ELO Ratings and the Bradley-Terry Model

1. Elo Model

- Elo Rating/Strength of team i is $\beta_i \in \mathbb{R}$ (model parameters)
- (Known) CDF $F(\cdot)$, so $\begin{cases} \lim_{z \to -\infty} F(z) = 0, \\ \lim_{z \to +\infty} F(z) = 1, \\ F \text{ is an increasing function}, \\ F \text{ is continuous} \end{cases}$
- Model $\quad P_{i,j} := \mathbb{P}(\text{team } i \text{ beats team } j) = F(\beta_i - \beta_j)$

"the probability that i beats j is a function of the difference of strengths"

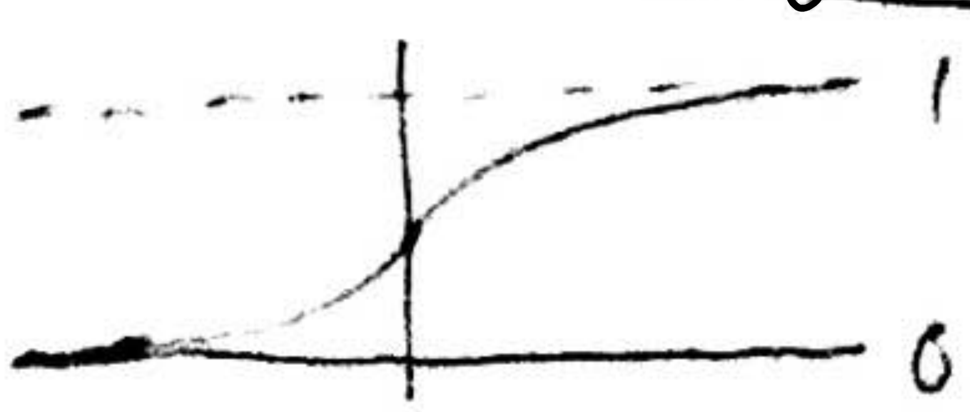
- Also, $\quad 1 - F(z) = F(-z)$

Since $\quad P_{i,j} + P_{j,i} = 1 \quad$ (assuming no ties)

and $\quad 1 - F(\beta_i - \beta_j) = 1 - P_{i,j} = P_{j,i} = F(\beta_j - \beta_i) = F(-(\beta_i - \beta_j))$

2. Bradley-Terry Model

- Use the Logistic CDF, $\quad F(z) = \sigma(z) = \dfrac{1}{1+e^{-z}}$



- Model $\quad P_{ij} = \sigma(\beta_i - \beta_j) = \dfrac{1}{1+e^{-(\beta_i - \beta_j)}} = \dfrac{e^{\beta_i}}{e^{\beta_i} + e^{\beta_j}}$

Equivalently, log-odds satisfy $\quad \log\left(\dfrac{P_{ij}}{1-P_{ij}}\right) = \beta_i - \beta_j$

- Model is OVERPARAMETERIZED: can add constant c to each β_i, and then the differences $\beta_i - \beta_j$ remain unchanged. Can fix this by setting $\beta_i \equiv 0$ for some team i. If doing NBA Elo ratings, set $\beta_{\text{knicks}} = 0$ since they suck (usually).
- ORDER a game between teams i and j as (i,j), where i = home team, j = away team. Then, we can add a Home Court advantage term via an intercept term α:

- Bradley-Terry model $\boxed{P_{ij} = \sigma(\beta_i - \beta_j + \alpha) = \dfrac{1}{1+e^{-(\beta_i - \beta_j + \alpha)}}}$

3. Bradley-Terry Model is Logistic Regression

[redacted]

- Setup

K teams $\{1, \dots, K\}$

n matches $(i_1, j_1), \dots, (i_n, j_n)$

n outcomes $y_1, \dots, y_n \in \{0, 1\}$

$0 \to$ away team wins

$1 \to$ home team wins

$y_m = y_{i_m, j_m} = \mathbb{1}\{i_m \text{ beat } j_m\}$.

Model

$y_m \sim \text{Bernoulli}(P_m)$.

K parameters/weights $\boxed{\vec{\theta} = (\alpha, 0, \beta_2, \dots, \beta_K)}$

Recall $\beta_1 = \beta_{\text{knicks}} = 0$

Match-index-vector $\boxed{\vec{X}_{ij} = (1, 0, \dots, 0, 1, 0, \dots, 0, -1, 0, \dots, 0)}$

index 0 is a 1 for α

index i is a 1 for β_i

index j is a -1 for β_j

Therefore $\theta^T X_{ij} = \beta_i - \beta_j + \alpha$

Hence $\boxed{P_{ij} = \sigma(\beta_i - \beta_j + \alpha) = \sigma(\theta^T X_{ij})}$ $\boxed{y_{ij} \sim \text{Bernoulli}(P_{ij})}$

- Bradley-Terry Elo Ratings are the parameters/weights of a logistic regression used to predict pairwise game outcomes!

- <u>Training Dataset</u> $\boxed{\mathcal{D} = \{(\vec{X}_m, y_m)\}_{m=1}^{n}}$ [redacted]

match $m \in \{1, \dots, n\}$ is (i_m, j_m) home team i_m, away team j_m

match-index-vector $\vec{X}_m = X_{i_m, j_m}$

match outcome $y_m \in \{0, 1\}$, $y_m = \mathbb{1}\{i_m \text{ beats } j_m\}$

(4.) Training our Logistic Regression Model

- Cross-Entropy Loss Function. schedule → power scores

$$\hat{\theta} = \underset{\vec{\theta}}{\operatorname{argmax}}\ \mathbb{P}(\underbrace{y_1, \dots, y_n}_{\text{Game Outcomes}} \mid \vec{x}_1, \dots, \vec{x}_n, \vec{\theta})$$ maximum likelihood estimation

$$= \underset{\theta}{\operatorname{argmax}} \prod_{m=1}^{n} \mathbb{P}(y_m \mid \vec{x}_m, \theta)$$

$$= \underset{\theta}{\operatorname{argmax}} \prod_{m=1}^{n} p_m^{\,y_m} (1-p_m)^{1-y_m}$$

since we modelled $y_m \sim \text{Bernoulli}(p_m)$ where $p_m = p_{i_m, j_m} = \sigma(\theta^T X_{i_m, j_m})$

$$= \underset{\theta}{\operatorname{argmin}} \ -\sum_{m=1}^{n} \left[y_m \log p_m + (1-y_m) \log(1-p_m) \right]$$

$$= \underset{\theta}{\operatorname{argmin}}\ L(\theta)$$

where $L(\theta)$ is the cross-entropy loss function

- This minimization problem has no known [redacted] analytic/closed-form solution. However, L is convex, so we can use (Stochastic) Gradient Descent

- Note $\frac{\partial}{\partial z}\sigma(z) = \frac{\partial}{\partial z}\left(\frac{1}{1+e^{-z}}\right) = \frac{e^{-z}}{(1+e^{-z})^2} = \left(\frac{1}{1+e^{-z}}\right)\left(\frac{e^{-z}}{1+e^{-z}}\right) = \sigma(z)(1-\sigma(z))$

Hence [redacted]

$$\nabla_\theta p_m = \nabla_\theta\, \sigma(\theta^T \vec{x}_m) = \sigma(\theta^T \vec{x}_m)(1-\sigma(\theta^T \vec{x}_m))\, \vec{x}_m = p_m(1-p_m)\, \vec{x}_m$$

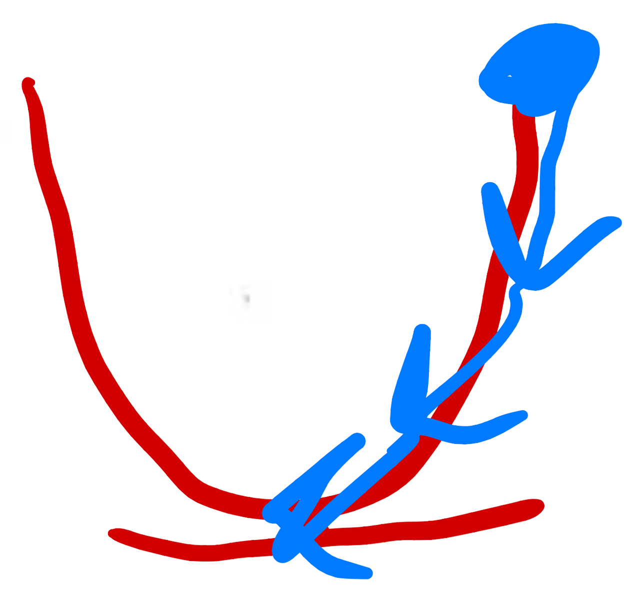
- Gradient of loss function

$$\nabla_\theta L(\theta) = -\sum_{m=1}^{n} \left[y_m (\nabla_\theta \log p_m) + (1-y_m)(\nabla_\theta \log(1-p_m)) \right]$$

newton Raphson

$$= -\sum_{m=1}^{n} \left[\frac{y_m}{p_m} (\nabla_\theta p_m) - \frac{(1-y_m)}{(1-p_m)} (\nabla_\theta p_m) \right]$$

$$= -\sum_{m=1}^{n} \left(\frac{y_m}{p_m} - \frac{(1-y_m)}{(1-p_m)} \right) \left(p_m (1-p_m)\, \vec{x}_m \right)$$



$$= -\sum_{m=1}^{n} \left(y_m (1-p_m) - (1-y_m) p_m \right) \vec{x}_m$$

$$= -\sum_{m=1}^{n} (y_m - p_m)\, \vec{x}_m = -\sum_{m=1}^{n} (y_m - \sigma(\theta^T \vec{x}_m))\, \vec{x}_m$$

- <u>Gradient Descent Update</u> $\vec{\theta}^{(t+1)} = \vec{\theta}^{(t)} - \eta \nabla_\theta L(\vec{\theta}^{(t)})$

$$\boxed{\vec{\theta}^{(t+1)} = \vec{\theta}^{(t)} + \eta \sum_{m=1}^{n} \left(y_m - \sigma\left(\theta^{(t)T} \vec{X}_m\right)\right) \vec{X}_m}$$

η = learning rate hyperparameter

Since loss function $L(\theta)$ is convex, g.d. will find global min, regardless of initial value!

- <u>updating the Elo Rating after a New Game is Played</u>

New match (i, j) becomes the $(n+1)^{st}$ match in our dataset.
New data $(\vec{X}_{n+1}, y_{n+1})$.

<u>G.D. update</u>

$$\theta^{(t+1)} = \theta^{(t)} + \eta \left(y_{n+1} - \sigma\left(\theta^{(t)T} \vec{X}_{n+1}\right)\right) \vec{X}_{n+1}$$

This Reduces to

$$\begin{cases} \beta_i \leftarrow \beta_i + \eta \left(y_{ij} - \sigma(\beta_i - \beta_j + \alpha)\right) \\ \beta_j \leftarrow \beta_j - \eta \left(y_{ij} - \sigma(\beta_i - \beta_j + \alpha)\right) \\ \alpha \leftarrow \alpha + \eta \left(y_{ij} - \sigma(\beta_i - \beta_j + \alpha)\right) \end{cases}$$

where $y_{ij} = \mathbb{1}\{i \text{ beat } j\}$

~~Thus we may interpret η as the amount of~~

- However, actual Elo implementations use <u>Early stopping</u>: perform only 1 gradient descent update, using the <u>K-factor</u> K as the learning rate instead of (tiny) η. (WHY?!) (prevents overfitting)
- Also, α is not updated in general, after its initial setting. So, treat α more like a hyperparameter than a parameter. (WHY??)

Extra Adjustments for NFL Elo Ratings, from 538

- Home field adjustment (+55 pts)
- Travel adjustment +4 for every 1000 miles travelled
- Rest Adjustment +25 if off a bye week
- playoff adjustment multiply elodiff by 1.2
- QB adjustment
- K factor: how much a single new game will impact the elo ratings
- Margin of victory multiplier – more credit for blowout wins