Vizualizace dat z Chess.com

Adam Hoštička

ČVUT–FIT

hostiada@fit.cvut.cz

8. ledna 2023

1 Úvod

Semestrální práce je na téma vizualizace dat z Chess.com. Mým cílem tedy bylo nejdříve dostat z Chess.com API data o hráčích online šachů, o šachových partiích, a poté je vizualizovat. K získání dat jsem použil knihovnu requests. Na vizualizaci jsem použil knihovnu dash a plotly express.

2 Získání dat z Chess.com API

Když jsem si zvolil téma vizualizace šachových dat, tak jsem se snažil najít nějaký vhodný dataset. Chtěl jsem vizualizovat jak nějaké statistiky o hráčích, tak i statistiky z šachových partií. Jenže jsem nikde nenašel dobré hráčské datasety. Pak jsem objevil Chess.com API[3]...

2.1 Chess.com API

Chess.com má jednoduché API, z kterého jsem použil následující endpointy:

- Country Profile - Pomocí ISO kódů (hard-coded) získávám informace o zemích z Chess.com a uložím je do souboru countries.pkl.
- Country Players - Pomocí ISO kódů již získaných z API si pro každou zemi uložím základní informace o hráčích z dané země. Endpoint nabízí pouze uživatelská jména, maximálně jeden tisíc. Do souboru players.pkl si tedy uložím záznamy o uživatelských jménech společně s tím, z jaké jsou země.
- Player profile - Pomocí již uložených uživatelských jmen z endpointu dostaneme dodatečné informace o hráčích, týkající se jejich profilu na Chess.com.
- Player stats - Pomocí již uložených uživatelských jmen z endpointu dostaneme dodatečné informace o hráčích, týkající se jejich herních statistik.
- Complete monthly archives - Pro daného hráče z endpointu dostaneme všechny jeho dokončené hry za daný měsíc a rok. V semestrální práci jsem pracoval s daty pouze z prosince roku 2022.

2.2 Funkčnost

Snažil jsem se, aby i fetching dat byl uživatelsky přívětivý. Funkčnost je tedy rozdělena do několik tříd a všechno se spouští z jednoho souboru 2.3. Jednotlivé třídy se nachází v app.src.fetcht_data a jsou jimi:

- fetch_country: Volání Country Profile, data se přemažou.
- fetch_country_player: Volání Country Players, data se sjednotí.
- fetch_player: Volání zbývajících dvou Player endpointů, data se sjednotí. Oba endpointy Player profile a Player stats se volají v této metodě.
- fetch_game: Volání Complete monthly archives, data se sjednotí.

Protože jsem chtěl předejít problémům s rate limitem, raději jsem posílal requesty po jednom. Je to ale za cenu toho, že jeden hráč či jedna hra nás stojí přibližně jednu sekundu.

2.3 Spuštění

V kořenovém adresáři je soubor fetch_data.py, který se z příkazové řádky spustí příkazem 'python3 ./fetch_data.py'. Použijte přepínač -h pro výpis argumentů. Je zde argument –fetch-all-and-reformat, který spustí všechny skripty. Kdybychom třeba poté chtěli informace o více hráčích, stačí spustit soubor s přepínačem -FP=5, což nám uloží informace o dalších 500 hráčích. Přepínač -FG=5 nezíská informace o pěti stech hrách, ale zavolá Complete monthly archives endpoint pro pět set hráčů, takže her můžeme očekávat násobně více.
Pokud chcete mít stejná data jako já, stáhněte si je z mého Google Drive `https://drive.google.com/drive/folders/1dgBMUqg1FfT6FbSC7lRpOqctkeQG6Yrf?usp=sharing`. Ponechal jsem zde i soubor games.pkl, abyste si mohli tato data případně rozřířit.

2.4 Přeformátování dat

Přestože API posílá data v pro nás celkem vhodném formátu, musíme data přeformátovat. Pro hráče to znamená vesměs pouze smazání pár parametrů. Pro hry už je toho víc. Slouží na to funkce reformat_games v app.src.format_data.reformat_games.py. Při implementování jsem se inspiroval z jedné práce z kaggle.com[1]. Některé metody jsem rozšířil, jiné jsem zcela zkopíroval. Další metody jsem již implementoval sám.

2.5 Místo na zlepšení

Do budoucna by se mohlo z příkazové řádky ovládat, po kolika se budou načítat hráči a hry (zatím je to po stovkách), také z jakého měsíce a roku hry budou, či z jaké země chceme hráče (zatím je to náhodně). Requesty by se mohly posílat asynchroně. Kód by šel také lépe rozčlenit, což by umožnilo důkladnější testování.

3 Vizualizace dat

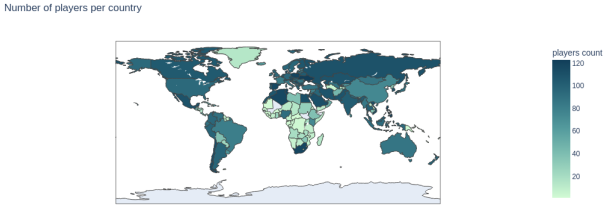
Vizualizaci jsem realizoval v Python Plotly Dash. Struktura projektu je inspirována jedním repozitářem z githubu[2]. Téma vizualizace pro mě bylo nové, a nedostatek zkušeností mě stál spoustu času. Informace jsem čerpal především z plot.ly dokumentace[5] a ze stackoverflow.com[6].

3.1 Výsledky vizualizace

3.1.1 Vizualizace hráčů

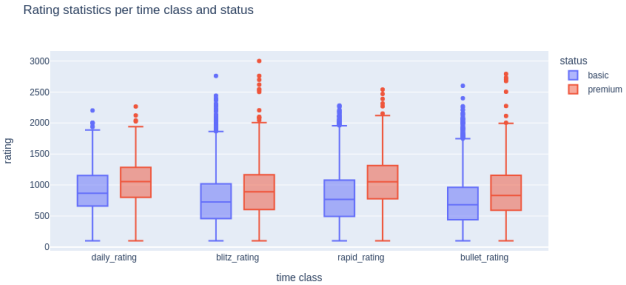
Pojmy:

- Time class = časový formát
  - Daily - Hráč má většinou jeden den na tah.
  - Rapid - Hráč má 10-30 minut na hru.
  - Blitz - Hráč má 3-5 minut na hru.
  - Bullet - Hráč má 1-2 minuty na hru.
- Status hráče - basic nebo premium.



Obrázek 1: Odkud jsou hráči z datasetu

Obrázek 2: Průměrný rating per time class



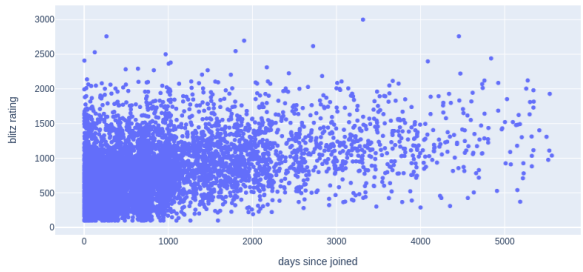
Obrázek 3: Detaily ratingu v závislosti na time class a statusu hráče

2Moje hypotéza byla, že Češi mají vyšší průměrný rating než Slováci, což se potvrdilo. Bohužel ale máme z každé země jenom okolo stovky hráčů, a tak se z toho nedají dělat žádné závěry.

3S jistotou se ukázalo, že hráči se statusem premium mají lepší rating ve všech typech her.

Obrázek 4: Závislot time class ratingu na nejvyšším ratingu v tactics

Obrázek 5: Závislot time class ratingu na nejvyšším skóre v úlohách



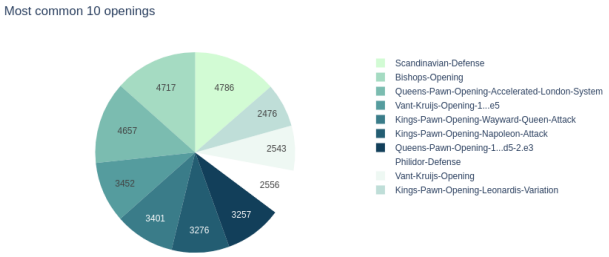
Obrázek 6: Závislot time class ratingu na době od zaregistrování

Pomocí korelačních diagramů z obrázků 4, 5, 6 jsem se snažil dokázat, že při rostoucím počtu ratingu v taktikách, úlohách a doby od registrace, roste také rating hráče. Myslím, že to v grafech trochu vidět je.

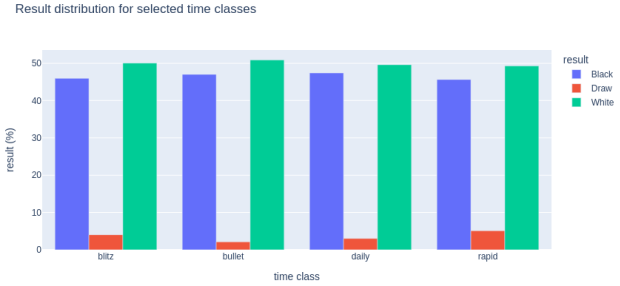
3.1.2 Vizualizace her

Pojmy:

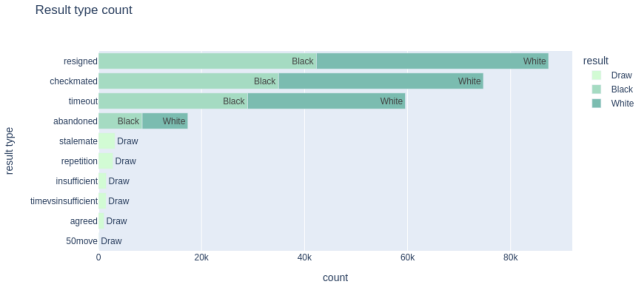
- Eco name = název šachové otevíračky
- Result type = výsledek hry (důvod ukončení)
- Increment = počet sekund, který se hráči přidá po každém tahu.
- Rated = jestli je partie bodově ohodnocená



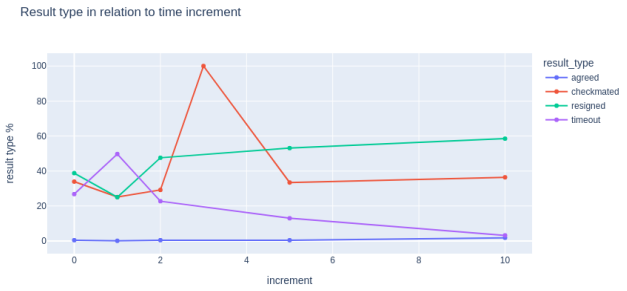
Obrázek 7: Nejčastější šachové otevíračky



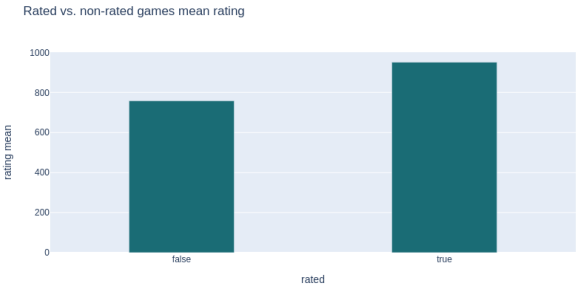
Obrázek 8: Průměrné rozdělení vítězů



Obrázek 9: Rozdělení podle výsledku hry



Obrázek 10: Výběr výsledků v závislosti na časovém inkrementu



Obrázek 11: Rating v závislosti na ohodnocení hry

7Kupodivu zde nevidím Sicilskou obranu či Italskou hru, což jsou jedny z nejznámějších otevíračk. Důvodem bude nejspíše to, že jsme hry získali z malého počtu hráčů, a od každého máme spoustu her, tudíž jejich časté otevíračky velmi ovlivní výsledek. . .

8Potvrdilo se, že bílý hráč má opravdu v šachách výhodu. Očekával jsem, že zhruba 5% her končí remízou, a to se také potvrdilo.

9Jak jsem čekal, nejčastějším ukončením hry je, že se jeden z hráčů vzdá. Čekal jsem, že vypršení času bude na stejno s šach matem, ale mám nejspíše zkreslenou představu, protože hraji spíše kratší hry.

10Chtěl jsem ukázat, že s rostoucím inkrementem klesá počet her končících vypršením času. To se celkem prokázalo, až na odchylku při inkrementu jedné sekundy. Ostatní ukončení hry zobrazuji spíše pro zajímavost.

11Jak jsem očekával, nováčci spíše hrají nehodnocené hry, a proto je pro ně nižší průměrný rating.

3.2 Spuštění

V kořenovém adresáři je soubor index.py, který se z příkazové řádky spustí příkazem 'python3 ./index.py'. Přepínač -d spustí aplikaci v debug módu. Aplikaci si lze otevřít na adrese vypsané v konzoli.

3.3 Místo na zlepšení

Měl jsem v plánu udělat mnohem více grafů, jako např. že se s rostoucím časem hráči více vzdávají, kdo se první odchýlí od známé teorie ten prohraje, nebo jestli hráči s vyšším ratingem mají častěji remízy. Avšak kvůli implementaci získávání dat z API a velkého množství testování jsem se k tomu nedostal. Aplikace by šla také značně vylepšit po grafické stránce.

4 Testování

Kvůli velkému množství funkcí a metod, jsem si nevyzkoušel veškeré techniky, např. fixtures. Přesto by testy měly pokrývat většinu kódu a plnit svůj účel.

4.1 Spuštění

Testy se spustí z kořenového adresáře příkazem 'python3 -m pytest'. Testy se také spustí při každém pushi do větve semestral nebo semestral-tests, díky .gitlab-ci.yml[4] souboru.

4.2 Místo na zlepšení

Po lepším rozčlenění fetch_data logiky bych mohl otestovat jejich funkcionalitu lépe a využil bych více mockování a možná i fixtures.

5 Závěr

Mrzí mě, že jsem si dal za cíl i sběr dat. Myslím, že rozsah mé semestrální práce byl příliš velký, a ztrácel jsem hodiny tam, kde to nakonec není skoro vidět. Vizualizace mne bavila, ale bohužel na ní nebylo dostatek času, ani dostatečné množství dat. Na vizualizaci by se toho dalo zlepšit opravdu hodně. Na druhou stranu mám radost ze struktury, kterou jsem vytvořil. Přidat graf už teď není žádný problém. Fetching dat je také za mě docela spolehlivý. Semestrální práci bych tedy ohodnotil semi-pozitivně. Zkusil jsem si od všeho něco, a i když vizuální výsledek není zrovna nejlepší, věřím, že v mém kódu je vidět kus práce.

Reference

[1] AdityaJha1504. Those features won't engineer themselves. online, 2022. [cit. 2020–01–08] https://www.kaggle.com/code/adityajha1504/those-features-won-t-engineer-themselves.

[2] ArjanCodes. Dash project. online, 2022. [cit. 2020–01–08] https://github.com/ArjanCodes/2022-dash/tree/0a14c8f59d9b657cdf03fb69940c1fdded6e9641/part3.

[3] Chess.com. Chess.com api. online, 2022. [cit. 2020–01–08] https://www.chess.com/news/view/published-data-api.

[4] Eugene Okulik. Gitlab ci. online, 2021. [cit. 2020–01–08] https://eokulik.com/run-pytest-tests-with-report-publication-on-gi

[5] plotly.com. Plotly documentation. online, 2022. [cit. 2020–01–08] https://plotly.com/python/.

[6] stackoverflow.com. Stack overflow. online, 2022. [cit. 2020–01–08] https://stackoverflow.com/.