ell(\beta) = -\log(c_y(\beta))$$

where $c_y(\beta)$ is the constant market equilibrium price for ground truth label y. We chose the squared Euclidean distance $D(\beta, \beta^t) = \|\beta - \beta^t\|_2^2$ as our Bregman divergence and learning rate $\eta_t = 1/\sqrt{t}$. To ensure that $c = \sum_{m=1}^M h_m \beta_m = H\beta$ is a valid probability vector, the feasible solution set is therefore $S = \{\beta \in [0,1]^M : \sum_{m=1}^M \beta_m = 1\}$. This gives the following update scheme

$$\tilde{\beta}^{t+1} = \beta^t + \eta_t \frac{1}{p}(H^y)^T$$

$$\beta^{t+1} = \underset{\beta \in S}{\arg\min}\left\{\|\beta - \tilde{\beta}^{t+1}\|_2^2\right\}$$

where $H^y = \left(h_1^y,\ h_2^y,\ \ldots,\ h_M^y\right)$ is the vector of classifier outputs for the true label y, $q = H^y \beta^t$, $r = H^y (H^y)^T$ and $p = \frac{1}{2}\left(q + \sqrt{q^2 + 4\eta_t r}\right)$.

The results presented in Table 2 are obtained by 10 fold cross-validation. The cross-validation errors were averaged over 10 different permutations of the data in the cross-validation folds.

The results from CB online and implicit online are obtained in one epoch. The results from the CB offline and implicit offline columns are obtained in an off-line fashion using an appropriate number of epochs (up to 10) to obtain the smallest cross-validated error on a random permutation of the data that is different from the 10 permutations used to obtain the results.

The comparisons are done with paired t-tests and shown with $*$ and $\ddagger$ when the constant betting market is significantly ($\alpha < 0.01$) better or worse than the corresponding implicit online learning. We also performed a comparison with our RF implementation, and significant differences are shown with $\bullet$ and $\dagger$.

Compared to RF, implicit online learning won 5-0, CB online won in 9-1 and CB offline won 12-0.

Compared to implicit online, which performed identical with implicit offline, both CB online and CB offline won 9-0.

7.5 Comparison with Adaboost for Lymph Node Detection

Finally, we compared the linear aggregation capability of the artificial prediction market with adaboost for a lymph node detection problem. The system is setup as described in

Table 2: Comparison with Implicit Online Learning and random forest using 10-fold cross-validation.

Dataset	N_{train}	N_{test}	F	K	RF	Implicit Online	CB Online	Implicit Offline	CB Offline
breast-cancer	683	–	9	2	3.1	3.1	3	3.1	3
sonar	208	–	60	2	15.1	15.2	15.3	15.1	14.6
vowel	990	–	10	11	3.2	3.2	3.2	3.2	2.9 ●∗
ecoli	336	–	7	8	13.7	13.7	13.6	13.7	13.6
german	1000	–	24	2	23.6	23.5	23.5	23.5	23.4
glass	214	–	9	6	21.4	21.4	21.3	21.4	21
image	2310	–	19	7	1.9	1.9	1.9	1.9	1.8 ●
ionosphere	351	–	34	2	6.4	6.5	6.5	6.5	6.5
letter-recognition	20000	–	16	26	3.3	3.3	3.3 ●∗	3.3	3.3
liver-disorders	345	–	6	2	26.4	26.4	26.4	26.4	26.4
pima-diabetes	768	–	8	2	23.2	23.2	23.2	23.2	23.2
satimage	4435	2000	36	6	8.8	8.8	8.8	8.8	8.7 ●
vehicle	846	–	18	4	24.8	24.7	24.9	24.7	24.9
voting-records	232	–	16	2	3.5	3.5	3.5	3.5	3.5
zipcode	7291	2007	256	10	6.1	6.1	6.2	6.1	6.2
abalone	4177	–	8	3	45.5	45.5	45.6 †	45.5	45.5
balance-scale	625	–	4	3	17.7	17.7	17.7	17.7	17.7
car	1728	–	6	4	2.3	2.3	1.8 ●∗	2.3	1.1 ●∗
connect-4	67557	–	42	3	19.9	19.9 ●	19.5 ●∗	19.9 ●	18.2 ●∗
cylinder-bands	277	–	33	2	21.4	21.3	21.2	21.3	20.8 ●
hill-valley	606	606	100	2	43.8	43.7	43.7	43.7	43.7
isolet	1559	–	617	26	6.9	6.9	6.9	6.9	6.9
king-rk-vs-king	28056	–	6	18	21.6	21.6 ●	19.6 ●∗	21.5 ●	15.7 ●∗
king-rk-vs-k-pawn	3196	–	36	2	1	1	0.7 ●∗	1	0.5 ●∗
magic	19020	–	10	2	11.9	11.9 ●	11.8 ●∗	11.9 ●	11.7 ●∗
madelon	2000	–	500	2	26.8	26.5 ●	25.6 ●∗	26.4 ●	21.6 ●∗
musk	6598	–	166	2	1.7	1.7 ●	1.6 ●∗	1.7 ●	1 ●∗
splice-junction-gene	3190	–	59	3	4.3	4.3	4.2 ●∗	4.3	4.1 ●∗
SAheart	462	–	9	2	31.5	31.5	31.6	31.5	31.6
yeast	1484	–	8	10	37.3	37.3	37.3	37.3	37.3

Barbu et al. (2012), namely a set of lymph node candidate positions (x, y, z) are obtained using a trained detector. Each candidate is segmented using gradient descent optimization and about 17000 features are extracted from the segmentation result. Using these features, adaboost constructed 32 weak classifiers. Each weak classifier is associated with one feature, splits the feature range into 64 bins and returns a predefined value (1 or -1), for each bin.

Thus, one can consider there are $M = 32 \times 64 = 2048$ specialized participants, each betting for one class (1 or -1) for any observation that falls in its domain. The participants are given budgets $\beta_{ij}, i = 1, .., 32, j = 1, .., 64$ where i is the feature index and j is the bin index. The participant budgets $\beta_{ij}, j = 1, ..., 64$ corresponding to the same feature i are initialized the same value β_i, namely the adaboost coefficient. For each bin, the return class 1 or -1 is the outcome for which the participant will bet its budget.

The constant betting market of the 2048 participants is initialized with these budgets and trained with the same training examples that were used to train the adaboost classifier.

The obtained constant market probability for an observation $\mathbf{x} = (x_1, ..., x_{32})$ is based on the bin indexes $\mathbf{b} = (b_1(x_1), ..., b_{32}(x_{32}))$:

$$p(y = 1|\mathbf{b}) = \frac{\sum_{i=1}^{32} \beta_{i,b_i} h_i(b_i)}{\sum_{i=1}^{32} \beta_{i,b_i}} \tag{20}$$

An important issue is that the number N_{pos} of positive examples is much smaller than the number N_{neg} of negatives. Similar to adaboost, the sum of the weights of the positive examples should be the same as the sum of weights of the negatives. To accomplish this in the market, we use the weighted update rule Eq. (18), with $w_{pos} = \frac{1}{N_{pos}}$ for each positive example and $w_{neg} = \frac{1}{N_{neg}}$ for each negative.
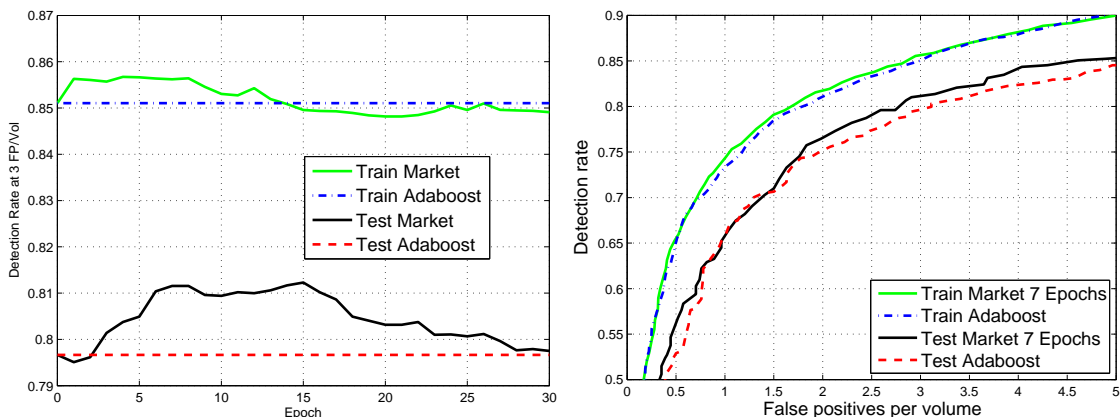

Figure 8: Left: Detection rate at 3 FP/vol vs. number of training epochs for a lymph node detection problem. Right: ROC curves for adaboost and the constant betting market with participants as the 2048 adaboost weak classifier bins. The results are obtained with six-fold cross-validation.

The adaboost classifier and the constant market were evaluated for a lymph node detection application on a dataset containing 54 CT scans of the pelvic and abdominal region, with a total of 569 lymph nodes, with six-fold cross-validation. The evaluation criterion

is the same for all methods, as specified in Barbu et al. (2012). A lymph node detection is considered correct if its center is inside a manual solid lymph node segmentation and is incorrect if it not inside any lymph node segmentation (solid or non-solid).

In Figure 8, left, is shown the training and testing detection rate at 3 false positives per volume (a clinically acceptable false positive rate) vs the number of training epochs. We see the detection rate increases to about 81% for epochs 6 to 16 epochs and then gradually decreases. In Figure 8, right, are shown the training and test ROC curves of adaboost and the constant market trained with 7 epochs. In this case the detection rate at 3 false positives per volume improved from 79.6% for adaboost to 81.2% for the constant market. The p-value for this difference was 0.0276 based on paired t-test.

8. Conclusion and Future Work

This paper presents a theory for artificial prediction markets for the purpose of supervised learning of class conditional probability estimators. The artificial prediction market is a novel online learning algorithm that can be easily implemented for two class and multi class applications. Linear aggregation, logistic regression as well as certain kernel methods can be viewed as particular instances of the artificial prediction markets. Inspired from real life, specialized classifiers that only bet on subsets of the instance space Ω were introduced. Experimental comparisons on real and synthetic data show that the prediction market usually outperforms random forest, adaboost and implicit online learning in prediction accuracy.

The artificial prediction market shows the following promising features:

1. It can be *updated online* with minimal computational cost when a new observation $(\mathbf{x}, y)$ is presented.

2. It has a simple form of the update iteration that can be easily implemented.

3. For multi-class classification it can *fuse information* from all types of binary or multi-class classifiers: e.g. trained *one-vs-all, many-vs-many*, multi-class decision tree, etc.

4. It can obtain meaningful probability estimates when only a subset of the market participants are involved for a particular instance $\mathbf{x} \in X$. This feature is useful for learning on manifolds (Belkin and Niyogi, 2004; Elgammal and Lee; Saul and Roweis, 2003), where the location on the manifold decides which market participants should be involved. For example, in face detection, different face part classifiers (eyes, mouth, ears, nose, hair, etc) can be involved in the market, depending on the orientation of the head hypothesis being evaluated.

5. Because of their betting functions, the *specialized* market participants can decide for which instances they bet and how much. This is another way to combine classifiers, different from the boosting approach where all classifiers participate in estimating the class probability for each observation.

We are currently extending the artificial prediction market framework to regression and density estimation. These extensions involve contracts for uncountably many outcomes but the update and the market price equations extend naturally.

Future work includes finding explicit bounds for the generalization error based on the number of training examples. Another item of future work is finding other generic types specialized participants that are not leaves of random or adaboost trees. For example, by clustering the instances $\mathbf{x} \in \Omega$, one could find regions of the instance space Ω where simple classifiers (e.g. logistic regression, or betting for a single class) can be used as specialized market participants for that region.

Acknowledgments

References

K. J. Arrow, R. Forsythe, M. Gorham, R. Hahn, R. Hanson, J. O. Ledyard, S. Levmore, R. Litan, P. Milgrom, and F. D. Nelson. The promise of prediction markets. *Science*, 320(5878):877, 2008.

A. Barbu, M. Suehling, X. Xu, D. Liu, S. Zhou, and D. Comaniciu. Automatic detection and segmentation of lymph nodes from ct data. *IEEE Trans. on Medical Imaging*, 31(2):240–250, 2012.

S. Basu. Investment performance of common stocks in relation to their price-earnings ratios: A test of the efficient market hypothesis. *The Journal of Finance*, 32(3):663–682, 1977.

M. Belkin and P. Niyogi. Semi-supervised learning on Riemannian manifolds. *Machine Learning*, 56(1):209–239, 2004.

A.L. Berger, V.J.D. Pietra, and S.A.D. Pietra. A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):39–71, 1996.

C. Blake and CJ Merz. UCI repository of machine learning databases [http://www. ics. uci. edu/ mlearn/MLRepository. html], Department of Information and Computer Science. *University of California, Irvine, CA*, 1998.

A. Bordes, S. Ertekin, J. Weston, and L. Bottou. Fast kernel classifiers with online and active learning. *The Journal of Machine Learning Research*, 6:1619, 2005.

L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

G. Cauwenberghs and T. Poggio. Incremental and decremental support vector machine learning. In *NIPS*, page 409, 2001.

Y. Chen and J.W. Vaughan. A new understanding of prediction markets via no-regret learning. In *Proceedings of the 11th ACM conference on Electronic commerce*, pages 189–198. ACM, 2010.

Y. Chen, J. Abernethy, and J.W. Vaughan. An optimization-based framework for automated market-making. *Proceedings of the EC*, 11:5–9, 2011.

C. Chow. On optimum recognition error and reject tradeoff. *IEEE Trans. on Information Theory*, 16(1):41–46, 1970.

B. Cowgill, J. Wolfers, and E. Zitzewitz. Using prediction markets to track information flows: Evidence from Google. *Dartmouth College*, 2008.

J. Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:30, 2006.

A. Elgammal and C.S. Lee. Inferring 3d body pose from silhouettes using activity manifold learning. In *CVPR 2004*.

E.F. Fama. Efficient capital markets: A review of theory and empirical work. *Journal of Finance*, pages 383–417, 1970.

C. Ferri, P. Flach, and J. Hernández-Orallo. Delegating classifiers. In *International Conference in Machine Learning*, 2004.

Y. Freund and R.E. Schapire. Experiments with a new boosting algorithm. In *International Conference in Machine Learning*, pages 148–156, 1996.

J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28(2):337–407, 2000.

J.H. Friedman and B.E. Popescu. Predictive learning via rule ensembles. *Ann. Appl. Stat.*, 2(3): 916–954, 2008.

S. Gjerstad and M.C. Hall. Risk aversion, beliefs, and prediction market equilibrium. *Economic Science Laboratory, University of Arizona*, 2005.

J. Kivinen, AJ Smola, and RC Williamson. Online Learning with Kernels. *IEEE Trans. on Signal Processing*, 52:2165–2176, 2004.

B. Kulis and P.L. Bartlett. Implicit Online Learning. In *International Conference in Machine Learning*, 2010.

N. Lay and A. Barbu. Supervised Aggregation of Classifiers using Artificial Prediction Markets. In *International Conference in Machine Learning*, 2010.

B.G. Malkiel. The efficient market hypothesis and its critics. *The Journal of Economic Perspectives*, 17(1):59–82, 2003.

W. Robert Mann. Mean Value Methods in Iteration. *Proc. Amer. Math. Soc.*, 4:506–510, 1953.

C.F. Manski. Interpreting the predictions of prediction markets. *Economics Letters*, 91(3):425–429, 2006.

J. Perols, K. Chari, and M. Agrawal. Information Market-Based Decision Fusion. *Management Science*, 55(5):827–842, 2009.

C.R. Plott, J. Wit, and W.C. Yang. Parimutuel betting markets as information aggregation devices: Experimental results. *Economic Theory*, 22(2):311–351, 2003.

P.M. Polgreen, F.D. Nelson, and G.R. Neumann. Use of prediction markets to forecast infectious disease activity. *Clinical Infectious Diseases*, 44(2):272–279, 2006.

C. Polk, R. Hanson, J. Ledyard, and T. Ishikida. The policy analysis market: an electronic commerce application of a combinatorial information market. In *ACM Conf. on Electronic Commerce*, pages 272–273, 2003.

A. Ratnaparkhi et al. A maximum entropy model for part-of-speech tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, volume 1, pages 133–142, 1996.

L.K. Saul and S.T. Roweis. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *The Journal of Machine Learning Research*, 4:119–155, 2003.

R.E. Schapire. The boosting approach to machine learning: An overview. *Lect. Notes in Statistics*, pages 149–172, 2003.

A. Storkey. Machine learning markets. *AISTATS*, 2011.

A. Storkey, J. Millin, and K. Geras. Isoelastic agents and wealth updates in machine learning markets. *ICML*, 2012.

F. Tortorella. Reducing the classification cost of support vector classifiers through an ROC-based reject rule. *Pattern Analysis & Applications*, 7(2):128–143, 2004.

J. Wolfers and E. Zitzewitz. Prediction markets. *Journal of Economic Perspectives*, pages 107–126, 2004.

S.C. Zhu, Y. Wu, and D. Mumford. Filters, random fields and maximum entropy (FRAME): Towards a unified theory for texture modeling. *International Journal of Computer Vision*, 27(2):107–126, 1998.

Appendix: Proofs

Proof [of Theorem 1] From eq. (3), the total budget $\sum_{m=1}^{M} \beta_m$ is conserved if and only if

$$\sum_{m=1}^{M}\sum_{k=1}^{K}\beta_m\phi_m^k(\mathbf{x},\mathbf{c}) = \sum_{m=1}^{M}\beta_m\phi_m^y(\mathbf{x},\mathbf{c})/c_y \tag{21}$$

Denoting $n = \sum_{m=1}^{M}\sum_{k=1}^{K}\beta_m\phi_m^k(\mathbf{x},\mathbf{c})$, and since the above equation must hold for all y, we obtain that eq. (4) is a necessary condition and also $c_k \neq 0, k = 1,...,K$, which means $c_k > 0, k = 1,...,K$. Reciprocally, if $c_k > 0$ and eq. (4) hold for all k, dividing by c_k we obtain eq. (21).

∎

Proof [of Remark 2] Since the total budget is conserved and is positive, there exists a $\beta_m > 0$, therefore $\sum_{m=1}^{M}\beta_m\phi_m^k(\mathbf{x},0) > 0$, which implies $\lim_{c_k\to 0} f_k(c_k) = \infty$. From the fact that $f_k(c_k)$ is continuous and strictly decreasing, with $\lim_{c_k\to 0} f_k(c_k) = \infty$ and $\lim_{c_k\to 1} f_k(c_k) = 0$, it implies that for every $n > 0$ there exists a unique c_k that satisfies $f_k(c_k) = n$.

∎

Proof [of Theorem 3] From Remark 2 we get that for every $n \geq n_k, n > 0$ there is a unique $c_k(n)$ such that $f_k(c_k(n)) = n$. Moreover, following the proof of Remark 2 we see that $c_k(n)$ is continuous and strictly decreasing on (n_k, ∞), with $\lim_{n\to\infty} c_k(n) = 0$.

If $\max_k n_k > 0$, take $n^* = \max_k n_k$. There exists $k \in \{1,...,K\}$ such that $n_k = n^*$, so $c_k(n^*) = 1$, therefore $\sum_{j=1}^{K} c_j(n^*) \geq 1$.

If $\max_k n_k = 0$ then $n_k = 0, k = 1,...,K$ which means $\phi_m^k(\mathbf{x},1) = 0, k = 1,...,K$ for all m with $\beta_m > 0$. Let $a_m^k = \min\{c|\phi_m^k(\mathbf{x},c) = 0\}$. We have $a_m^k > 0$ for all k since

$\phi_m^k(\mathbf{x}, 0) > 0$. Thus $\lim_{n \to 0_+} c_k(n) = \max_m a_m^k \geq a_1^k$, where we assumed that $\phi_1(\mathbf{x}, \mathbf{c})$ satisfies Assumption 2. But from Assumption 2 there exists k such that $a_1^k = 1$. Thus $\lim_{n \to 0_+} \sum_{k=1}^K c_k(n) \geq \sum_{k=1}^K a_1^k > 1$ so there exists n^* such that $\sum_{k=1}^K c_k(n^*) \geq 1$.

Either way, since $\sum_{k=1}^K c_k(n)$ is continuous, strictly decreasing, and since $\sum_{k=1}^K c_k(n^*) \geq 1$ and $\lim_{n \to \infty} \sum_{k=1}^K c_k(n) = 0$, there exists a unique $n > 0$ such that $\sum_{k=1}^K c_k(n) = 1$. For this n, from Theorem 1 follows that the total budget is conserved for the price $\mathbf{c} = (c_1(n), ..., c_K(n))$. Uniqueness follows from the uniqueness of $c_k(n)$ and the uniqueness of n. ∎

Proof [of Theorem 4] The price equations (4) become:

$$\sum_{m=1}^M \beta_m \phi_m^k(\mathbf{x}) = c_k \sum_{k=1}^K \sum_{m=1}^M \beta_m \phi_m^k(\mathbf{x}), \quad \forall k = 1, ..., K.$$

which give the result from eq. (8).

If $\phi_m^k(\mathbf{x}) = \eta h_m^k(\mathbf{x})$, using $\sum_{k=1}^K h_m^k(\mathbf{x}) = 1$, the denominator of eq. (8) becomes

$$\sum_{k=1}^K \sum_{m=1}^M \beta_m \phi_m^k(\mathbf{x}) = \eta \sum_{m=1}^M \beta_m \sum_{k=1}^K h_m^k(\mathbf{x}) = \eta \sum_{m=1}^M \beta_m$$

so

$$c_k = \frac{\eta \sum_{m=1}^M \beta_m h_m^k(\mathbf{x})}{\eta \sum_{m=1}^M \beta_m} = \sum_m \alpha_m h_m^k(\mathbf{x}), \quad \forall k = 1, ..., K$$

∎

Proof [of Theorem 5] For the current parameters $\gamma = (\gamma_1, ..., \gamma_M) = (\sqrt{\beta_1}, ..., \sqrt{\beta_m})$ and an observation $(\mathbf{x}_i, y_i)$, we have the market price for label y_i:

$$c_{y_i}(\mathbf{x}_i) = \sum_{m=1}^M \gamma_m^2 \phi_m^{y_i}(\mathbf{x}_i) / (\sum_{m=1}^M \sum_{k=1}^K \gamma_m^2 \phi_m^k(\mathbf{x}_i)) \tag{22}$$

So the log-likelihood is

$$L(\gamma) = \frac{1}{N} \sum_{i=1}^N \log c_{y_i}(\mathbf{x}_i) = \frac{1}{N} \sum_{i=1}^N \log \sum_{m=1}^M \gamma_m^2 \phi_m^{y_i}(\mathbf{x}_i) - \frac{1}{N} \sum_{i=1}^N \log \sum_{m=1}^M \sum_{k=1}^K \gamma_m^2 \phi_m^k(\mathbf{x}_i) \tag{23}$$

We obtain the gradient components:

$$\frac{\partial L(\gamma)}{\partial \gamma_j} = \frac{1}{N} \sum_{i=1}^N \left(\frac{\gamma_j \phi_j^{y_i}(\mathbf{x}_i)}{\sum_{m=1}^M \gamma_m^2 \phi_m^{y_i}(\mathbf{x}_i)} - \frac{\gamma_j \sum_{k=1}^K \phi_j^k(\mathbf{x}_i)}{\sum_{m=1}^M \sum_{k=1}^K \gamma_m^2 \phi_m^k(\mathbf{x}_i)} \right) \tag{24}$$

Then from (22) we have $\sum_{m=1}^M \gamma_m^2 \phi_m^{y_i}(\mathbf{x}_i) = B(\mathbf{x}_i) c_{y_i}(\mathbf{x}_i)$. Hence (24) becomes

$$\frac{\partial L(\gamma)}{\partial \gamma_j} = \frac{\gamma_j}{N} \sum_{i=1}^N \frac{1}{B(\mathbf{x}_i)} \left(\frac{\phi_j^{y_i}(\mathbf{x}_i)}{c_{y_i}(\mathbf{x}_i)} - \sum_{k=1}^K \phi_j^k(\mathbf{x}_i) \right).$$

Write $u_j = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{B(\mathbf{x}_i)} \left(\frac{\phi_j^{y_i}(\mathbf{x}_i)}{c_{y_i}(\mathbf{x}_i)} - \sum_{k=1}^{K} \phi_j^k(\mathbf{x}_i) \right)$, then $\frac{\partial L(\gamma)}{\partial \gamma_j} = \gamma_j u_j$. The batch update (14) is $\beta_j \leftarrow \beta_j + \eta \beta_j u_j$. By taking the square root we get the update in γ

$$\gamma_j \leftarrow \gamma_j \sqrt{1 + \eta u_j} = \gamma_j + \gamma_j(\sqrt{1 + \eta u_j} - 1) = \gamma_j + \gamma_j \frac{\eta u_j}{\sqrt{1 + \eta u_j} + 1} = \gamma_j'.$$

We can write the Taylor expansion:

$$L(\gamma') = L(\gamma) + (\gamma' - \gamma)^T \nabla L(\gamma) + \frac{1}{2}(\gamma' - \gamma)^T H(L)(\zeta)(\gamma' - \gamma)$$

so

$$L(\gamma') = L(\gamma) + \sum_{j=1}^{M} \gamma_j u_j \frac{\eta \gamma_j u_j}{\sqrt{1 + \eta u_j} + 1} + \eta^2 A(\eta) = L(\gamma) + \eta \sum_{j=1}^{M} \frac{\gamma_j^2 u_j^2}{\sqrt{1 + \eta u_j} + 1} + \eta^2 A(\eta)$$

where $|A(\eta)|$ is bounded in a neighborhood of 0.

Now assume that $\nabla L(\gamma) \neq 0$, thus $\gamma_j u_j \neq 0$ for some j. Then $\sum_{j=1}^{M} \frac{\gamma_j^2 u_j^2}{\sqrt{1 + \eta u_j} + 1} > 0$ hence $L(\gamma') > L(\gamma)$ for any η small enough.

Thus as long as $\nabla L(\gamma) \neq 0$ the batch update (14) with any η sufficiently small will increase the likelihood function.

The batch update (14) can be split into N per-observation updates of the form (15). ∎