How strong is my opponent?

Using Bayesian methods for skill assessment

Dr. Darina Goldin
Bayes Esports Solutions

github.com/drdarina/ratings_talk

Who’s talking?

Darina Goldin

Lead Data Scientist at
Bayes Esports Solutions

- Data exchange platform
- Esports directory
- Esports betting odds

bayes

What are we talking about?

Skill rating: an estimation of the true skill of a competitor from their observed competition results

Expectation management

We will:

- Get a feeling for ratings
- Talk a lot about Elo
- Talk a little about Glicko
- Briefly mention TrueSkill
- Use Bundesliga as an example

We will not:

- Talk a lot about Bundesliga
- Have understood G2 and TS in great mathematical detail
- Have learned implementation details (code available on github)
- Have learned details about factor graphs
- Have discussed player-based TS / partial play

bayes

1. Introduction

Why do we need accurate ratings?

- Predicting match outcomes
- Identifying upsets
- Qualification for tournaments
- Incentive to improve performance
- Entertainment
- Balancing matches

bayes

Why do we need accurate ratings?

- Predicting match outcomes
- Identifying upsets
- Qualification for tournaments
- Incentive to improve performance
- Entertainment
- Balancing matches

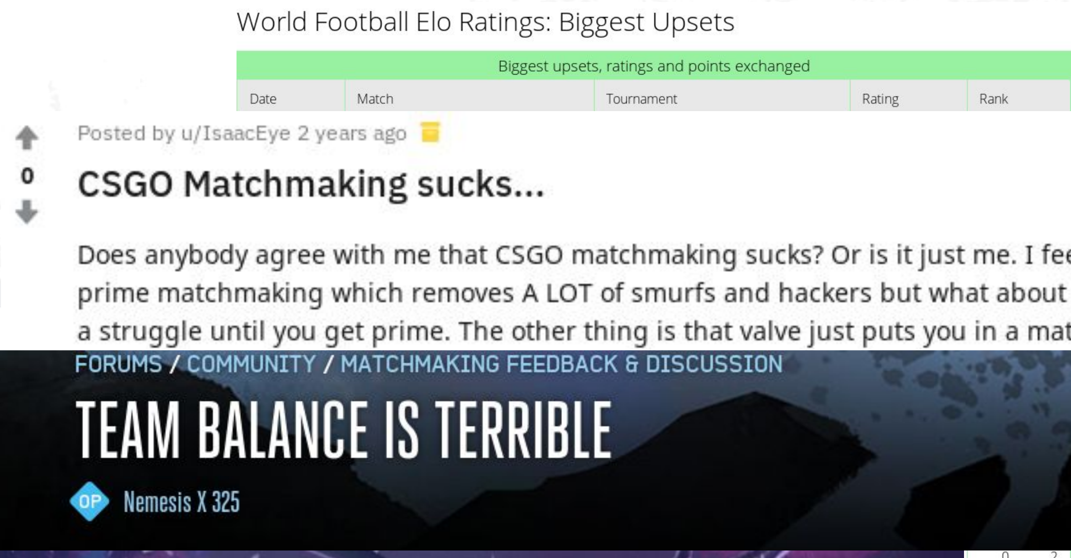
World Football Elo Ratings: Biggest Upsets

Biggest upsets, ratings and points exchanged							
Date	Match		Tournament	Rating		Rank	
August 28 1920	Norway England	3 1	Olympic Games in Belgium	+87 −87	1565 1955	0 −3	25 4
May 25 1924	Italy Spain	1 0	Olympic Games in France	+53 −53	1730 1965	+5 −1	17 2
July 14 1930	Yugoslavia Brazil	2 1	World Cup in Uruguay	+53 −53	1628 1877	+6 −2	27 9
May 27 1934	Sweden Argentina	3 2	World Cup in Italy	+52 −52	1783 2013	+3 −1	14 2
August 4 1936	Japan Sweden	3 2	Olympic Games in Germany	+56 −56	1332 1701	+9 −7	58 21
August 7 1936	Germany Norway	0 2	Olympic Games in Germany	−76 +76	1830 1790	−3 +6	11 13
August 5 1948	Denmark Italy	5 3	Olympic Games in England	+75 −75	1834 1969	+4 0	12 3
June 29 1950	United States England	1 0	World Cup in Brazil	+57 −57	1625 2024	+14 −1	30 2
July 16 1950	Brazil Uruguay	1 2	World Cup in Brazil	−53 +53	2014 1871	0 +3	2 10
June 16 1982	Algeria West Germany	2 1	World Cup in Spain	+56 −56	1704 2052	+5 0	33 2
June 13 1998	Nigeria Spain	3 2	World Cup in France	+52 −52	1758 1987	+11 −1	26 4
May 31 2002	Senegal France	1 0	World Cup in South Korea	+54 −54	1756 2042	+11 0	29 1
June 17 2006	Ghana Czechia	2 0	World Cup in Germany	+82 −82	1688 1923	+9 −6	46 9
June 16 2010	Switzerland Spain	1 0	World Cup in South Africa	+53 −53	1814 2059	+9 −1	16 2
June 27 2018	South Korea Germany	2 0	World Cup in Russia	+80 −80	1756 1964	+20 −5	25 7

bayes

Why do we need accurate ratings?

- Predicting match outcomes
- Identifying upsets
- Qualification for tournaments
- Incentive to improve performance
- Entertainment
- Balancing matches



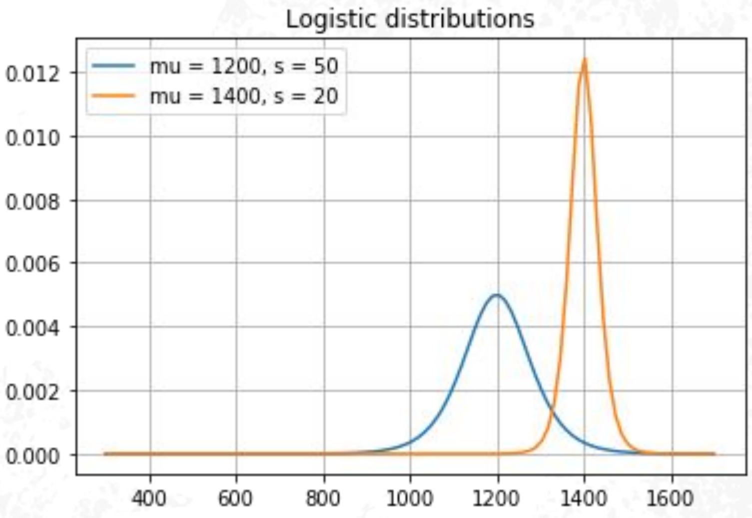
bayes

Requirements for ratings

- Easy to use in matchmakers
- Minimal tuning: should run once installed
- Good prediction of match outcome
- Fast convergence
- Easy to add new players
- No stagnation: every match has some impact
- Hard to manipulate by gaming the system

A mathematical-ish formulation

1. Assume each team has a skill drawn from a **distribution** (e.g. a Logistic distribution with mean mu_0 and scale s_0)
2. After each match, the team's skill **changes** by an unknown amount
3. Team exhibits a **real performance** in a match
4. Skill distribution **parameters are updated** after the result



For an efficient skill rating algorithm we need to:

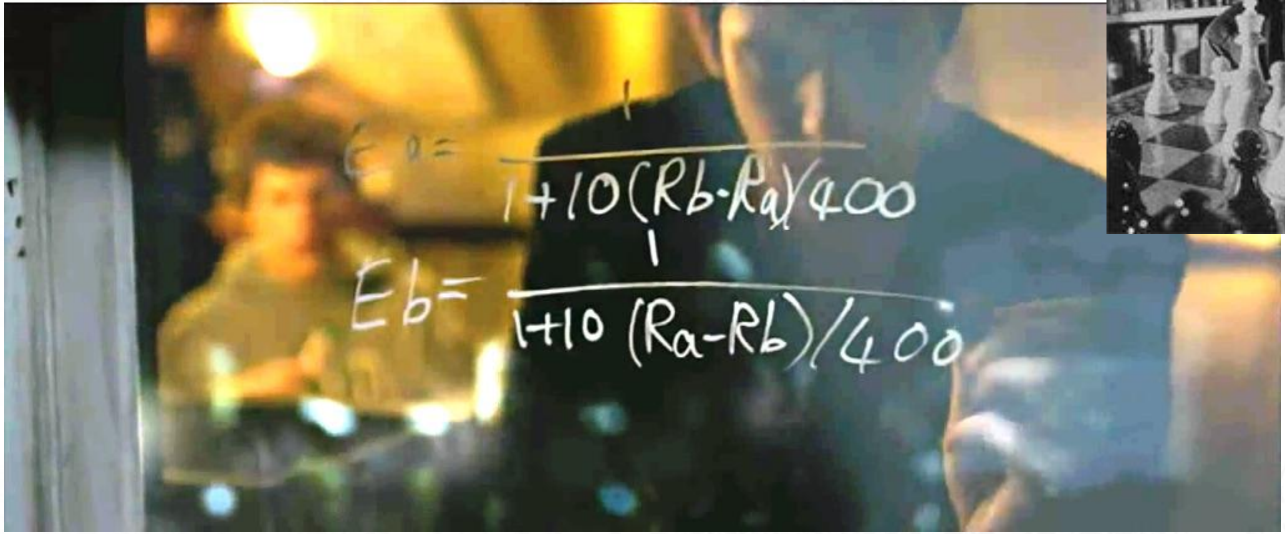
- Choose a distribution
- Choose its parameters
- Find an update rule

bayes

3. Elo

Elo

- Created by Arpad Elo in the 1960s
- Adopted by World Chess Federation in 1970
- Currently used in chess, baseball, basketball, ...



bayes

Elo

- Player starts with fixed amount of points as initial rating $\boldsymbol{R_A}$
- True player skill is approximated by a **logistic** distribution around $\boldsymbol{R_A}$ with scale ***s***
- Player plays against player B with rating $\boldsymbol{R_B}$ with same scale ***s***
- We can calculate the expected score of A vs. B:

$$E_A = \frac{1}{1 + 10^{\frac{R_B - R_A}{n}}}$$

What is n?

Logistic function CDF:

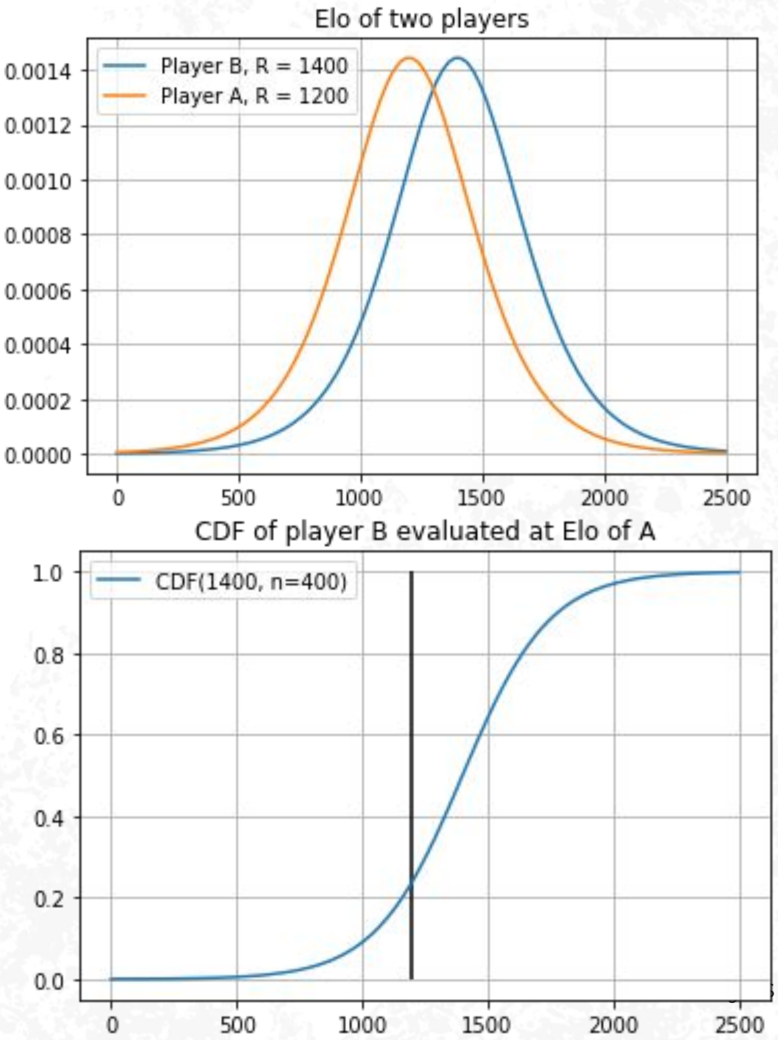
$$F(x;\mu,s) = \frac{1}{1+e^{-\frac{x-\mu}{s}}}$$

Substituting

$$s = n/ln(10) \qquad \mu = R_B$$

we obtain

$$E_A = \frac{1}{1+10^{\frac{R_B-R_A}{n}}}$$



Elo

- Player starts with fixed amount of points as initial rating R_A
- True player skill is approximated by a **logistic** distribution around R_A with scale ***s***
- Player plays against player B with rating R_B with same scale ***s***
- We can calculate the expected score of A vs. B:

$$E_A = \frac{1}{1 + 10^{\frac{R_B - R_A}{n}}}$$

=> n: number of Elo points for a player to be considered 10x better than another player

bayes

Elo

- Player starts with fixed amount of points as initial rating R_A
- True player skill is approximated by a **logistic** distribution around R_A with scale ***s***
- Player plays against player B with rating R_B with same scale ***s***
- We can calculate the expected score of A vs. B:

$$E_A = \frac{1}{1 + 10^{\frac{R_B - R_A}{n}}}$$

=> n: number of Elo points for a player to be considered 10x better than another player

- Real match outcome is given by S_A : 1 for win, 0.5 for draw, 0 for loss
- Rating update is performed by

$$\boxed{R'_A = R_A + K(S_A - E_A)}$$

bayes

What is K? $R'_A = R_A + K(S_A - E_A)$

- determines the "**sensitivity**" of how wins and losses impact Elo
- **K should be appropriate for n!**
- K should be appropriate for match format:
    - Many games (baseball, chess) → one match less important → K is small
    - Few games (NFL, BB) → every match matters → K is large
- **World chess federation**: Experience-based K
        - K = 40 for new players until 30 games
        - K = 20 for players with over 30 games but who never had Elo over 2400
        - K = 10 for everyone else
- **Football rankings**: Importance-based K
        - K = 60 for World Cup
        - K = 40 for World Cup Qualifiers
        - K = 30 for other tournaments
        - K = 20 for friendly matches

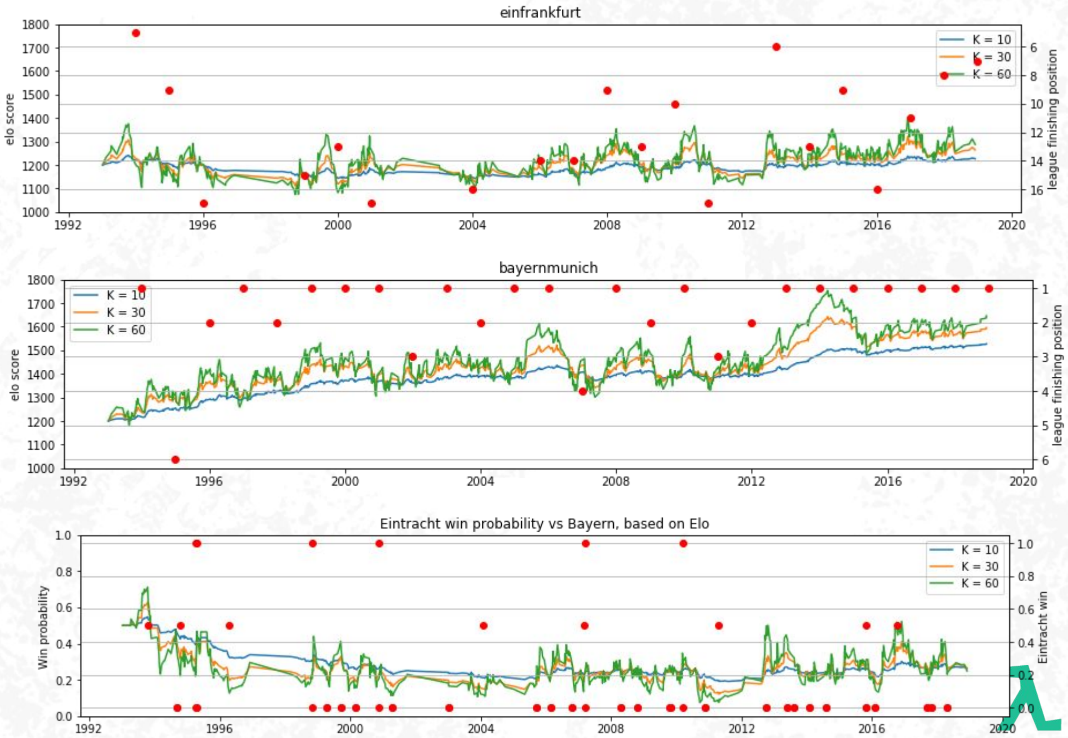
bayes

Influence of K on Elo convergence

Created using Elo implementation by Heungsub Lee:
github.com/sublee/elo
Available through pip

```
#: Default K-factor.
K_FACTOR = 10
#: Default rating class.
RATING_CLASS = float
#: Default initial rating.
INITIAL = 1200
#: Default Beta value.
BETA = 200
```

$$\beta = n/2$$



bayes

Dataset source: http://www.football-data.co.uk/germanym.php

Summary

Key ideas:

- Player skill approximated by a logistic distribution of fixed scale
- Win probability given by logistic distribution $CDF(R_B)$ evaluated at R_A
- Distribution mean is updated after each match, resulting in a new rating

Tuning:

- Choose initial rating
- Choose n
- Choose K
- Make adjustments for your domain:
  - variable/decaying K
  - home advantage
  - margin of victory
  - sky's the limit

bayes

How to tune parameters?

- Goal: minimize error in expected score vs. outcome
- Brier score:

$$BS = \frac{1}{N}\sum_{i=1}^{N}(E_i - S_i)^2$$

- Entire dataset is its own test set
- Ignore the calibration period (= first x scores)

bayes

Known issues

- **Inactivity** doesn't affect ratings
- Players protecting their rankings
- Time between matches not factored
- New players overvalued
- Inflation/deflation of rankings due to people with less than average/higher than average ratings retiring/joining → problem for **comparing historical data** with current
- Only two ratings updated at a time

bayes

Known issues

```python
# simplest elo rating calculator
def compute_elo_change(R1, R2, w1):
    K = 32
    n = 400
    E1 = 1 / (1 + 10**((R2 - R1)/400))
    E2 = 1 / (1 + 10**((R1 - R2)/400))
    S1 = 1 if w1 else 0
    S2 = 1 - S1
    R1 = R1 + K * (S1 - E1)
    R2 = R2 + K * (S2 - E2)
    return R1, R2
```

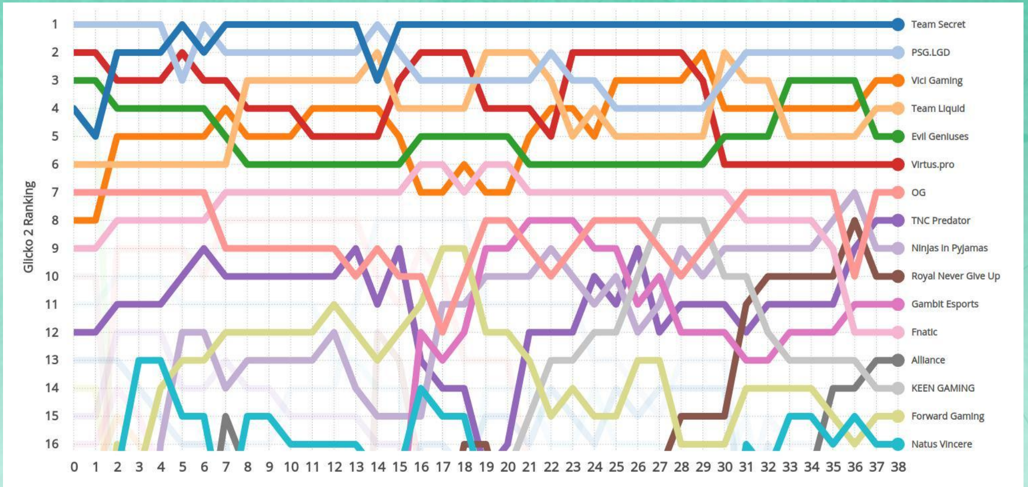
- **Inactivity** doesn't affect ratings
- Players protecting their rankings
- Time between matches not factored
- New players overvalued
- Inflation/deflation of rankings due to people with less than average/higher than average ratings retiring/joining → problem for **comparing historical data** with current
- Only two ratings updated at a time

...**but that's okay**, because Arpad Elo created the algorithm specifically so that every chess player could easily calculate their rating with pen and paper.

In computer times, we can do more!

bayes

4. Glicko



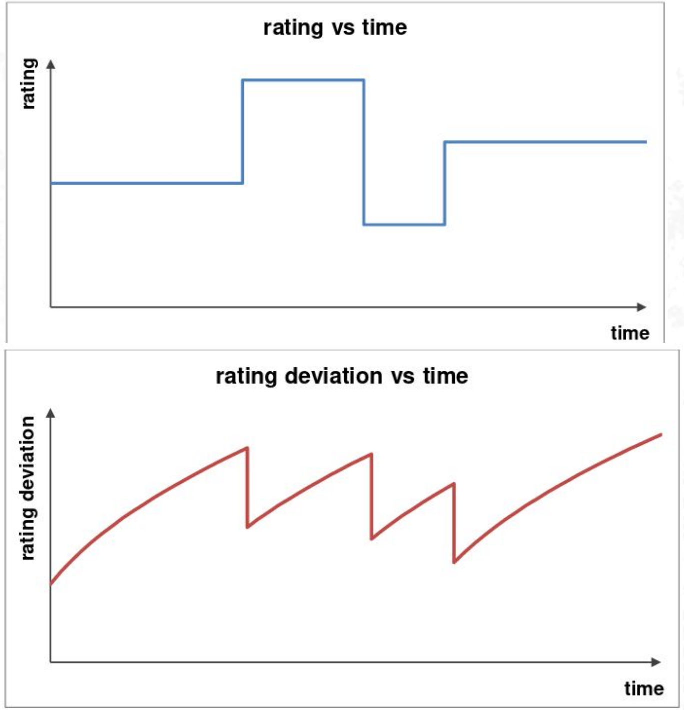
http://www.datdota.com/

Glicko-2

- Glicko developed 1995 by Mark Glickman as an improvement of the Elo
- **Glicko-2** improves on Glicko, released 2012
- http://www.glicko.net/glicko.html
- Implemented in Pokémon Showdown, CS:GO, TF2, Go, Chess.com...
- Key ideas:
  - Player skill described by **rating** and **rating deviation** (RD)
  - RD decreases with match results and **increases** during inactivity
  - RD depends on **volatility**, which measures inconsistency in player performance
  - Skill is given by a **confidence interval**: Player with a rating of 1500 and RD 50 has a real strength between 1400 and 1600 (two std from 1500) with 95% confidence
  - **Rating periods**: Matches played within one rating period are assumed to have occurred simultaneously to assure same uncertainty, ca. 10-15 matches

bayes

Glicko(-2)



Glicko 1:

- RD decays with fixed speed **c**
- Decay speed is tuning parameter

$$c = \sqrt{\frac{RD_{UNR}^2 - RD_{NOM}^2}{t}}$$

Glicko 2:

- RD is a function of **volatility** σ ('degree of rating fluctuation') updated after each rating period
- Volatility change over time is **constrained** by τ , which is a tuning parameter

Images source: https://www.englishchess.org.uk/wp-content/uploads/2012/04/The_Glicko_system_for_beginners1.pdf

Glicko-2

Before: $R'_A = R_A + K(S_A - E_A).$

Now: $$R'_A = R_A + \underbrace{\left(\sqrt{\frac{1}{\phi_A^2 + \sigma'^2} + \frac{1}{v}}\right)^{-1}}_{\phi'_A} \sum_{j=1}^{m} g(\phi_j)(S_j - E(S_j | R_A, R_j, \phi_j))$$

Glicko-2

Before: $R'_A = R_A + K(S_A - E_A).$

Now: $$R'_A = R_A + \underbrace{\left(\sqrt{\frac{1}{\phi_A^2 + \sigma'^2} + \frac{1}{v}}\right)^{-1}}_{\phi'_A} \sum_{j=1}^{m} g(\phi_j)(S_j - E(S_j|R_A, R_j, \phi_j))$$

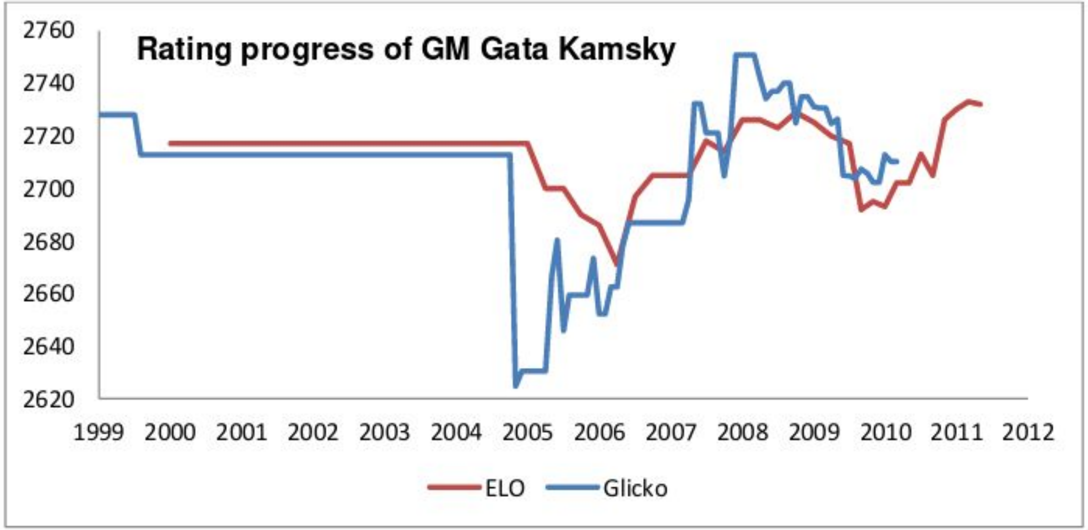
$$R_{new} = R_{old} + K \sum g(S - E)$$

- K is a function of current rating and RD ϕ_A and every opponent's rating and RD, with own RD having most influence
- g < 1 depends on the opponent's RD and **weights the importance of the result** against that opponent
- σ (volatility) used to update ϕ_A
- **Rating period should have 5-10 games per player on average**
- **Equal RD leads to equal point loss/gain after match, different RD does not**

bayes

Glicko-2

GM Gata Kamsky returning to chess after many years of inactivity

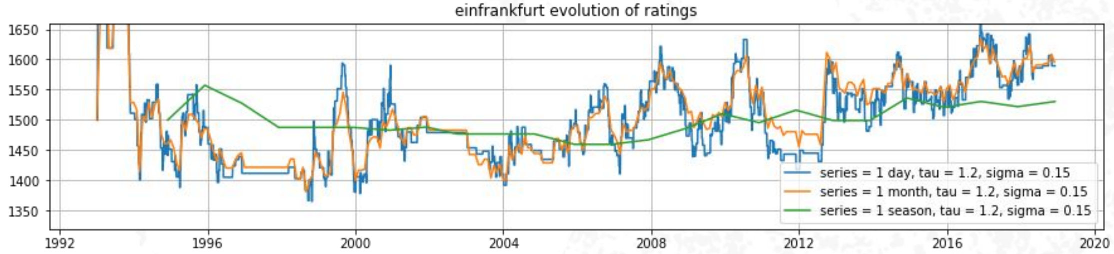


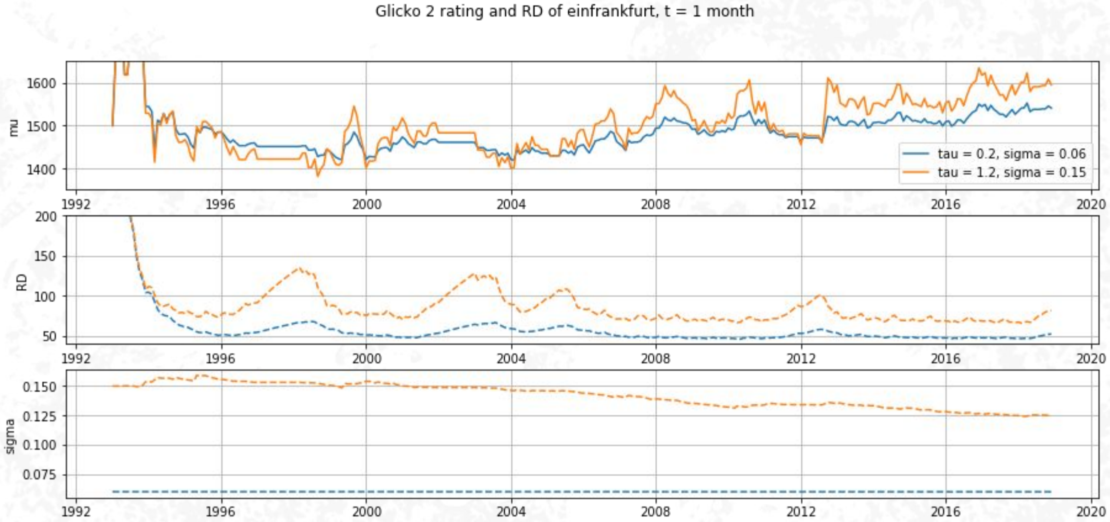
Source: https://www.englishchess.org.uk/wp-content/uploads/2012/04/The_Glicko_system_for_beginners1.pdf

bayes

Glicko 2

```python
MU = 1500
PHI = 350
SIGMA = 0.06
TAU = 1.0
EPSILON = 0.000001
#: A constant which is used to standardize the logistic function to
#: `1/(1+exp(-x))` from `1/(1+10^(-r/400))`
Q = math.log(10) / 400
```





Glicko 2 python implementation available from Heungsub Lee, **github.com/sublee/glicko2**

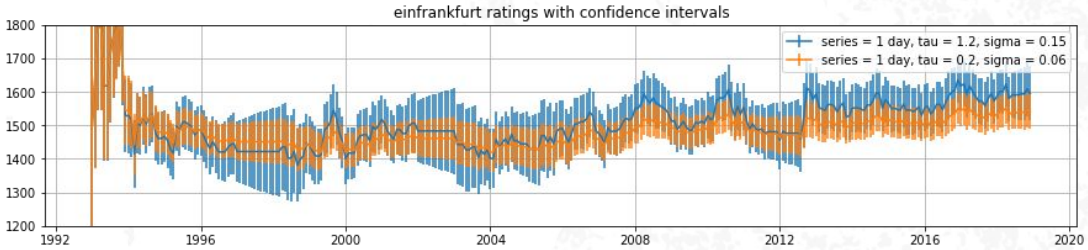
Glicko 2 python implementation available from Heungsub Lee, **github.com/sublee/glicko2**

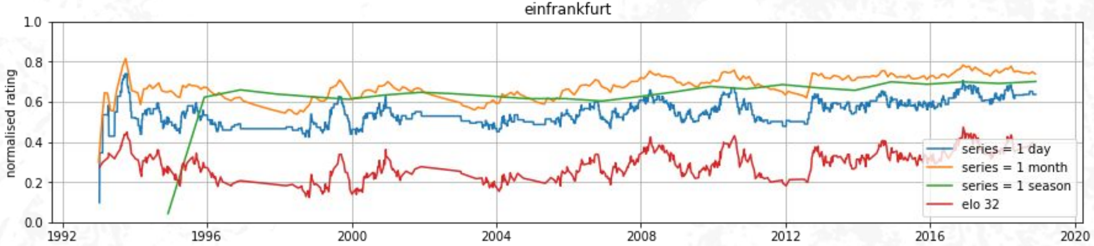
bayes

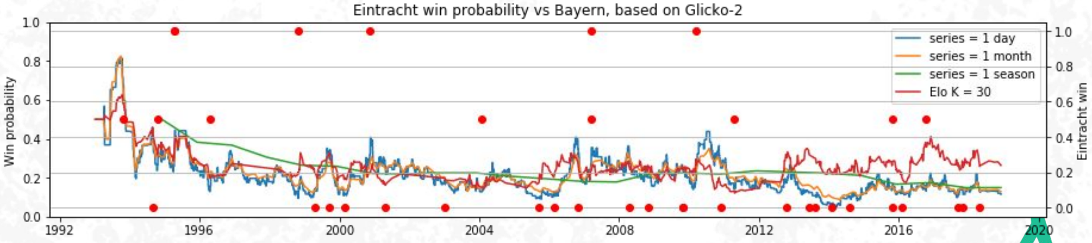
Glicko 2

```python
MU = 1500
PHI = 350
SIGMA = 0.06
TAU = 1.0
EPSILON = 0.000001
#: A constant which is used to standardize the logistic function to
#: `1/(1+exp(-x))` from `1/(1+10^(-r/400))`
Q = math.log(10) / 400
```

Glicko 2 python implementation available from Heungsub Lee, **github.com/sublee/glicko2**







Summary

Key ideas:

- Rating becomes more uncertain over time → **Rating deviation**
- Rating deviation is a function of performance **volatility**
- **Rating period** of several games

Tuning:

- Choose rating period
- Choose initial values (default sigma 1500, RD 350)
- (Only Glicko-1) choose c: how long it should take for a rating to decay
- (Only Glicko-2) choose sigma: base volatility (default 0.06) and tau: volatility change constraint (default between 0.3 and 1.2) based on expected number of upsets

bayes

Known issues

- Ratings only valid at the time of their computation
- Only provisional ratings available during rating period
- Impact of one match on the rating not transparent
- Hard to explain

bayes

5. TrueSkill

TrueSkill

Solves a different problem: Find ad-hoc balanced matches → **maximize draw probability**

- Initially developed for matchmaking in Halo
- => **need to estimate team skill from player skill**

- Assume **player skill is normally distributed**

- Team is composed of several players and **team skill is the joint distribution**

- After the match **update posterior distribution of all players**

A python implementation was written by Heungsub Lee, **https://trueskill.org/**

bayes

TrueSkill

Benefits:

- **Can update after each match**
- **Fast** convergence for new players
- Tracks player skill in shifting teams
- Solutions for partial play,

  multi-team games, etc.

Shortcomings:

- **Proprietary** algorithm by Microsoft, patented and trademarked
- Player-based: requires all players to have played a min amount of games
- Prediction requires prior knowledge of the team lineup
- Difficult to tune model parameters

Easier to model a team based on their **historic performance together**

bayes

Summary

- Rating algorithms need to be **tuned**
- Rating algorithms need **time to converge**
- Rating algorithms can be used to predict match outcome
- **There is no “best”**
- **Elo is often good enough**
- Glicko-2 is good when **players don’t play regularly**
- Trueskill is good when teams are created ad-hoc
- Domain knowledge > fancy algorithm

bayes

Thank you!

For code please go to: github.com/drdarina/ratingsl_talk

Sources

Mark Glickman, “Example of the Glicko-2 system”, 2013 http://www.glicko.net/glicko/glicko2.pdf

Ralf Herbrich, Tom Minka, Thore Graepel, “Trueskill™: A Bayesian Skill Rating System” https://papers.nips.cc/paper/3079-trueskilltm-a-bayesian-skill-rating-system.pdf

Tom Minka, Ryan Cleven, Yordan Zaykov, “TrueSkill2: An improved Bayesian skill rating system” https://www.microsoft.com/en-us/research/uploads/prod/2018/03/trueskill2.pdf

Fide chess implementation of Elo: https://www.fide.com/fide/handbook.html?id=172&view=article

Jeff Moser, “Computing Your Skill” http://www.moserware.com/2010/03/computing-your-skill.html

Michalis Kaloumenos, “The Glicko System for Beginners” https://www.englishchess.org.uk/wp-content/uploads/2012/04/The_Glicko_system_for_beginners1.pdf

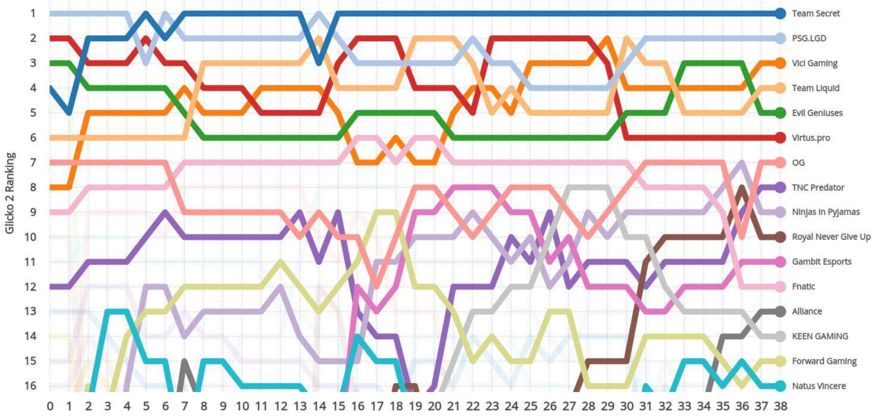
Deloitte/FIDE Chess Rating Challenge, https://www.kaggle.com/c/ChessRatings2/

Huge thank you to
Ben “noxville” Steenhuisen
https://www.datdota.com
https://twitter.com/follownoxville

bayes

Bayes Brier Scores

- Dota 2 professional matches since 2017
- Only matches with RD < 150 considered for Glicko
- Rating period 1 week

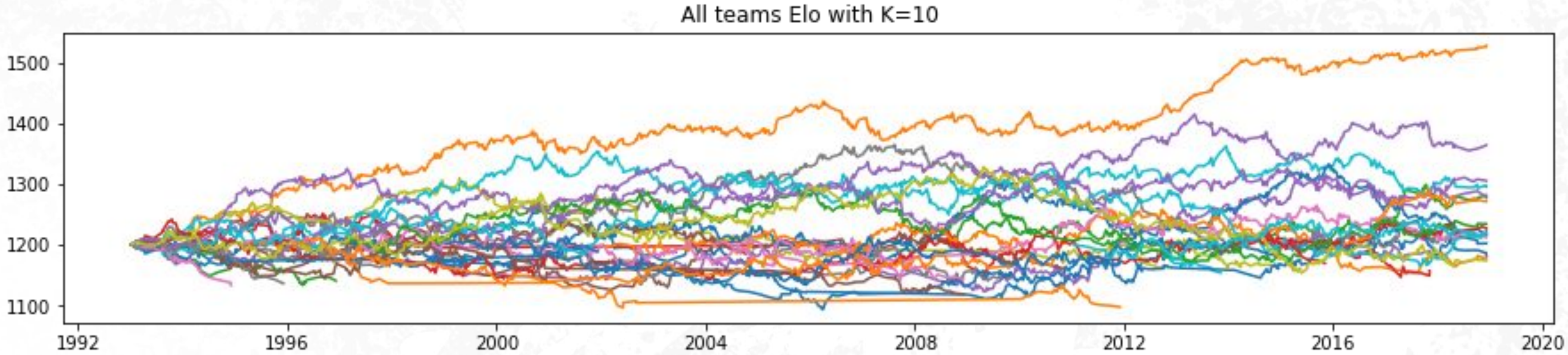


algorithm	correct_pred	wrong_pred	Brier
elo32	5094	3333	0.2330
elo64	5029	3398	0.2508
glicko1	2065	1284	0.2340
glicko2	3671	2202	0.2281
homebrewn	2169	1178	0.2139

bayes

Alternatives

- WHR/HIST: uses entire history of play
- Ree/Rankade: proprietary, player-based
- Least squares ratings: regression on score difference
- Network-based models: opponent's wins propagate through a network
- Markovian methods: transition of fans between teams


All teams Elo with K=10
1500
1400
1300
1200
1100
1992
1996
2000
2004
2008
2012
2016
2020

bayes

Glicko 1->2 conversion

- Adds rating volatility sigma: degree of expected fluctuation, incorporated in the RD value. Default: 0.06
- 173.7178 = 400 / ln(10) -> scales down the rating
- 10-15 games within rating period
- Conversion from glicko 1:

$$\begin{aligned}\mu &= (r - 1500)/173.7178 \\ \phi &= \mathrm{RD}/173.7178\end{aligned}$$

- Innovation: track players' abilities who improve more quickly than the rating system can handle

Step 6. Update the rating deviation to the new pre-rating period value, ϕ^*:

$$\phi^* = \sqrt{\phi^2 + \sigma'^2}$$

Step 7. Update the rating and RD to the new values, μ' and ϕ':

$$\begin{aligned}\phi' &= 1\Big/\sqrt{\frac{1}{\phi^{*2}} + \frac{1}{v}} \\ \mu' &= \mu + \phi'^2 \sum_{j=1}^{m} g(\phi_j)\{s_j - \mathrm{E}(\mu, \mu_j, \phi_j)\}\end{aligned}$$

Step 8. Convert ratings and RD's back to original scale:

$$\begin{aligned}r' &= 173.7178\mu' + 1500 \\ \mathrm{RD}' &= 173.7178\phi'\end{aligned}$$

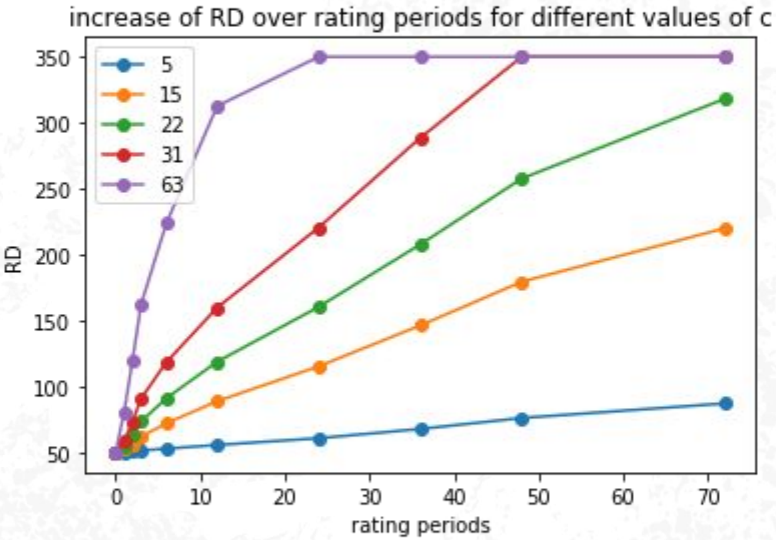
Note that if a player does not compete during the rating period, then only Step 6 applies. In this case, the player's rating and volatility parameters remain the same, but the RD increases according to

$$\phi' = \phi^* = \sqrt{\phi^2 + \sigma^2}.$$

bayes

Glicko 1 tuning

- Choose rating period
- Choose c value: how long it takes for the RD to increase back to default (Chess: ~10 years)
- Since higher uncertainty leads to higher potential loss/gain of rating, playing regularly helps you protect your rating!



bayes

Glicko: RD influence on expected outcome

- RD: confidence in the rating
- Outcome probability depends on RD as well as R
- Outcome probability also depends on opponents' RD
- Higher rated player is **expected to perform worse** if the opponent's rating is considered unreliable (lower rated is expected to perform better if opponent's rating is unreliable)
- The amount rating changes after the game depends on the RD, the smaller the RD the higher confidence in the rating, the less change occurs, limsum(RD) = 350

R_{old}	R_{opp}	RD_{opp}	E
2400	2330	50	0.598
2200	2130	50	0.598
1900	1830	50	0.598
1600	1530	50	0.598
2400	2330	100	0.595
2200	2130	100	0.595
1900	1830	100	0.595
1600	1530	100	0.595
2400	2330	200	0.584
2200	2130	200	0.584
1900	1830	200	0.584
1600	1530	200	0.584

RD_{opp}	E
50	0.402
100	0.405
200	0.416

bayes

Source: https://www.englishchess.org.uk/wp-content/uploads/2012/04/The_Glicko_system_for_beginners1.pdf