Hide code

```
1 battles
```

```
def compute_mle_elo(df, SCALE=400, BASE=10, INIT_RATING=1000):
                                                                             2
        from sklearn.linear_model import LogisticRegression
3
        models = pd.concat([df["model a"], df["model b"]]).unique()
4
        models = pd.Series(np.arange(len(models)), index=models)
5
        # duplicate battles
6
7
        df = pd.concat([df, df], ignore_index=True)
8
        p = len(models.index)
9
        n = df.shape[0]
10
11
        X = np.zeros([n, p])
12
        X[np.arange(n), models[df["model_a"]]] = +math.log(BASE)
        X[np.arange(n), models[df["model_b"]]] = -math.log(BASE)
13
14
15
        # one A win => two A win
16
        Y = np.zeros(n)
        Y[df["winner"] == "model_a"] = 1.0
17
18
19
        # one tie => one A win + one B win
20
        # find tie + tie (both bad) index
21
        tie idx = (df["winner"] == "tie") | (df["winner"] == "tie")
    (bothbad)")
22
        tie_idx[len(tie_idx)//2:] = False
23
        Y[tie_idx] = 1.0
24
25
        lr = LogisticRegression(fit_intercept=False, penalty=None, tol=1e-
    8)
26
        lr.fit(X,Y)
27
28
        elo scores = SCALE * lr.coef [0] + INIT RATING
29
30
        # set anchor as mixtral = 1114
31 ,,
        if "mixtral-8x7b-instruct-v0.1" in models.index:
32
            elo scores += 1114 - elo scores[models["mixtral-8x7b-instruct-
    v0.1"]]
33
        return pd.Series(elo_scores, index =
    models.index).sort_values(ascending=False)
```

```
def get_bootstrap_result(battles, func_compute_elo, num_round):
    rows = []
    for i in tqdm(range(num_round), desc="bootstrap"):
        rows.append(func_compute_elo(battles.sample(frac=1.0, replace=True)))
    df = pd.DataFrame(rows)
    return df[df.median().sort_values(ascending=False).index]
```

```
def compute_online_elo(battles, K=4, SCALE=400, BASE=10,
INIT_RATING=1000):
```



```
2
        rating = defaultdict(lambda: INIT_RATING)
3
4 ^
        for rd, model_a, model_b, winner in battles[['model_a',
    'model_b', 'winner']].itertuples():
5
            ra = rating[model_a]
6
            rb = rating[model_b]
7
            ea = 1 / (1 + BASE ** ((rb - ra) / SCALE))
8
            eb = 1 / (1 + BASE ** ((ra - rb) / SCALE))
9 v
            if winner == "model_a":
10
                sa = 1
11 '
            elif winner == "model_b":
12
                 sa = 0
13 V
            elif winner == "tie" or winner == "tie (bothbad)":
14
                sa = 0.5
15 V
            else:
16
                raise Exception(f"unexpected vote {winner}")
            rating[model_a] += K * (sa - ea)
17
18
            rating[model_b] += K * (1 - sa - eb)
19
20
        # calibrate llama-13b to 800
21
        delta = (800-rating["llama-13b"])
22 V
        for model in battles["model_a"].unique():
23
            rating[model] += delta
24
25
        return rating
```

```
Model Elo rating
                                            1256.17
   1
        ### Model A: claude-3-opus-20240229
   2
        ### Model B: claude-3-opus-20240229
                                            1226.26
     ### Model B: claude-3-sonnet-20240229
                                            1181.77
         ### Model B: gpt-4-turbo-2024-04-09
                                            1176.08
   5
                   claude-3-opus-20240229
                                            1161.58
                                                 ...
 224
                  ### Model B: gemma-2b-it
                                             829.69
 225
                  ### Model B: gemma-7b-it
                                             822.82
 226
                  ### Model A: gemma-2b-it
                                             819.47
 227
                             dolly-v2-12b
                                             811.64
 228
                                llama-13b
                                             800.00
228 rows × 2 columns
1 ,
   def preety_print_model_ratings(ratings):
2 ,
        df = pd.DataFrame([
3
             [n, ratings[n]] for n in ratings.keys()
4
        ], columns=["Model", "Elo rating"]).sort_values("Elo rating",
   ascending=False).reset_index(drop=True)
5
        # df["Elo rating"] = (df["Elo rating"] + 0.5).astype(int)
        df.index = df.index + 1
6
7
        return df
8
```

```
online_elo_ratings = compute_online_elo(battles)
```



```
Model Elo rating
   1
       ### Model A: claude-3-opus-20240229
                                             1363.86
      ### Model B: claude-3-opus-20240229
                                             1360.53
        ### Model B: qpt-4-turbo-2024-04-09
                                             1340.93
        ### Model A: gpt-4-turbo-2024-04-09
                                             1339.17
      ### Model A: gpt-4-0125-assistants-api
                                             1322.65
 224
              ### Model B: gemma-1.1-2b-it
                                              868.82
225
                             fastchat-t5-3b
                                              866.37
226
                   stablelm-tuned-alpha-7b
                                              831.28
 227
                              dolly-v2-12b
                                              809.97
 228
                                llama-13b
                                              792.67
228 rows × 2 columns
1
    elo_mle_ratings = compute_mle_elo(battles)
2
    preety_print_model_ratings(elo_mle_ratings)
```

```
BOOTSTRAP_ROUNDS = 100

np.random.seed(42)
bootstrap_elo_lu = get_bootstrap_result(battles, compute_mle_elo, BOOTSTRAP_ROUNDS)
```

```
1 v def predict_win_rate(elo_ratings, SCALE=400, BASE=10,
    INIT_RATING=1000):
2
        names = sorted(list(elo_ratings.keys()))
3
        wins = defaultdict(lambda: defaultdict(lambda: 0))
4 ,
        for a in names:
5 <sub>v</sub>
            for b in names:
6
                 ea = 1 / (1 + BASE ** ((elo_ratings[b] - elo_ratings[a])
    / SCALE))
7
                 wins[a][b] = ea
                 wins[b][a] = 1 - ea
8
9
10
        data = {
11
            a: [wins[a][b] if a != b else np.NAN for b in names]
12
            for a in names
13
        }
14
15
        df = pd.DataFrame(data, index=names)
16
        df.index.name = "model_a"
17
        df.columns.name = "model b"
18
```

```
### Model A: claude-3-opus-20240229
                                                                                 ##
                                                                                     ##
                                                                                                           Model B:
                                                                                                                                                   Model
                                                                                                                                                       Model
                                                                                                                                               Model B:
                                                                       Model B: claude-3-opus-20240229
                                                                                Model B: gpt-4-turbo-2024-04-09
                                                                                         ## Model B: gpt-4-0125-preview
                                                                                                      Model
                                                                            Model B: llama-3-70b-chat-hf
                                                                                                                                                   ₽
                                                                                    A: gpt-4-turbo-2024-04-09
                                                                                                           claude-3-sonnet-20240229
                                                                                                               claude-3-sonnet-20240229
                                                                                                                                              gpt-4-0125-assistants-api
                                                                                                                                                   gpt-4-0125-assistants-ap
                                                                                                                                                       claude-3-haiku-20240307
                                                                                                                    claude-3-opus-20240229
                                                                                                                        gpt-4-turbo-2024-04-09
                                                                                                      Ð.
                                                                                                  Φ...
                                                                                                                                 Model B: gpt-4-0314
                                                                                                                                     Model A: gpt-4-0314
                                                                                                  gpt-4-1106-preview
                                                                                                      gpt-4-1106-preview
                                                                                             gpt-4-0125-preview
                                                                                                                             gpt-4-1106-preview
                                                                                                                                          gpt-4-0125-preview
  ### Model A: claude-3-opus-20240229
                                                                       0.50 0.51 0.52 0.53 0.58 0.58 0.58 0.59 0.62 0.62 0.64 0.65 0.65 0.65 0.66 0.66 0.66 0.66 0.68 0.6
                                                                           0.51 0.52 0.53 0.57 0.57 0.58 0.59 0.61 0.62 0.64 0.65 0.65 0.65 0.65 0.65 0.66 0.66 0.68 0.6
  ### Model B: claude-3-opus-20240229 0.50
           ### Model B: Ilama-3-70b-chat-hf 0.49 0.49
                                                                                0.51 0.52 0.56 0.56 0.57 0.58 0.60 0.61 0.63 0.63 0.64 0.64 0.64 0.64 0.65 0.65 0.67 0.
    ### Model B: gpt-4-turbo-2024-04-09 0.48 0.48 0.49 0
                                                                                    0.51 0.55 0.55 0.56 0.57 0.59 0.60 0.62 0.63 0.63 0.63 0.63 0.63 0.64 0.64 0.66 0.6
    ### Model A: qpt-4-turbo-2024-04-09 0.47 0.47 0.48 0.49
                                                                                        0.54 0.54 0.55 0.56 0.58 0.59 0.61 0.62 0.62 0.62 0.62 0.63 0.63 0.63 0.65 0.6
           ### Model B: gpt-4-0125-preview 0.42 0.43 0.44 0.45 0.46
                                                                                            0.50 0.51 0.52 0.54 0.55 0.57 0.57 0.58 0.58 0.58 0.59 0.59 0.59 0.61 0.
           ### Model A: gpt-4-0125-preview 0.42 0.43 0.44 0.45 0.46 0.50 0
                                                                                                  0.51 0.52 0.54 0.55 0.57 0.57 0.58 0.58 0.58 0.58 0.59 0.59 0.61 0.0
           ### Model B: gpt-4-1106-preview 0.42 0.42 0.43 0.44 0.45 0.49 0.49 0.49 0.51 0.53 0.54 0.56 0.57 0.57 0.57 0.58 0.58 0.58 0.58 0.58 0.60 0.6
           ### Model A: gpt-4-1106-preview 0.41 0.42 0.43 0.44 0.48 0.49 0.00 0.52 0.53 0.55 0.56 0.56 0.56 0.57 0.57 0.57 0.57 0.57 0.59 0.6
### Model B: claude-3-sonnet-20240229 0.38 0.39 0.40 0.41 0.42 0.46 0.47 0.48 0.50 0.51 0.53 0.53 0.54 0.54 0.54 0.55 0.55 0.57 0.59
### Model A: claude-3-sonnet-20240229 0.38 0.38 0.39 0.40 0.41 0.45 0.45 0.46 0.47 0.49 0
                          claude-3-opus-20240229 0.36 0.36 0.37 0.38 0.39 0.43 0.43 0.44 0.45 0.47 0.48 0.30
                            qpt-4-turbo-2024-04-09 0.35 0.35 0.37 0.37 0.38 0.43 0.43 0.43 0.44 0.47 0.47 0.49 0
                                                                                                                            0 0.51 0.51 0.51 0.51 0.51 0.51 0.54 0.55
                                   gpt-4-1106-preview 0.35 0.35 0.36 0.37 0.38 0.42 0.42 0.43 0.44 0.46 0.47 0.49 0.49 0.49
                         ### Model B: qpt-4-0314 0.35 0.35 0.36 0.37 0.38 0.42 0.42 0.43 0.44 0.46 0.47 0.49 0.49 0.50 0.50 0.50 0.50 0.51 0.51 0.53 0.56
                         gpt-4-0125-preview 0.34 0.35 0.36 0.37 0.37 0.41 0.42 0.42 0.43 0.46 0.46 0.48 0.49 0.49 0.50 0.50 0.50 0.50 0.50 0.53 0.54
 ### Model B: gpt-4-0125-assistants-api 0.34 0.34 0.35 0.36 0.37 0.41 0.41 0.42 0.43 0.45 0.46 0.48 0.49 0.49 0.49 0.50 0.50
 ### Model A: gpt-4-0125-assistants-api 0.34 0.34 0.34 0.35 0.36 0.37 0.41 0.42 0.43 0.45 0.46 0.48 0.49 0.49 0.49 0.50 0.50 0.50
  ### Model A: claude-3-haiku-20240307 0.32 0.32 0.32 0.33 0.34 0.35 0.39 0.39 0.40 0.41 0.43 0.44 0.45 0.46 0.47 0.47 0.47 0.47 0.48 0.48
                             ### Model B: claude-1 0.31 0.31 0.32 0.33 0.34 0.38 0.38 0.39 0.40 0.42 0.43 0.45 0.45 0.46 0.46 0.46 0.46 0.47 0.47 0.49
  ### Model B: claude-3-haiku-20240307 0.31 0.31 0.32 0.33 0.34 0.38 0.38 0.39 0.40 0.42 0.43 0.44 0.45 0.46 0.46 0.46 0.46 0.46 0.49 0.50
     ### Model A: bard-jan-24-gemini-pro 0.30 0.30 0.31 0.32 0.33 0.37 0.37 0.38 0.39 0.41 0.42 0.44 0.44 0.45 0.45 0.45 0.45 0.46 0.46 0.48 0.46
```

Predicted Win Rate Using Elo Ratings for Model A in an A

Model A: gpt-4-0613 0.29 0.29 0.30 0.31 0.32 0.36 0.36 0.37 0.40 0.40 0.42 0.43 0.43 0.44 0.44 0.44 0.44 0.44 0.47 0.48

```
win_rate = predict_win_rate(dict(bootstrap_elo_lu.quantile(0.5)))
ordered_models =
win_rate.mean(axis=1).sort_values(ascending=False).index
ordered_models = ordered_models[:30]
```

Model A

```
4 v fig = px.imshow(win_rate.loc[ordered_models, ordered_models],
5
                     color_continuous_scale='RdBu', text_auto=".2f",
6
                     title="Predicted Win Rate Using Elo Ratings for Model
    A in an A vs. B Battle")
7 v fig.update_layout(xaxis_title="Model B",
8
                       yaxis_title="Model A",
9
                       xaxis_side="top", height=900, width=900,
                       title_y=0.07, title_x=0.5)
10
11 v fig.update_traces(hovertemplate=
12
                       "Model A: %{y}<br>Model B: <math>%{x}<br>Win Rate: <math>%{z}
    <extra></extra>")
13
    fig
```

```
from collections import defaultdict
import json, math, gdown
import numpy as np
import pandas as pd
import plotly.express as px
import sklearn
from tqdm import tqdm
import requests
pd.options.display.float_format = '{:.2f}'.format
```

```
url =
  "https://storage.googleapis.com/arena_external_data/public/clean_battl
  e_20240419.json"

response = requests.get(url)

with open('local_file_name.json', 'wb') as file:
  file.write(response.content)

# load the JSON data from the local file
with open('local_file_name.json', 'r') as file:
  battles = pd.read_json(file).sort_values(ascending=True, by=
["tstamp"])
```

```
1 | import marimo as mo
```