Strategic Games.

N players.

Each player has strategy set. $\{S_1,\dots,S_N\}$.

Vector valued payoff function: $u(s_1,\dots,s_n)$ (e.g., $\in \Re^N$).

Example:

2 players

Player 1: { **D**efect, **C**ooperate }.
Player 2: { **D**efect, **C**ooperate }.

Payoff:

	C	**D**
C	(3,3)	(0,5)
D	(5,0)	(1,1)

Famous because?

	C	**D**
C	(3,3)	(0,5)
D	(5,0)	(.1.1)

What is the best thing for the players to do?

Both cooperate. Payoff $(3,3)$.

If player 1 wants to do better, what do they do?

Defects! Payoff $(5,0)$

What does player 2 do now?

Defects! Payoff $(.1,.1)$.

Stable now!

Nash Equilibrium: neither player has incentive to change strategy.

Digression..

What situations?

Prisoner's dilemma:
Two prisoners separated by jailors and asked to betray partner.

Basis of the free market.
Companies compete, don't cooperate.
No Monopoly:
E.G., OPEC, Airlines, .
Should defect.
Why don't they?
Free market economics ...not so much?
More sophisticated models ,e.g, iterated dominance, coalitions, complexity..
Lots of interesting Game Theory!

This class(today): simpler version.

Two Person Zero Sum Games

2 players.

Each player has strategy set:
m strategies for player 1 n strategies for player 2

Payoff function: $u(i,j) = (-a,a)$ (or just a).
"Player 1 pays a to player 2."

Zero Sum: Payoff for any pair of strategies sums to 0.

Payoffs by m by n matrix: A.
Row player minimizes, column player maximizes.

Roshambo: rock,paper, scissors.

	R	P	S
R	0	1	-1
P	-1	0	1
S	1	-1	0

Any Nash Equilibrium?

(R,R)? no. (R,P)? no. (R,S)? no.

Mixed Strategies.

		R	P	S
		.33	.33	.33
R	.33	0	1	-1
P	.33	-1	0	1
S	.33	1	-1	0

How do you play?

Player 1: play each strategy with equal probability.
Player 2: play each strategy with equal probability.

Definitions.

Mixed strategies: Each player plays distribution over strategies.

Pure strategies: Each player plays single strategy.

Payoffs: Equilibrium.

		R	P	S
		.33	.33	.33
R	.33	0	1	-1
P	.33	-1	0	1
S	.33	1	-1	0

Payoffs? Can't just look it up in matrix!.

Average Payoff. Expected Payoff.

Sample space: $\Omega = \{(i,j) : i,j \in [1,..,3]\}$
Random variable X (payoff).

$$E[X] = \sum_{(i,j)} X(i,j)Pr[(i,j)].$$

Each player chooses independently:
$Pr[(i,j)] = \frac{1}{3} \times \frac{1}{3} = \frac{1}{9}$.

$$E[X] = 0.^1$$

[1] Remember zero sum games have one payoff.

Equilibrium

		R	P	S
		$.3\overline{3}$	$.3\overline{3}$	$.3\overline{3}$
R	$.3\overline{3}$	0	1	-1
P	$.3\overline{3}$	-1	0	1
S	$.3\overline{3}$	1	-1	0

Will Player 1 change strategy? Mixed strategies uncountable!

Expected payoffs for pure strategies for player 1.

Expected payoff of Rock? $\frac{1}{3} \times 0 + \frac{1}{3} \times 1 + \frac{1}{3} \times -1 = 0.$
Expected payoff of Paper? $\frac{1}{3} \times -1 + \frac{1}{3} \times 0 + \frac{1}{3} \times 1 = 0.$
Expected payoff of Scissors? $\frac{1}{3} \times 1 + \frac{1}{3} \times -1 + \frac{1}{3} \times 0 = 0.$

No better pure strategy. $\implies$ No better mixed strategy!

Mixed strat. payoff is weighted av. of payoffs of pure strats.

$E[X] = \sum_{(i,j)} (Pr[i] \times Pr[j]) X(i,j) = \sum_i Pr[i] (\sum_j Pr[j] \times X(i,j))$

Mixed strategy can't be better than the best pure strategy.

Player 1 has no incentive to change! Same for player 2.

Equilibrium!

Another example plus notation.

Rock, Paper, Scissors, prEempt.
PreEmpt ties preEmpt, beats everything else.
Payoffs.

	R	P	S	E
R	0	1	-1	1
P	-1	0	1	1
S	1	-1	0	1
E	-1	-1	-1	0

Equilibrium? **(E,E)**. Pure strategy equilibrium.
Notation: Rock is 1, Paper is 2, Scissors is 3, prEmpt is 4.
Payoff Matrix.

$$A = \begin{bmatrix} 0 & 1 & -1 & 1 \\ -1 & 0 & 1 & 1 \\ 1 & -1 & 0 & 1 \\ -1 & -1 & -1 & 0 \end{bmatrix}$$

Playing the boss...

Row has extra strategy:Cheat.
Ties with Rock, Paper, beats scissors.
Payoff matrix:
Rock is strategy 1, Paper is 2, Scissors is 3, and Cheat is 4 (for row.)

$$A = \begin{bmatrix} 0 & 1 & -1 \\ -1 & 0 & 1 \\ 1 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

Note: column knows row cheats.
Why play?
Row is column's advisor.
... boss.

Equilibrium: play the boss...

$$A = \begin{bmatrix} 0 & 1 & -1 \\ -1 & 0 & 1 \\ 1 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

Equilibrium:
Row: $(0, \frac{1}{3}, \frac{1}{6}, \frac{1}{2})$. Column: $(\frac{1}{3}, \frac{1}{2}, \frac{1}{6})$.

Payoff? Remember: weighted average of pure strategies.

Row Player.

Strategy 1: $\frac{1}{3} \times 0 + \frac{1}{2} \times 1 + \frac{1}{6} \times -1 = \frac{1}{3}$
Strategy 2: $\frac{1}{3} \times -1 + \frac{1}{2} \times 0 + \frac{1}{6} \times 1 = -\frac{1}{6}$
Strategy 3: $\frac{1}{3} \times 1 + \frac{1}{2} \times -1 + \frac{1}{6} \times 0 = -\frac{1}{6}$
Strategy 4: $\frac{1}{3} \times 0 + \frac{1}{2} \times 0 + \frac{1}{6} \times -1 = -\frac{1}{6}$

Payoff is $0 \times \frac{1}{3} + \frac{1}{3} \times (-\frac{1}{6}) + \frac{1}{6} \times (-\frac{1}{6}) + \frac{1}{2} \times (-\frac{1}{6}) = -\frac{1}{6}$

Column player: every column payoff is $-\frac{1}{6}$.

Both only play optimal strategies! Complementary slackness.

Why not play just one? Change payoff for other player!

Two person zero sum games.

$m \times n$ payoff matrix A.

Row mixed strategy: $x = (x_1, \ldots, x_m)$.
Column mixed strategy: $y = (y_1, \ldots, y_n)$.

Payoff for strategy pair (x, y):

$$p(x,y) = x^t A y$$

That is,

$$\sum_i x_i \left(\sum_j a_{i,j} y_j \right) = \sum_j \left(\sum_i x_i a_{i,j} \right) y_j.$$

Recall row minimizes, column maximizes.

Equilibrium pair: (x^*, y^*)?

$$(x^*)^t A y^* = \max_y (x^*)^t A y = \min_x x^t A y^*.$$

(No better column strategy, no better row strategy.)

Equilibrium.

Equilibrium pair: (x^*, y^*)?

$$p(x,y) = (x^*)^t A y^* = \max_y (x^*)^t A y = \min_x x^t A y^*.$$

(No better column strategy, no better row strategy.)

No row is better:
$\min_i A^{(i)} \cdot y = (x^*)^t A y^*$. [2]

No column is better:
$\max_j (A^t)^{(j)} \cdot x = (x^*)^t A y^*$.

[2] $A^{(i)}$ is ith row.

Best Response

Column goes first:
Find y, where best row is not too low..

$$R = \max_y \min_x (x^t A y).$$

Note: x can be $(0,0,\ldots,1,\ldots 0)$.

Example: Roshambo. Value of R?

Row goes first:
Find x, where best column is not high.

$$C = \min_x \max_y (x^t A y).$$

Agin: y of form $(0,0,\ldots,1,\ldots 0)$.

Example: Roshambo. Value of C?

Duality.

$$R = \max_y \min_x (x^t Ay).$$
$$C = \min_x \max_y (x^t Ay).$$

Weak Duality: $R \leq C$.
Proof: Better to go second.
Blindly play go-first strategy. $\square$

At Equilibrium (x^*, y^*), payoff v:
row payoffs (Ay^*) all $\geq v \implies R \geq v$.
column payoffs $((x^*)^t A)$ all $\leq v \implies v \geq C$.
$\implies R \geq C$

Equilibrium $\implies R = C$!

Strong Duality: There is an equilibrium point! and $R = C$!

Doesn't matter who plays first!

Equilibrium existence.

Linear programs.

Column player: find y to maximize row payoffs.
$\max z, Ay \geq z, \sum_i y_i = 1$
Row player: find x to minimize column payoffs.
$\min z, A^T x \leq z, \sum_i x_j = 1$.

Primal dual optimal are equilibrium solution.

Strong Duality: linear program.

Aproximate equilibrium ...

$C(x) = \max_y x^t Ay$

$R(y) = \min_x x^t Ay$

Always: $R(y) \leq C(x)$
For $R(y)$, minimizer x "goes second", but goes first for $C(x)$.

Strategy pair: (x,y)

Equilibrium: (x,y)

$R(y) = C(x) \to C(x) - R(y) = 0$.

Approximate Equilibrium: $C(x) - R(y) \leq \varepsilon$.

With $R(y) < C(x)$
$\to$ "Defense y to x is within ε of best response"
$\to$ "Defense x to y is within ε of best response"

Games and experts

Again: find (x^*, y^*), such that

$(\max_y x^* Ay) - (\min_x xAy^*) \leq \varepsilon$

$C(x^*) \quad - \quad R(y^*) \leq \varepsilon$

Experts Framework:
n Experts, T days, L^* -total loss of best expert.

Multiplicative Weights Method yields loss L where

$L \leq (1+\varepsilon)L^* + \frac{\log n}{\varepsilon}$

Games and Experts.

Assume: A has payoffs in $[0,1]$.

For $T = \frac{\log n}{\varepsilon^2}$ days:

1) m pure row strategies are experts.
Use multiplicative weights, produce row distribution.

Let x_t be distribution (row strategy) on day t.

2) Each day, adversary plays best column response to x_t.
Choose column of A that maximizes row's expected loss.

Let y_t be indicator vector for "best" response column.

Approximate Equilibrium!

Experts: x_t is strategy on day t, y_t is best column against x_t.

Let $y^* = \frac{1}{T}\sum_t y_t$ and $x^* = \text{argmin}_{x_t} x_t A y_t$.

Claim: (x^*, y^*) are 2ε-optimal for matrix A.

Column payoff: $C(x^*) = \max_y x^* Ay$.
Loss on day t, $x_t A y_t \geq x^* A y_{t^*} = C(x^*)$ by the choice of x^* .
Thus, algorithm loss, L, is $\geq T \times C(x^*)$.

Best expert: L^*- best row against all the columns played.

best row against $\sum_t A y_t$ and $T \times y^* = \sum_t y_t$
$\rightarrow$ best row against $T \times Ay^*$.
$\rightarrow L^* \leq T \times R(y^*)$.

Multiplicative Weights: $L \leq (1+\varepsilon)L^* + \frac{\ln n}{\varepsilon}$

$T \times C(x^*) \leq (1+\varepsilon)T \times R(y^*) + \frac{\ln n}{\varepsilon} \rightarrow C(x^*) \leq (1+\varepsilon)R(y^*) + \frac{\ln n}{\varepsilon T}$
$\rightarrow C(x^*) - R(y^*) \leq \varepsilon R(y^*) + \frac{\ln n}{\varepsilon T}$.

$T = \frac{\ln n}{\varepsilon^2}$, $R(y^*) \leq 1$
$\rightarrow C(x^*) - R(y^*) \leq 2\varepsilon$.

Approximate Equilibrium: slightly different!

Experts: x_t is strategy on day t, y_t is best column against x_t.

Let $x^* = \frac{1}{T}\sum_t x_t$ and $y^* = \frac{1}{T}\sum_t y_t$.

Claim: (x^*, y^*) are 2ε-optimal for matrix A.

Left as exercise.

Comments

For any ε, there exists an ε-Approximate Equilibrium.
Does an equilibrium exist? Yes.

Something about math here?
Limit of a sequence on some closed set..hmmm..

More comments

Complexity?
$T = \frac{\ln n}{\varepsilon^2} \rightarrow O(nm\frac{\log n}{\varepsilon^2})$. Basically linear!

Versus Linear Programming: $O(n^3 m)$ Basically quadratic.
(Faster linear programming: $O(\sqrt{n+m})$ linear system solves.)
Still much slower ... and more complicated.

Dynamics: best response, update weight according to loss, ...

Near integrality.
Only $\ln n/\varepsilon^2$ non-zero column variables.
Average $1/T$, so not too many nonzeros and not too small.

Not stochastic at all here, the column responses are adversarial.

Various assumptions: $[0,1]$ losses, other ranges takes some work.

Toll/Congestion

Given: $G = (V, E)$.
Given $(s_1, t_1) \ldots (s_k, t_k)$.

Problem: Route path for each pair and minimize maximum congestion.

Congestion is maximum number of paths that use any edge.

Note: Number of paths is exponential.
Can encode in polysized linear program, but large.

Toll/Congestion

Given: $G = (V, E)$.
Given $(s_1, t_1) \ldots (s_k, t_k)$.
Row: choose routing of all paths. (Exponential)
Column: choose edge.
Row pays if column chooses edge on any path.

Matrix:
row for each routing: r
column for each edge: e

$A[r, e]$ is congestion on edge e by routing r

Offense: (Best Response.)
Router: route along shortest paths.
Toll: charge most loaded edge.

Defense: Toll: maximize shortest path under tolls.
Route: minimize max congestion on any edge.

Two person game.

Row is router.

An exponential number of rows!

Two person game with experts won't be so easy to implement.

Version with row and column flipped may work.

$A[e,r]$ - congestion of edge e on routing r.

m rows. Exponential number of columns.

Multiplicative Weights only maintains m weights.

Adversary only needs to provide best column each day.

Runtime only dependent on m and T (number of days.)

Congestion minimization and Experts.

Will use gain and $[0,\rho]$ version of experts:

$G \geq (1-\varepsilon)G^* - \frac{\rho \log n}{\varepsilon}$.

Let $T = \frac{k \log n}{\varepsilon^2}$

1. Row player runs multiplicative weights on edges:
   $w_i = w_i(1+\varepsilon)^{g_i/k}$.
2. Column routes all paths along shortest paths.
3. Output the average of all routings: $\frac{1}{T}\sum_t f(t)$.

Claim: The congestion, c_{max} is at most $C^* + 2k\varepsilon/(1-\varepsilon)$.

Proof:
$G \geq G^*(1-\varepsilon) - \frac{k \log n}{\varepsilon T} \rightarrow G^* - G \leq \varepsilon G^* + \frac{k \log n}{\varepsilon}$

$G^* = T * c_{\max}$ – Best row payoff against average routing (times T).

$G \leq T \times C^*$ – each day, gain is avg. congestion $\leq$ opt congestion.

$T = \frac{k \log n}{\varepsilon^2} \rightarrow Tc_{\max} - TC^* \leq \varepsilon Tc_{\max} + \frac{k \log n}{\varepsilon} \rightarrow$
$c_{max} - C^* \leq \varepsilon c_{max} + \varepsilon$ □

Better setup.

Runtime: $O(km \log n)$ to route in each step (using Dijkstra's)

$O(\frac{k \log n}{\varepsilon^2})$ steps
to get $c_{\max} - C^* < \varepsilon C^*$ (assuming $C^* > 1$) approximation.

To get constant c error.
$\rightarrow O(k^2 m \log n/\varepsilon^2)$ to get a constant approximation.
Exercise: $O(km \log n/\varepsilon^2)$ algorithm ! ! !

Fractional versus Integer.

Did we (approximately) solve path routing?
Yes? No?

No! Average of T routings.
We approximately solved fractional routing problem.

No solution to the path routing problem that is $(1+\varepsilon)$ optimal!

Decent solution to path routing problem?

For each s_i, t_i, choose path p_i at random from "daily" paths.

Congestion $c(e)$ edge has expected congestion, $\tilde{c}(e)$, of $c(e)$.

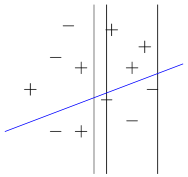
"Concentration" (law of large numbers)
$c(e)$ is relatively large ($\Omega(\log n)$)
$\rightarrow \tilde{c}(e) \approx c(e)$.

Concentration results? later.

Learning

Learning just a bit.

Example: set of labelled points, find hyperplane that separates.



Looks hard.

1/2 of them? Easy.
Arbitrary line. And Scan.

Useless. A bit more than 1/2 Correct would be better.

Weak Learner: Classify $\geq \frac{1}{2} + \varepsilon$ points correctly.

Not really important but ...

Weak Learner/Strong Learner

Input: n labelled points.

Weak Learner:
produce hypothesis correctly classifies $\frac{1}{2} + \varepsilon$ fraction

Strong Learner:
produce hyp. correctly classifies $1 + \mu$ fraction

That's a really strong learner!
Strong Learner:
produce hypothesis correctly classifies $1 - \mu$ fraction

Same thing?

Can one use weak learning to produce strong learner?

Boosting: use a weak learner to produce strong learner.

Poll.

Given a weak learning method (produce ok hypotheses.)
produce a great hypothesis.

Can we do this?

(A) Yes

(B) No

If yes. How?

The idea: Multiplicative Weights.
Standard online optimization method reinvented in many areas.

Boosting/MW Framework

Points lose when classified correctly.
The little devils want to fool the learner.

Learner classifies weighted majority of points correctly.

Strong learner algorithm from many weak learners!

Initialize: all points have weight 1.

Do $T = \frac{2}{\varepsilon^2} \ln \frac{1}{\mu}$ rounds

1. Find $h_t(\cdot)$ correct on $1/2+\gamma$ of weighted points.
2. Multiply each point that is correct by $(1-\varepsilon)$.

Output hypotheses $h(x)$: majority of $h_1(x), h_2(x), \ldots, h_T(x)$.

Claim: $h(x)$ is correct on $1-\mu$ of the points ! ! !

Cool!

Really? Proof?

Logarithm

$ln(1-x) = (-x - x^2/2 - x^3/3....)$ Taylors formula for $|x| < 1$.

Implies: for $x \le 1/2$, that $-x - x^2 \le \ln(1-x) \le -x$.

The first inequality is from geometric series.

$x^3/3 + ... = x^2(x/3 + x^2/4 + ..) \le x^2(1/2)$ for $|x| < 1/2$.

The second is from truncation.

Second implies: $(1-\varepsilon)^x \le e^{-\varepsilon x}$, by exponentiation.

Adaboost proof.

Claim: $h(x)$ is correct on $1-\mu$ of the points!

Let S_{bad} be the set of points where $h(x)$ is incorrect.

majority of $h_t(x)$ are wrong for $x \in S_{bad}$.

point $x \in S_{bad}$ is winning – loses less than $\frac{1}{2}$ the time.

$W(T) \ge (1-\varepsilon)^{\frac{T}{2}} |S_{bad}|$

Each day t, weak learner penalizes $\ge \frac{1}{2} + \gamma$ of the weight.

Loss $L_t \ge (1/2 + \gamma)$

$\rightarrow W(t+1) \le W(t)(1 - \varepsilon(L_t)) \le W(t)e^{-\varepsilon L_t}$

$\rightarrow W(T) \le ne^{-\varepsilon \sum_t L_t} \le ne^{-\varepsilon(\frac{1}{2}+\gamma)T}$

Combining

$|S_{bad}|(1-\varepsilon)^{T/2} \le W(T) \le ne^{-\varepsilon(\frac{1}{2}+\gamma)T}$

Calculation..

$|S_{bad}|(1-\varepsilon)^{T/2} \le ne^{-\varepsilon(\frac{1}{2}+\gamma)T}$

Set $\varepsilon = \gamma$, take logs.

$\ln\left(\frac{|S_{bad}|}{n}\right) + \frac{T}{2}\ln(1-\gamma) \le -\gamma T(\frac{1}{2}+\gamma)$

Again, $-\gamma - \gamma^2 \le \ln(1-\gamma)$,

$\ln\left(\frac{|S_{bad}|}{n}\right) + \frac{T}{2}(-\gamma - \gamma^2) \le -\gamma T(\frac{1}{2}+\gamma) \rightarrow \ln\left(\frac{|S_{bad}|}{n}\right) \le -\frac{\gamma^2 T}{2}$

And $T = \frac{2}{\gamma^2}\log\mu$,

$\rightarrow \ln\left(\frac{|S_{bad}|}{n}\right) \le \log\mu \rightarrow \frac{|S_{bad}|}{n} \le \mu.$

The misclassified set is at most μ fraction of all the points.

The hypothesis correctly classifies $1-\mu$ of the points !

Claim: Multiplicative weights: $h(x)$ is correct on $1-\mu$ of the points!

Some details...

Weak learner learns over distributions of points not points.

Make copies of points to simulate distributions.

Used often in machine learning.
Blending learning methods.

Theme:Good on average, hyperplane.

"Duality"

$\min cx, Ax \geq b, x \geq 0$.

Linear combination of constraints: $y^T Ax \geq y^T c$
Find a solution for just one constraint!!!
Best response.

Multiplicative weights: two person games (linear programs)
y is exponential weights on "how unsatisfied" each equation is.
$y_i \propto \sum_t (1+\varepsilon)^{(a_i x^{(t)} - b_i)}$
y "wins" $\equiv$ unsatisfiable linear combo of constraints.
Otherwise, x eventually "wins".
Or pair that are pretty close.

(Apologies: switched x and y in game setup.)

"Separating" Hyperplane?
y^T "separates" affine subspace Ax from $\geq y^T c$.

The math: $e = \lim_{n\to\infty}(1+1/n)^n$.

A step closer.

Another Algorithm.

Finding a feasible point: x^* for constraints.
If $x^{(t)}$ point violates constraint by $> \varepsilon$
move toward constraint.
Closer.

The Math:
Wrong side, angle to correct point is less than 90°

This is the idea in perceptron. But can do analysis directly.

Multiplicative weights and a step closer.

The solution is a distribution: p^*.

Every day each strategy loses (or not), $\ell_i^(t)$.
Assumption: Solution doesn't lose (much).

MW: keeps a distribution.

Closer?
Distance is $\sum_i \log(p_i^*/q_i)$.
Step in MW gets closer to p^* with this distance.
Idea: p^* loses less,
so new distribution plays losers less.
Move toward playing losers less.
Thus closer to p^*.

The math:
linear (and quadratic) approximation of e^x.
Advantage?
Distributions have entropy at most $O(\log n)$.

Reinforcement learning == Bandits.

Multiplicative Weights framework:
Update all experts.

Bandits.
Only update experts you choose.
No information about others.
(Named after one-armed bandit slot machine.)

Idea: "Learn" which expert is best.
Prof. Dragan's mantra: formulation as optimization.

Exploration: choose new bandit to get "data".
Exploitation: choose best bandit.

Strategy:
Multiplicative weights.
Update by $(1+\varepsilon)$.

Big ε.
Exploit or explore more? Exploit.

Perceptron also like bandits. One point at a time.

Online optimization: limited information.