

Previsão de resultado de partidas de Valorant utilizando técnicas de Machine Learning

João Pedro Darabas Cardoso

Objetivo

Criar um modelo capaz de obter
um bom desempenho prevendo
resultados de partidas futuras

Dataset

- match_id: identificador da partida
- date: data
- best_of: melhor de 3 ou 5
- tier: ‘S’, ‘A’, ou ‘B’
- team_A e team_B: nomes dos times
- A_score: placar do time A
- B_score: placar do time B
- team_X_player_Y: nome dos jogadores
- win: 0 para vitória do time A, 1 para time B

#	Column	Non-Null Count	Dtype
0	match_id	3246 non-null	int64
1	date	3246 non-null	datetime64[ns]
2	best_of	3246 non-null	int64
3	tournament	3246 non-null	object
4	tier	3246 non-null	object
5	team_A	3246 non-null	object
6	team_B	3246 non-null	object
7	A_score	3246 non-null	int64
8	B_score	3246 non-null	int64
9	team_A_player_1	3246 non-null	object
10	team_A_player_2	3246 non-null	object
11	team_A_player_3	3246 non-null	object
12	team_A_player_4	3246 non-null	object
13	team_A_player_5	3246 non-null	object
14	team_B_player_1	3246 non-null	object
15	team_B_player_2	3246 non-null	object
16	team_B_player_3	3246 non-null	object
17	team_B_player_4	3246 non-null	object
18	team_B_player_5	3246 non-null	object
19	win	3246 non-null	int64
dtypes: datetime64[ns](1), int64(5), object(14)			

Seleção e Preparação de dados

Aplicação do algoritmo Glicko-2

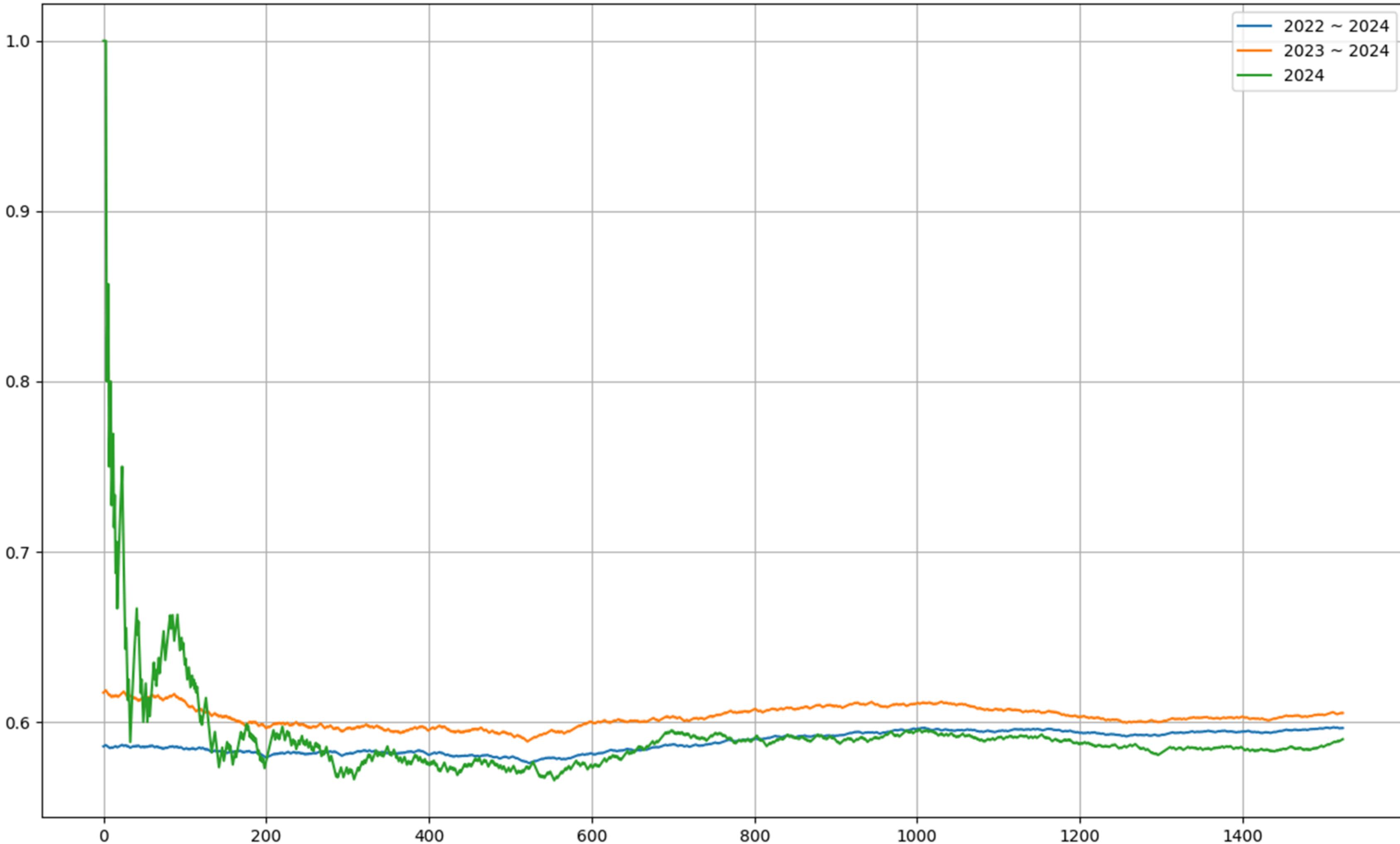
O algoritmo define um rating para os times, um RD (Rating Deviation) e um valor sigma que equivale a “volatilidade” do rating

O Algoritmo foi aplicado nos times e nos jogadores com base nas vitórias nas partidas, foram adicionadas features a partir da diferença do rating e uma classe para definir se a diferença está fora do RD ou não.

Histórico de últimos torneios

Foram adicionadas features com base na performance do torneio atual e do último. Como porcentagem de vitórias e diferença de rating do time entre o começo e o final do torneio.

Acurácia Glicko em 2024



Selecionando Modelo

O modelo RidgeClassifier da biblioteca Scikit-Learn se mostrou como mais eficiente entre os demais, com acurácia média de 61.84% usando os melhores parâmetros do grid_search

```
1 X, y = select_data(X_2)
2 X, encoder = encode_X(X)
3 X, scaler = scale_X(X)
4 print("SVC:")
5 svc_1 = svc_grid(X,y)
6 print("RF:")
7 RF_1 = RF_grid(X,y)
8 print("XGB:")
9 XGB_1 = XGB_grid(X,y)
10 print("RIDGE:")
11 ridge_1 = ridge_grid(X,y)

→ SVC:
Fitting 5 folds for each of 3 candidates, totalling 15 fits
{'C': 1.75, 'gamma': 1}
0.575875334573334

RF:
Fitting 5 folds for each of 2 candidates, totalling 10 fits
{'criterion': 'gini', 'max_depth': 4, 'max_features': 'sqrt', 'n_estimators': 1500}
0.6133729528648549

XGB:
Fitting 5 folds for each of 72 candidates, totalling 360 fits
{'colsample_bytree': 0.7, 'gamma': 0.0, 'learning_rate': 0.01, 'max_depth': 3, 'n_estimators': 100}
0.6099786780383795

RIDGE:
Fitting 5 folds for each of 2 candidates, totalling 10 fits
{'alpha': 1.215, 'fit_intercept': True}
0.6184974821938937
```

RidgeClassifier:

- Converte os valores de y em $\{-1, 1\}$ e então trata a tarefa como uma regressão;
- Usa um termo de regularização adicionado à função de custo para reduzir overfitting;
- O parâmetro `solver` define qual algoritmo o modelo irá usar para resolver a regressão (por padrão ele escolhe automaticamente com base nos dados de X);
- O parâmetro `Alfa` altera o termo de regularização: para $Alfa = 0$, o modelo realiza uma simples regressão linear, para valores altos o modelo tende a por todos os pesos próximos a 0.

API

[GET] /teams

Retorna os times

[GET] /players

Retorna os players

[POST] /predict

(body: JSON)

{

“team_A”: “time A”,

“team_B”: “time B”

}

Retorna

{ “prediction”: 0 } para vitória do

time A

{ “prediction”: 1 } para vitória do

time B

Previsão de Resultados de Valorant

Ranking

Time	Pontuação
Gen.G Esports	1884.0045194651448
Paper Rex	1843.488063439974
EDward Gaming	1798.0025719081505
FunPlus Phoenix	1774.4427517949966
Sentinels	1759.4221218014973
Team Heretics	1756.7566123351019
DRX	1721.0358038703785
100 Thieves	1718.2378579790295
Natus Vincere	1695.8436325048342
FUT Esports	1635.1657353440266

[Adicionar partida](#)

Previsão de Resultados de Valorant

Adicionar partida

Time 1 Time 2

Time 1 — Time 2 —

100 Thieves LOUD

Adicionar

Team Heretics 1756.7566123351019

DRX 1721.0358038703785

100 Thieves 1718.2378579790295

Natus Vincere 1695.8436325048342

FUT Esports 1635.1657353440266

Previsão de Resultados de Valorant

Adicionar partida

100 Thieves X LOUD

Ranking

Time	Pontuação
Gen.G Esports	1884.0045194651448
Paper Rex	1843.488063439974
EDward Gaming	1798.0025719081505
FunPlus Phoenix	1774.4427517949966
Sentinels	1759.4221218014973
Team Heretics	1756.7566123351019
DRX	1721.0358038703785
100 Thieves	1718.2378579790295
Natus Vincere	1695.8436325048342
FUT Esports	1635.1657353440266