Supervised Aggregation Using Artificial Prediction Markets

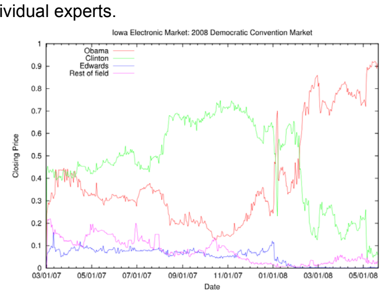
Nathan Lay

Advisor: Adrian Barbu, Department of Statistics

Co-Advisor: Anke Meyer-Baese, Department of Scientific Computing

Prediction Markets

- Forum where contracts are traded on future outcomes.
- Contracts pay contingent on the outcome.
- Trading price of contracts reflects combined knowledge and experience of participants.
- Trading price is an estimator of the probability.
- Can predict outcomes of elections, sporting events, and foreign affairs.
- Were demonstrated to be more accurate than polling or individual experts.

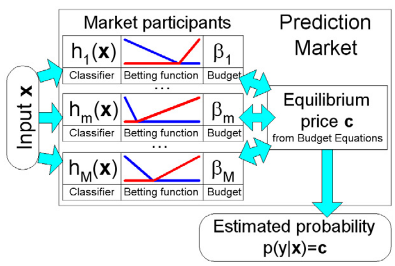


Trading prices of contracts on democratic nominees for the 2008 presidential election.

Overview

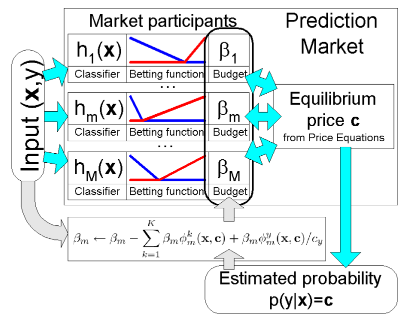
Idea

- Reinterpret events as *instances*, future outcomes as instance *labels*, and participants as *classifiers*, *regressors* or *densities*.
- For each instance, classifiers "purchase" contracts for each possible label.
- The trading price is a probability estimate for the instance.



Learning

- Each participant is allotted a budget.
- Each participant bids for contracts and are rewarded based on *correct* prediction.
- Budgets describe the prediction accuracy of each participant.
- The goal is to learn the budget configuration that improves the market's prediction accuracy.



Regression

Overview

- *Events* are instances, and the *outcomes* are real numbers
- Like classification, but with uncountably many *labels*
- Participants are conditional densities h(y|**x**)

Equilibrium

- Equilibrium price conserves the budget sum for each update
- Estimates the true conditional density p(y|**x**)

$$c(y|\mathbf{x};\boldsymbol{\beta}) = \sum_{m=1}^{M} \beta_m h_m(y|\mathbf{x})$$

Update Rule

- Sequential update for each instance **x** and label *y*
- Introduce reward kernel $K(t;y)$ to distribute winnings around *y*

$$\beta_m \leftarrow \beta_m + \eta\beta_m\left(\int_Y K(t;y)\frac{h_m(t|\mathbf{x})}{c(t|\mathbf{x};\boldsymbol{\beta})}dt - 1\right)$$

Delta Update

- When $K(t;y) = \delta(t-y)$
- Analogous update as constant classification market.
- Prone to overfitting.

$$\beta_m \leftarrow \beta_m + \eta\beta_m\left(\frac{h_m(y|\mathbf{x})}{c(y|\mathbf{x})} - 1\right)$$

Gaussian Update

- When

$$K(t;y) = \frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(t-y)^2}{2\sigma^2}}$$

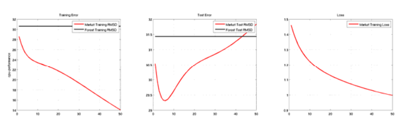
- Update defined in terms of an integral.
- Can be estimated with Hermite-Gauss quadrature.

$$\beta_m \leftarrow \beta_m + \eta\beta_m\left(-1 + \frac{1}{\sqrt{\pi}}\sum_{i=1}^{n}\omega_i\frac{h_m(y+\sqrt{2}\sigma t_i|\mathbf{x})}{c(y+\sqrt{2}\sigma t_i|\mathbf{x})}\right)$$

Loss Function

- The update rule maximizes the average log likelihood

$$\ell(\boldsymbol{\beta}) = -\frac{1}{N}\sum_{n=1}^{N}\log(c(y_n|\mathbf{x}_n;\boldsymbol{\beta}))$$



Training error, test error, and negative log likelihood for the cpu-performance data set.

Results

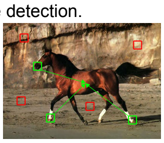
- Real data sets are from UCI and LIAAD repository. There are 23 total.
- Participants are regression tree branches from a regression forest.

Data	N_{train}	N_{test}	F	Y	RFB	RF	DM	GM
abalone	4177	–	8	[1.00, 29.00]	4.600	4.571	4.571	4.571
friedman1	200	2000	10	[4.30, 26.03]	5.700	4.343+	4.335•+	4.193•+
friedman2	200	2000	4	[−167.99, 1633.87]	19600.0	19431.852	19232.482•	18369.546•+
friedman3	200	2000	4	[0.13, 1.73]	0.022	0.028−	0.028•−	0.026•−
housing	506	–	13	[5.00, 50.00]	10.200	10.471	10.130•	10.128•
ozone	330	–	8	[1.00, 38.00]	16.300	16.916	16.925	16.917
servo	167	–	4	[0.13, 7.10]	0.246	0.336	0.295	0.322
ailerons	7154	6596	40	[−0.00, −0.00]	–	2.814e-008	2.814e-008•	2.814e-008•
auto-mpg	392	–	7	[9.00, 46.60]	–	6.469	6.444	6.405•
auto-price	159	–	15	[5118.00, 35056.00]	–	3823550.43	3723413.430	3815863.98
bank	4500	3693	32	[0.00, 0.67]	–	7.238e-003	7.212e-003•	7.210e-003•
breast cancer	194	–	32	[1.00, 125.00]	–	1112.270	1112.509	1108.325
cartexample	40768	–	10	[−12.69, 12.20]	–	1.233	1.233†	1.232•
computeractivity	8192	–	21	[0.00, 99.00]	–	5.414	5.398•	5.414†
diabetes	43	–	2	[3.00, 6.60]	–	0.415	0.426†	0.415
elevators	8752	7847	18	[0.01, 0.08]	–	9.319e-006	9.288e-006•	9.225e-006•
forestfires	517	–	12	[0.00, 1090.84]	–	5834.819	5844.493†	5680.131•
kinematics	8192	–	8	[0.04, 1.46]	–	0.013	0.013•	0.013•
machine	209	–	6	[6.00, 1150.00]	–	3154.521	2991.798•	3042.336
poletelecomm	5000	10000	48	[0.00, 100.00]	–	29.813	28.855•	29.863†
pumadyn	4499	3693	32	[−0.09, 0.09]	–	9.237e-005	8.917e-005•	8.888e-005•
pyrimidines	74	–	27	[0.10, 0.90]	–	0.013	0.013	0.012
triazines	186	–	60	[0.10, 0.90]	–	0.015	0.015	0.015

Table of MSE for forests and markets on UCI and LIAAD data sets. The *F* column is the number of inputs, *Y* is the range of regression, RFB is Breiman's reported error, RF is our forest implementation, DM is the Market with delta updates, and GM is the Market with Gaussian updates. Bullets/daggers represent pairwise significantly better/worse than RF while +/- represent significantly better/worse than RFB.

Hough Forest

- Predict the location of the center of an object.
- Predict based on parts.
- Hybrid of a regression and classification forest
- aggregate Hough Forest branches with the Regression Market to improve detection.



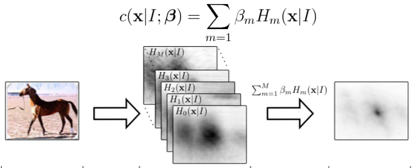
Hough forest regresses the center of the horse based on parts of the horse.

Detection

Equilibrium

- **Unnormalized** price map
- Negative leaves do not vote

$$c(\mathbf{x}|I;\boldsymbol{\beta}) = \sum_{m=1}^{M}\beta_m H_m(\mathbf{x}|I)$$



Hough Market

Training

Equilibrium

- **Normalized** price map
- Negative leaves vote as a uniform mass

$$c(\mathbf{x}|I;\boldsymbol{\beta}) = \frac{1}{Z}\sum_{m=1}^{M}\beta_m H_m(\mathbf{x}|I)$$

Update Rule

- For positives, reward kernel is Gaussian centered about the ground truth center

$$K(\mathbf{x};\mathbf{x}^*) = e^{-\frac{1}{2\sigma^2}\|\mathbf{x}-\mathbf{x}^*\|^2}$$

- For negatives, reward kernel taken to be uniform
- Update integral estimated as a Riemann sum

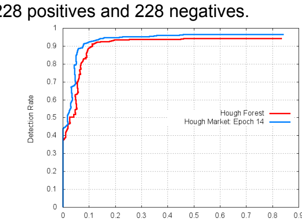
$$\beta_m \leftarrow \beta_m + \eta\beta_m\sum_{\mathbf{x}}\left[K(\mathbf{x};\mathbf{x}^*)\frac{H_m(\mathbf{x}|I)}{c(\mathbf{x}|I;\boldsymbol{\beta})} - \frac{H_m(\mathbf{x}|I)}{Z}\right]$$

- Step size depends on whether an image is positive or negative

$$\eta = \begin{cases} 0.05\eta_{\max} & \text{positive} \\ 0.5\eta_{\max} & \text{negative} \end{cases}$$

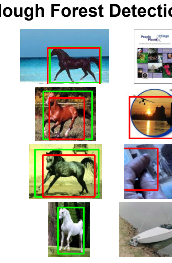
Results

- Trained on 100 positives and 50 negatives from the Weizmann Horse data set.
- Tested on 228 positives and 228 negatives.



ROC curve of Hough Market versus Hough Forest. Hough Forest attain 90% detection rate with 11% false alarm while Hough Market attains 90% detection rate with 8% false alarm.

Hough Forest Detections



Hough Market Detections