Predikce zápasů NHL (2.0)

Jakub Hruška (445634)

PA026 - Projekt z umělé inteligence

Abstrakt Cílem tohoto projektu bylo přepracovat nepříliš zdařilé řešení stejného problému (predikce zápasů NHL) z loňského roku. Oproti předešlému řešení jsem se chtěl zaměřit výhradně na týmové statistiky, včetně těch pokročilých, jako jsou CORSI nebo předpokládané góly (xG). Druhou významnou změnou byla změna přístupu k vyhodnocení problému z čistě akademického na sázkově orientovaný s využitím historických předzápasových kurzů. Podařilo se mi vytvořit sázecí strategie využívající naučený predikční model, které porazily baseline strategie pracující pouze s předzápasovými kurzy nabízenými sázkovou kanceláří. Je však potřeba říct, že i tyto lepší strategie jsou výdělečné pouze na trénovacích sezónách a u validačních se pohybují na hraně výdělečnosti a prodělečnosti.

1 Přehled

1.1 Repozitář

Kompletní kód projektu se nachází na GitHubu. Z důvodu velikosi zde však nejsou datasety ani naučené modely. Kód pro jejich vygenerování však v repozitáři je. Jedinou výjimkou jsou historické kurzy, které jsem získal scrapováním webových stránek, a proto bych jej raději nezveřejňoval.

Repozitář obsahuje instalovatelný balíček ve složce `nhl_predict`, který poskytuje veškerou funkcionalitu. Dále jsou zde složky `notebooks` a `scripts`, kde můžete najít jupyter notebooky pro debugging a testování a scripty, které pouštějí jednotlivé části pipeliny. Zbytek složek (`data`, `models`, `outputs`, atd.) jsou prázdné a jsou využívány pro ukládání příslušných souborů, které nejsou trackovány gitem.

1.2 Popis řešení

K natrénování finálního modelu pro predikci zápasů a vyhodnocení vsázecích strategií jej využívajících byly zapotřebí tyto kroky:

1. stáhnout oficiální záznamy (*play-by-play*) z každého zápasu z NHL API
2. vytvořit z play-by-play záznamů dataset pro natrénování xG modelu
3. natrénovat xG model
4. vypočítat z play-by-play záznamů vybrané týmové statistiky (pro každý odehraný zápas – *post-game*)
5. vypočítat xG statistiku a přidat ji k týmovým post-game statistikám
6. vytvořit z post-game statistik *pre-game* statistiky (průměry daného týmu za sezónu, či v posledních n zápasech, před zahájením daného zápasu)

7. natrénovat *game-prediction* model na pre-game statistikách
8. predikovat pravděpodobnosti výsledku zápasů (výhra domácích, hostů či remíza) a přepočítat pravděpodobnosti na šance (formát kurzů)
9. scrapenout historické předzápasové kurzy sázkových kanceláří a spárovat data o kurzech s predikovanými kurzy
10. vytvořit "bota", který na základě kruzů (od sázkových kanceláří, případně i modelem predikovaných) vsadí či nevsadí na daný zápas
11. připravit různé strategie pro bota – baseline bez využití game-prediction modelu a pokročilejší s jeho využitím
12. objektivně porovnat a vyhodnotit strategie

2 NHL API

Přístup k NHL API a formát zde uložených dat jsem důkladněji popisoval v první verzi projektu. Tentokrát jsem více pracoval s play-by-play záznamy. Zde jsou samostatně zaznamenány všechny měřené statistické události (vhazování, střela na bránu i mimo, bodyček, gól, ...), které se v zápase stanou. Včetně pozice na hřišti, kde se událost odehrála, a hráčů, kteří byli aktéry dané situace. Tyto informace jsou obzvlášť důležité pro xG model.

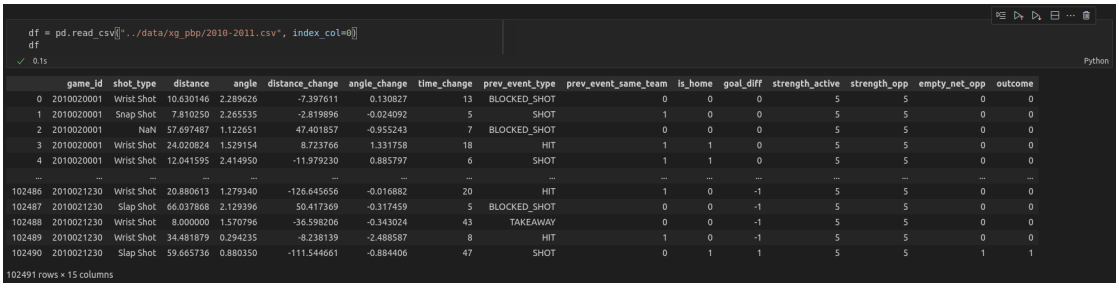
Dále jsem tentokrát přidal záznamy vyscrapované ze zápisu o utkání, kde je uveden i údaj o počtu hráčů na ledě v daný moment. Tím snadno můžeme rozdělit, které události se staly při rovnovážném počtu na ledě, které v přesilovce a které v oslabení. Rozlišování těchto statistik je dnes v hokejové analytické komunitě velmi běžné.

Použil jsem pouze statistiky ze zápasů základní části – 1271 zápasů za sezónu (720 v sezóně zkrácené vylukou a 1231 před rozšířením ligy o Las Vegas) – a pouze ze základní hrací doby (nikoli prodloužení). Kurzovní sázky jsou také vypsané většinou na stav po uplynutí základní hrací doby a povolují tedy remízu.

3 xG model

xG, neboli *expected goals* je statistika, která udává, kolik by měl tým (nebo hráč) vstřelit v zápase gólů z šancí, které si vytvořil. Z play-by-play dat jsem vybral všechny střely (na bránu i mimo bránu) a natrénoval jsem model, který každé střele přiřadí pravděpodobnost, že z ní padne gól, na základě aktuální situace na ledě: vzdálenost od brány, úhel k bráně, čas od předešlé události, typ předešlé události, typ střely apod. Střely z první ze slotu (nejnebezpečnější oblast před bránou), dorážky z opačné strany než první střela apod. mají výrazně větší šanci na gól než nahození z poloviny hřiště.

Tato statistika není specialitou pouze hokeje. Je také běžné, že existuje více modelů a různí analytici uvádějí různé xG hodnoty v závislosti na použitém xG



```
df = pd.read_csv("../data/xg_pbp/2010-2011.csv", index_col=0)
df
```

0.1s — Python

	game_id	shot_type	distance	angle	distance_change	angle_change	time_change	prev_event_type	prev_event_same_team	is_home	goal_diff	strength_active	strength_opp	empty_net_opp	outcome
0	2010020001	Wrist Shot	10.630146	2.289626	-7.397611	0.130827	13	BLOCKED_SHOT	0	0	0	5	5	0	0
1	2010020001	Snap Shot	7.810250	2.265535	-2.819896	-0.024092	5	SHOT	1	0	0	5	5	0	0
2	2010020001	NaN	57.697487	1.122651	47.401857	-0.955243	7	BLOCKED_SHOT	0	0	0	5	5	0	0
3	2010020001	Wrist Shot	24.020824	1.529154	8.723766	1.331758	18	HIT	1	1	0	5	5	0	0
4	2010020001	Wrist Shot	12.041595	2.414950	-11.979230	0.885797	6	SHOT	1	1	0	5	5	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
102486	2010021230	Wrist Shot	20.880613	1.279340	-126.645656	-0.016882	20	HIT	1	0	-1	5	5	0	0
102487	2010021230	Slap Shot	66.037868	2.129396	50.417369	-0.317459	5	BLOCKED_SHOT	0	0	-1	5	5	0	0
102488	2010021230	Wrist Shot	8.000000	1.570796	-36.598206	-0.343024	43	TAKEAWAY	0	0	-1	5	5	0	0
102489	2010021230	Wrist Shot	34.481879	0.294235	-8.238139	-2.488587	8	HIT	1	0	-1	5	5	0	0
102490	2010021230	Slap Shot	59.665736	0.880350	-111.544661	-0.884406	47	SHOT	0	1	1	5	5	1	1

102491 rows × 15 columns

Obrázek 1: Ukázka dat pro xG model. Jeden řádek odpovídá jedné střele. Outcome značí, zda padl ze střely gól.

modelu. Výhodou je, že lze takto spočítat xG i pro jendotlivé hráče v poli, ba i pro brankáře spočítat jak náročným střelám čelili. Pohledem na rozdíl reálně vstřelených gólů a xG pak lze usuzovat například, který brankář chytá nadprůměrně a zachránil svůj tým při náročných situacích, nebo který hráč si sice tvoří šance, ale nezvládá je proměňovat.

Já jsem zde však nic takového neřešil a zaměřil jsem se pouze na týmové statistiky. Týmové xG jsem získal prostým součtem xG celého týmu za daný zápas. xG v mém projektu slouží jako další statistika, která by měla vhodně vypovídat o kvalitě týmu (šancí, které si během zápasu vytváří). Predikce, které xG model (Extra Trees Regressor) produkuje, jsem přidal k ostatním post-game statistikám ze zápasu (počet střel, počet hitů apod. – viz 4.1).

Nejlepší klasifikátor a jeho hyperparametry byly nalezeny pomocí cross validation grid search a byl to Extra Trees Regressor s parametry:

- n_estimators = 10
- max_depth = 20
- min_samples_split = 256

Výstup tohoto modelu byl poté normalizován do rozmezí [0; 1]. Pro každou střelu tak model vrátí pravděpodobnost, s jakou z ní padne gól. Těmito pravděpodobnostmi jsem poté provážil střely v zápase, abych dostal týmové xG pro daný zápas.

4 Týmové statistiky

4.1 Post-game

Post-game statistiky udávají pro každý odehraný zápas statistické zhodnocení toho, co se na ledě odehrálo. Jsou to typicky počty střel, počty přesilovek a čas v nich strávený, zblokované střely, hity apod. Dále to jsou pokročilejší statistiky, jako jsou CORSI (součet střel na bránu i střel mimo) a již zmíněná statistika xG.

Každá statistika byla zaznamenána zvlášť ve 4 variantách: při rovnovážném stavu na ledě, při přesilovce daného týmu, při oslabění daného týmu a celkově bez ohledu na situaci na ledě. Výsledné statistiky poté vypadají např. takto: *away_BLK_ALL, away_BLK_EV, away_BLK_PP, away_BLK_SH, ...*

Kde *away* udává, že se jedná o hostující tým; *BLK* udává, že se jedná o zblokované střely a např. *EV* udává, že jde pouze o rovnovážný stav na ledě (even strength).

```
df = pd.read_csv("../data/games_stats/post_game/2010-2011.csv", index_col=0)
df
```

✓ 0.8s

	away_BLK_ALL	away_BLK_EV	away_BLK_PP	away_BLK_SH	away_CORSI_ALL	away_CORSI_EV	away_CORSI_PP	away_CORSI_SH	away_FOW_ALL	away_FOW_EV	...	home_SOG_SH	home_TAKE_ALL
1	21	14	1	6	65	51	6	8	23	17	...	0	6
2	16	12	1	3	51	42	4	5	22	15	...	1	9
3	19	14	1	4	59	55	3	1	33	25	...	0	8
4	10	9	0	1	60	36	15	9	27	21	...	1	12
5	17	14	0	3	79	55	13	11	29	19	...	0	11
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1226	10	9	0	1	44	37	5	2	28	25	...	1	1
1227	15	12	1	2	41	35	5	1	29	17	...	2	5
1228	14	12	0	2	40	33	3	4	15	10	...	0	11
1229	0	0	0	0	0	0	0	0	0	0	...	0	0
1230	14	14	0	0	48	30	14	4	29	20	...	0	9

1230 rows × 84 columns

Obrázek 2: Ukázka post-game statistik.

4.2 Pre-game

Post-game statistiky nejsou použitelné pro predikci zápasů, jelikož nám říkají, jak zápas dopadl poté, co se odehrál. My však potřebujeme vědět, jak na tom týmy jsou před začátkem záapsu. K tomu slouží pre-game statistiky. Ty jsem získal tak, že jsem zprůměroval post-game hodnoty z předešlých zápasy daného týmu v sezóně.

Pro větší výpovědní sílu jsem také přidal průměry pouze posledních domácích respektive hostujících zápasů daného týmu. Pro každou z těchto variant jsem nakonec vypočítal i průměr jen z posledních 3 zápasů jako reprezentaci aktuální formy týmu.

Výsledný dataset (jedna sezóna) měl cca 850 proměnných a 1271 (nebo 1230, případně 720) řádků. Každý řádek představoval jeden zápas a obsahoval souhrnné statistiky domácího a hostujícího týmu z dané sezóny v momentu před začátkem daného zápasu. Příklad sloupců: *away_GF_ALL_away, away_GA_ALL_away, away_GF_ALL_away_last3, away_SMISSF_SH_both, ...*

Kde první *away* znamená hostující tým v daném zápasu; *GF* znamená vstřelené góly (goals for) a *GA* zase obdržené (goals against); *ALL* a *SH* označují herní situaci (všechny herní situace resp. oslabení – short handed); druhé *away* případně *both* udávají, z jakých zápasů daného týmu byl průměr průměr počítaný a *last3*

značí, že se jedná o průměr z pouze posledních 3 zápasů (hostujících, domácích nebo jakýchkoliv) daného týmu.

```
df = pd.read_csv("../data/games_stats/pre_game/2010-2011.csv", index_col=0)
df
```

✓ 0.2s

	away_GF_ALL_away	away_GA_ALL_away	away_GF_ALL_away_last3	away_GA_ALL_away_last3	away_GF_EV_away	away_GA_EV_away	away_GF_EV_away_last3
1	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN
5	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...
1226	2.950	2.375	2.333333	3.666667	2.250	1.800	1.666667
1227	2.575	2.350	2.666667	2.333333	1.800	1.750	2.333333
1228	2.375	3.250	2.333333	3.666667	1.850	2.225	1.000000
1229	3.000	2.625	1.333333	3.000000	2.025	1.875	1.000000
1230	2.550	2.975	2.666667	2.666667	1.900	2.100	2.333333

1230 rows × 864 columns

Obrázek 3: Ukázka pre-game statistik. Po MinMax naškálování jednotlivých proměnných vstupují tato data do game-prediction NN modelu. NaN hodnoty jsou u takových zápasů, kde nelze spočítat průměry z předešlých post-game statistik. Například tedy první zápasy každého týmu v sezóně. Tyto zápasy byly z tréninku vyřazeny.

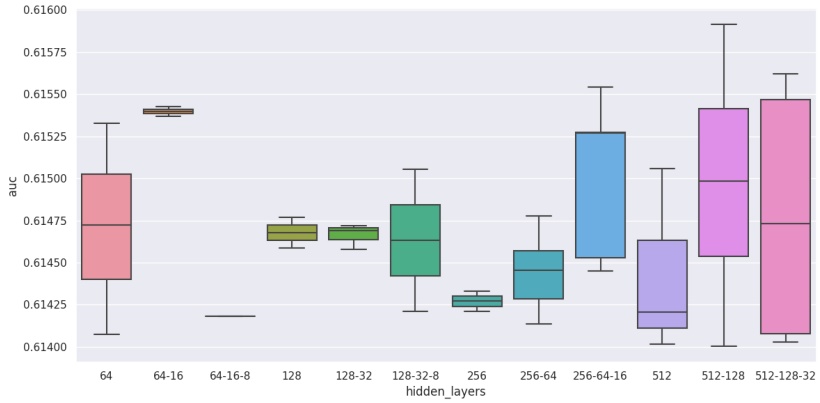
5 Game-prediciton model

Do hlavního modelu pro predikci výsledků zápasů vstupují pre-game statistiky. Tedy průměry post-game statistik z různě vybraných zápasů daného týmu (všechny doposud v sezóně odehrané, jen domácí, jen poslední 3, jen poslední 3 venkovní, apod.). Výstupem modelu pak je vektor pravděpodobností: (výhra domácích, remíza, výhra hostů), normalizovaný díky softmaxu do 1.

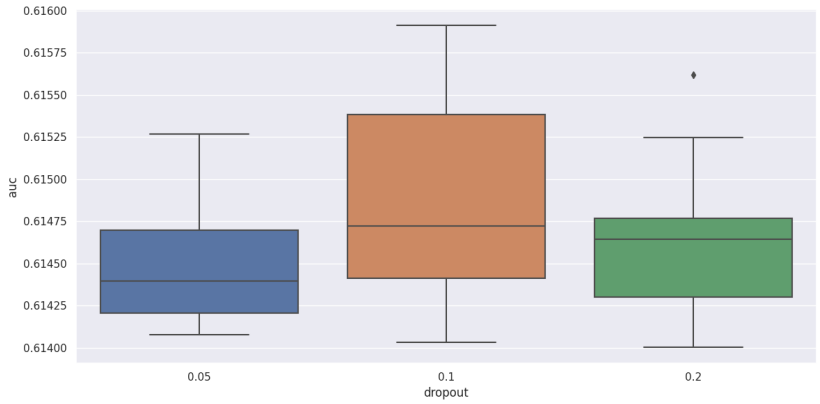
Vzhledem k dramatickému zpoždění jsem se pro model pro predikci zápasů rozhodl použít neuronovou síť a nezkoušel větší množství různých modelů. Grid search jsem proto aplikoval pouze na nalezení optimální topologie a hyperparameterů neuronové sítě.

V rámci grid searche byla opět použita i cross validation. Tentokrát upravená tak, že v každém foldu byly použity 3 po sobě jdoucí sezóny jako trénovací a 4. jako validační. V dalším foldu nejstarší trénovací vypadla, validační se přidala k trénovacím a jako nová validační byla vzata 5. sezóna v řadě.

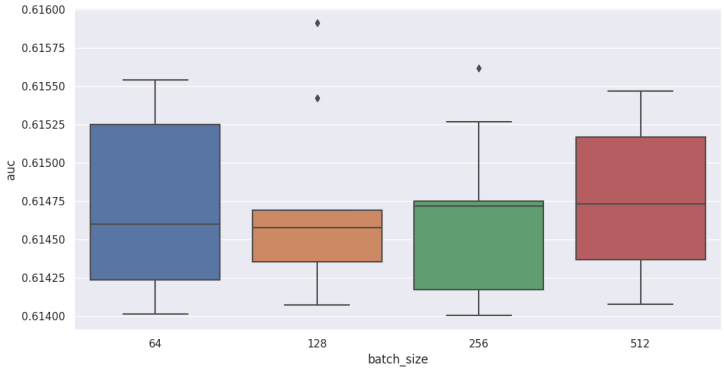
Pevně použité parametry byly: *Adam* optimizer, loss funkce *CategoricalCrossentropy*, *ReLU* jako aktivační funkce skrytých vrstev a *softmax* jako aktivační funkce poslední vrstvy, která měla 3 neurony (výhra domácích, remíza, výhra hostů). Dále byl použit *Early stopping* na validační hodnotě loss funkce s patience 15 epoch. Pro výběr nejlepších parametrů jsem použil metriku AUC na validačních datech.



Obrázek 4: Vliv topologie sítě na validační AUC.



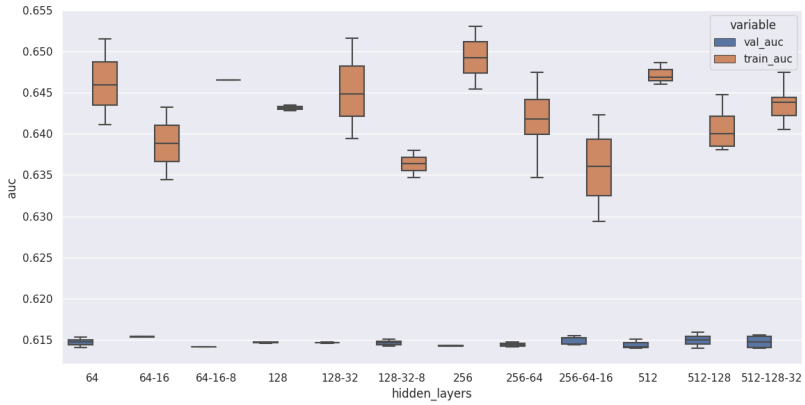
Obrázek 5: Vliv dropoutu na validační AUC.



Obrázek 6: Vliv velikosti minibatche na validační AUC.

Finálně byla vybrána síť 864-512-128-3 s dropoutem 0.1 po každé vrstvě skryté vrstvě a velikostí minibatche 128 (viz obrázky 4, 5 a 6). Mimo AUC jsem sledoval i accuracy, precision a recall. Nebylo však mým cílem vyhodnotit tuto síť z pohledu ML úlohy, ale za pomocí výdělečnosti strategií. Tyto metriky zde proto nejsou příliš relevantní. Dále zde bylo patrné velmi jednoduché přetrénování (obr. 7), což ovšem není u neuronových sítí nic překvapivého.

Tuto finální síť jsem nakonec natrénoval na 5 sezónách 2011-2015 a jako validační jsem použil 3 sezóny 2016-2018[1].



Obrázek 7: Neuronová síť byla velmi náchylná na přetrénování.

Jelikož je aktivační funkce poslední vrstvy softmax, můžeme nahlížet na výstup ze sítě jako na pravděpodobnost, že nastane daný jev. Pravděpodobnost P z rozsahu $[0; 1]$ můžeme do formátu kurzu (šance) převést pomocí vzorce $odd = \frac{1}{P}$. Pokud nám model vrátí pravděpodobnosti např. následovně: výhra domácích $(1) = 0.5$; výhra hostů $(2) = 0.2$; remíza $(X) = 0.3$, pak tyto pravděpodobnosti přepočítané na kurzy (šance) budou: $(1) = 2$; $(2) = 3.33$; $(X) = 5$.

6 Sázení

6.1 Historické kurzy

Předzápasové kurzy pro zápasy až do roku 2005 jsem nalezl na stránce OddsPortal. Z té jsem potřeboval dostat data pomocí scrapování. Ačkoliv by to z právního hlediska mělo být legální a v `robots.txt` jsem nic problematického nenašel, samotné

[1] Kurzy jsem nalezl až do sezóny 2005-2006, ovšem podrobné play-by-play statistiky NHL zveřejňuje až od sezóny 2011-2012. Poslední sezóny zase byly výrazně ovlivněny pandemií COVID-19 a mohly by být značně zkreslující.

scrapovaní nebylo vůbec jednoduché. Musel jsem obejít několik nástrah, které (ať už záměrně, či nikoli) znepříjemňovaly automatický přístup ke stránce nebo nalezení požadovaného obsahu na ní. Vše nakonec vyřešila knihovna `selenium`, která umožňuje otevřít opravdové okno prohlížeče a používat na stránce myš.

Druhý problém, se kterým jsem se musel vypořádat, že řada dat (převážně starších) byla různě špinavých. Některé zápasy chyběly úplně, někde chyběly jen některé kurzy, některé měly špatný formát apod. Vše bylo potřeba při vytváření datasetu pročistit.

6.2 Bot

Nejedná se v pravém slova smyslu o bota, který by snad reálně prováděl sázky u sázkové kanceláře. Jeho úkolem je vyhodnotit danou strategii s daným parametrem.

Buď může vsadit podle strategie na jednu sezónu a vrátit detailní DataFrame s tím, jaký byl na každý zápas kurz, jaký byl modelem predikovaný kurz, jestli byla provedena sázka, zda byla vítězná atd. Pro jednoduchost jsem nastavil vždy stejnou hodnotu sázky a to 10 Kč. Bot se pouze na základě strategie rozhodne, zda sázku vloží a případně na jaký výsledek[2]

Dále může vsadit podle dané strategie na více sezón. V takovém případě vrátí pro každou sezónu sumární statistiky:

- deposit – kolik peněz vložil do sázek
- revenue – kolik ze sázek vyhrál
- profit – rozdíl mezi revenue a deposit
- profit rate – podíl profit a deposit (profit rate = 0 → profit je nulový; profit rate = 1 → vydělal si jednou tolik, kolik do sázek vložil; profit rate = -1 → prohrál celý svůj deposit)
- bet rate (bet ratio) – na kolik procent zápasů bylo vsazeno (jak odvážný bot je)[3]

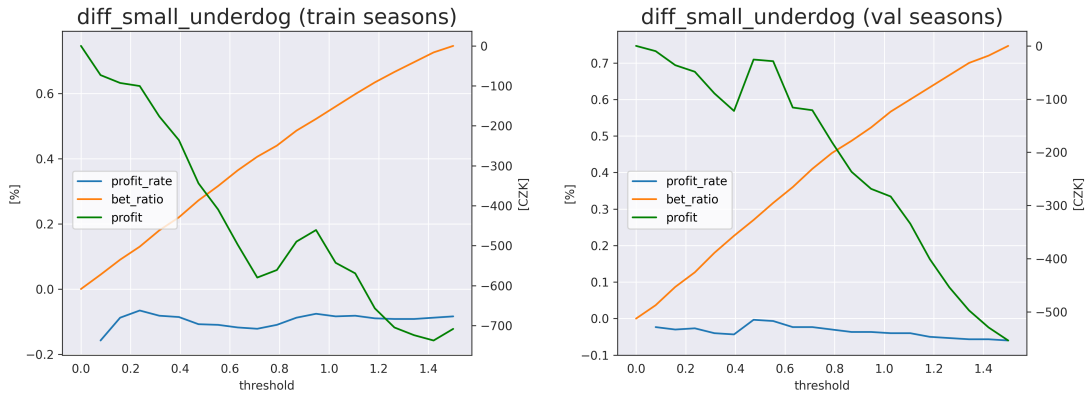
Poslední možností pak je tzv. bootstrapping. Tedy způsob, jak z náhodného vzorku (hodnoty pro jednotlivé sezóny) odhadnout průměr a interval spolehlivosti dané náhodné veličiny. Tímto způsobem pak dostaneme sumární statistiku o dané strategii.

6.3 Strategie

Strategie jsem nadefinoval jako funkce, které dostanou jeden řádek datasetu s kurzy (ze sázkové kanceláře, případně i predikované z modelu) + případný parametr a vrátí NaN, pokud doporučují nevsadit, nebo "1", "X", "2"podle toho, na který tým (nebo remízu) doporučují vsadit.

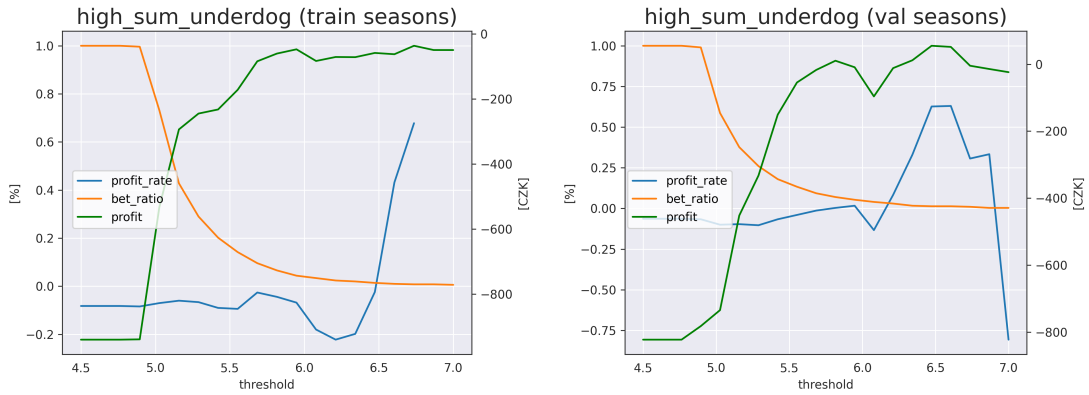
[2] Zde je zajisté prostor pro rozšíření – výše sázky na základě jistoty modelu.

[3] Je možné, že je někde profit rate a bet rate (bet ratio) uváděn v procentech a jinde jako desetinné číslo mezi 0 a 1.



Obrázek 8: Vliv thresholdu na strategii: Vsaď na outsidera, pokud je rozdíl mezi ním a favoritem menší než threshold.

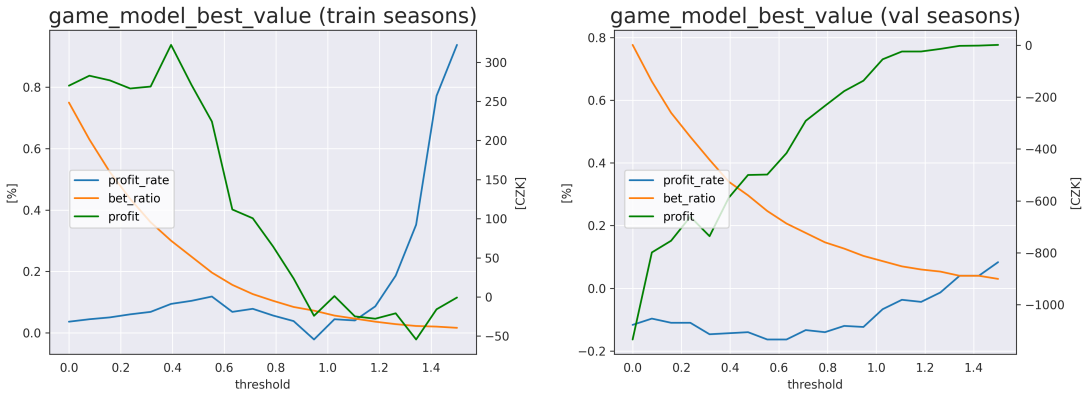
Baseline strategie jsou vesměs velmi jednoduché typu: "vsaď vždy na favorita", "vsaď vždy na domácí tým"apod., pokud je kurz pro ně v určitém rozsahu (*odd_range*). Pro porovnatelnost s model-based strategiemi jsem vyhodnocoval strategie zvlášť na trénovacích a validačních sezónách, i když na baseline strategie nemělo rozdělení žádný vliv, protože nebyly nijak učeny. Rozdíly mezi jednotlivými sezónami jsou tak spíše náhodné.



Obrázek 9: Vliv thresholdu na strategii: vsaď na ousidera pokud součet kurzů na favorita a outsidera je větší než threshold.

Model-based strategie využívaly ke svým rozhodnutím modelem predikované kurzy. Připravil jsem dvě strategie: "vsaď na favorita (nejnižší kurz) podle modelu, bez ohledu na reálné kurzy"a "vsaď na tu událost, kterou sázková kancelář nejvíc

podceňuje". Tedy takovou událost, kde se nejvíce liší kurz sázkové kanceláře a kurz modelu a model dává menší kurz (myslí si, že je událost pravděpodobnější než sázková kancelář).



Obrázek 10: Vliv thresholdu na strategii *game_model_best_value*.

První strategie (*game_model_favorite*) má parametr odd_range obdobně jako základní strategie favorit. Druhá strategie (*game_model_best_value*) má parametr threshold, který udává, jak minimálně velký musí být největší rozdíl mezi kurzy.

Nejlepší hodnoty parametrů jednotlivých strategií jsem nalezl pomocí zmiňovaného bootstrappingu. Výsledky jsem vynesl do grafů (obr. 11 - obr. 16).

7 Vyhodnocení

Jeidná baseline strategie, která nebyla výrazně prodělečná, byla *high_sum_underdog* s thresholdem 6.5. Nicméně zde byl extrémně nízký bet rate, takže bot vsadil za sezónu jen na jednotky až nízké desítky zápasů (z cca 1200) – viz tabulka 1.

Nejlepší threshold pro strategii *game_model_best_value* byl 0.5. Zatímco na tréninkových sezónách je tato strategie slušně výdělečná, na validačních je jednoznačně prodělečná. Bet rate se pohybuje okolo cca čtvrtiny až třetiny (tabulka 2).

Pro mne trochu překvapivě nejlépe dopadla strategie *game_model_favorite* s rozsahem od 1.3 do 2.1 (viz tabulka 3). Na základě článků a rad od sázkařů jsem očekával, že důležité bude nalézt právě výhodný kurz, ne nutně favorita zápasu. Očekával jsem proto, že předešlá strategie bude výnosnější. Je vidět, že bet rate je o něco nižší než u předešlé strategie a ve validačních sezónách se ještě snížil.

	deposit	revenue	bet_rate	profit	profit_rate
2011	70	105.1	0.01	35.1	0.50
2012	30	0.0	0.00	-30.0	-1.00
2013	140	55.4	0.01	-84.6	-0.60
2014	530	225.3	0.04	-304.7	-0.57
2015	30	103.1	0.00	73.1	2.44
2016	140	207.4	0.01	67.4	0.48
2017	100	264.5	0.01	164.5	1.64
2018	280	213.0	0.02	-67.0	-0.24

Tabulka 1: Strategie *high_sum_underdog* s thresholdem 6.5.

	deposit	revenue	bet_rate	profit	profit_rate
2011	2170	2724.5	0.18	554.5	0.26
2012	1300	1381.4	0.18	81.4	0.06
2013	2670	3042.0	0.22	372.0	0.14
2014	3830	3654.6	0.31	-175.4	-0.05
2015	3210	3633.6	0.26	423.6	0.13
2016	3240	2532.4	0.26	-707.6	-0.22
2017	3220	2691.1	0.25	-528.9	-0.16
2018	4140	3750.4	0.33	-389.6	-0.09

Tabulka 2: Strategie *game_model_best_value* s thresholdem 0.5.

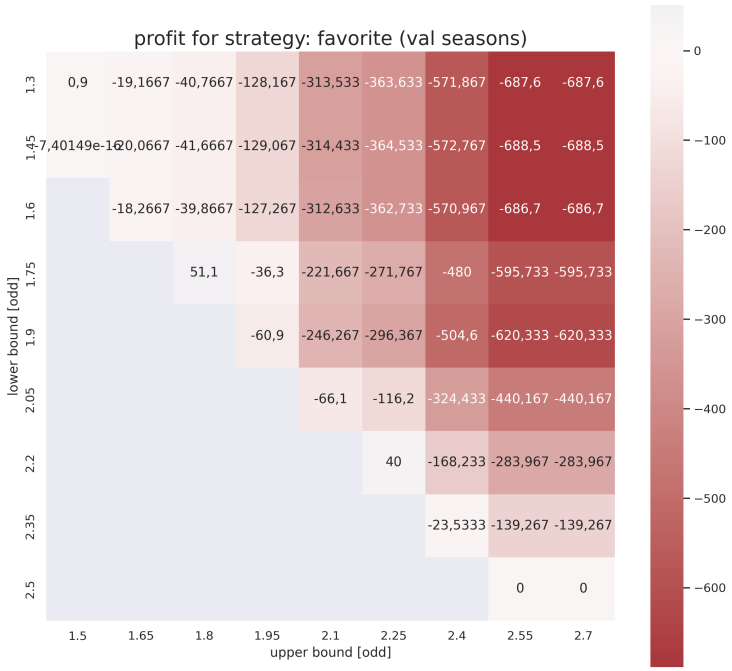
Profit na tréninkových sezónách mne výrazně překvapil. Upřímně jsem nečekal, že by bylo reálné něčeho tak výrazného dosáhnout (místy profit rate přes 10 %). Validační sezóny se však zdají být velmí kolísavé a nutí ke klidnění optimismu.

	deposit	revenue	bet_rate	profit	profit_rate
2011	2570	2833.3	0.21	263.3	0.10
2012	1600	1827.7	0.22	227.7	0.14
2013	2920	3117.8	0.24	197.8	0.07
2014	2540	2714.3	0.21	174.3	0.07
2015	1850	2145.0	0.15	295.0	0.16
2016	1580	1591.5	0.13	11.5	0.01
2017	1840	1501.1	0.14	-338.9	-0.18
2018	1570	1655.9	0.12	85.9	0.05

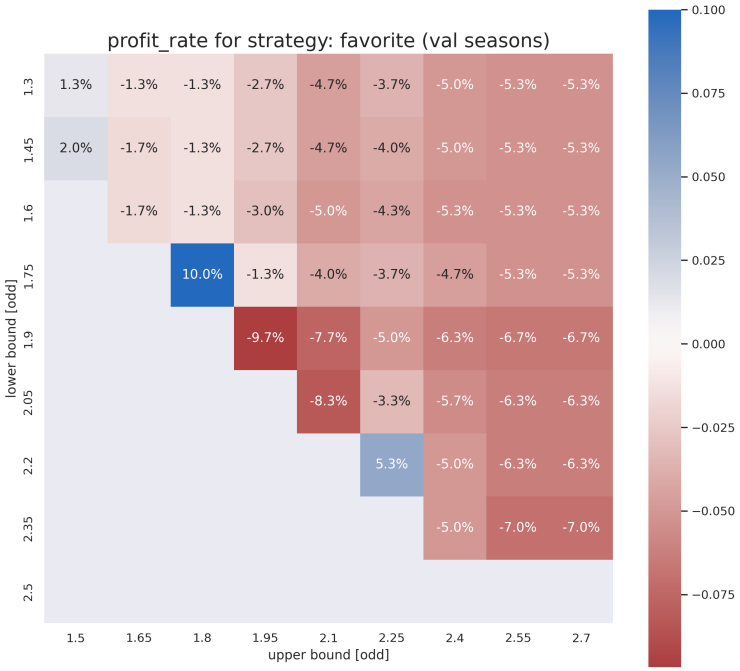
Tabulka 3: Strategie *game_model_favorit* s rozsahem (1.3; 2.1).

Celkově bych zhodnotil projekt jako zdařilý. Podařilo se mi natrénovat model, s jehož pomocí jsem zvládl jednoznačně porazit všechny baseline strategie. Nejlepší z nich (high_sum_underdog) je relativně konkurenceschopná jen s velmi přísně nastaveným thresholdem, kde spoléhá na výsledek jen několika málo zápasů. Zbylé baseline strategie jsou výrazně prodělečné.

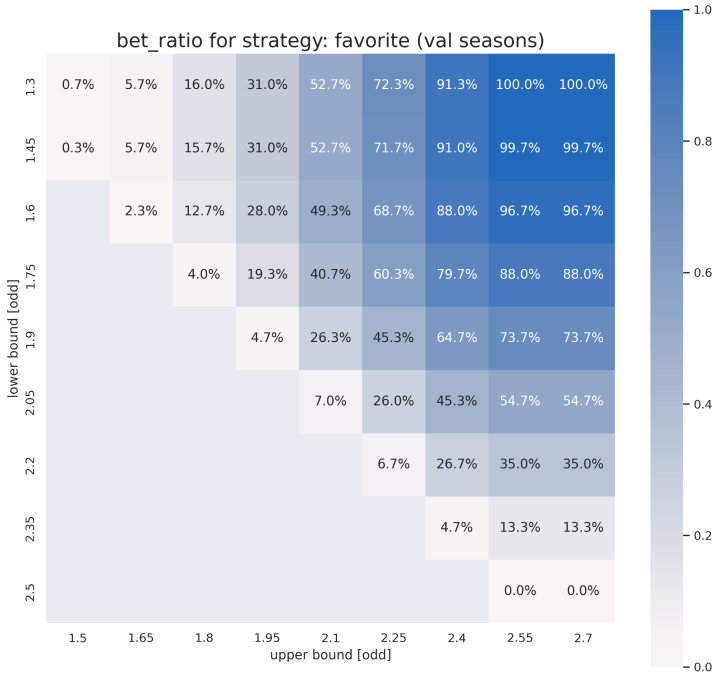
Míst, kde by se dalo zapracovat a zlepšit se, je mnoho. Jedním z nich zajisté bude prevence přeučení a vyrovnání rozdílu mezi výsledky na trénovacich a validačních datech. Dále by stálo za to vyzkoušet lepší feature engineering. Přidat víc a lepších statistik, zjistit vliv těch stávajících a případně je zredukovat. Z technologického hlediska zatím vůbec není nachystána pipeline pro predicki úplně nových zápasů (vypočítání statistik, zjištění kurzu, výpočet xG apod.).



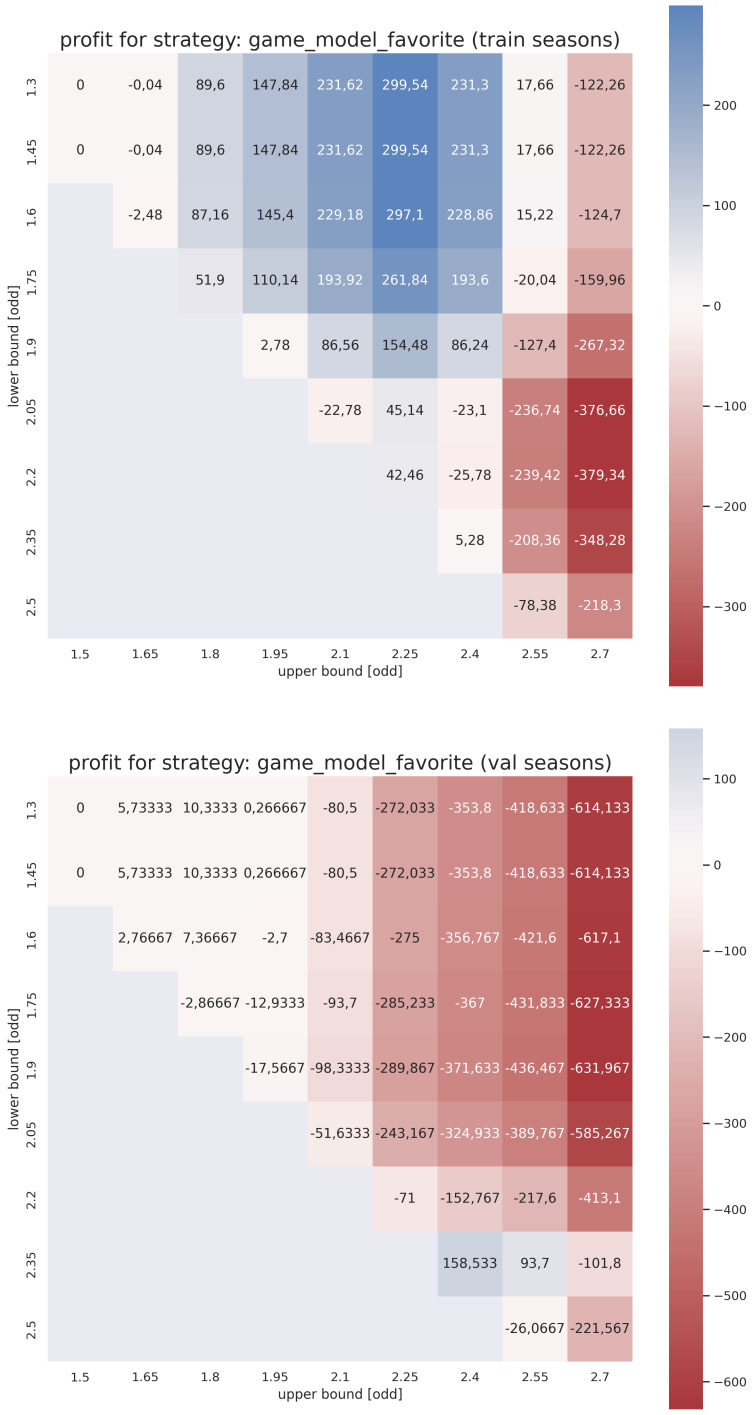
Obrázek 11: Vliv parametru odd_range na profit strategie favorit.



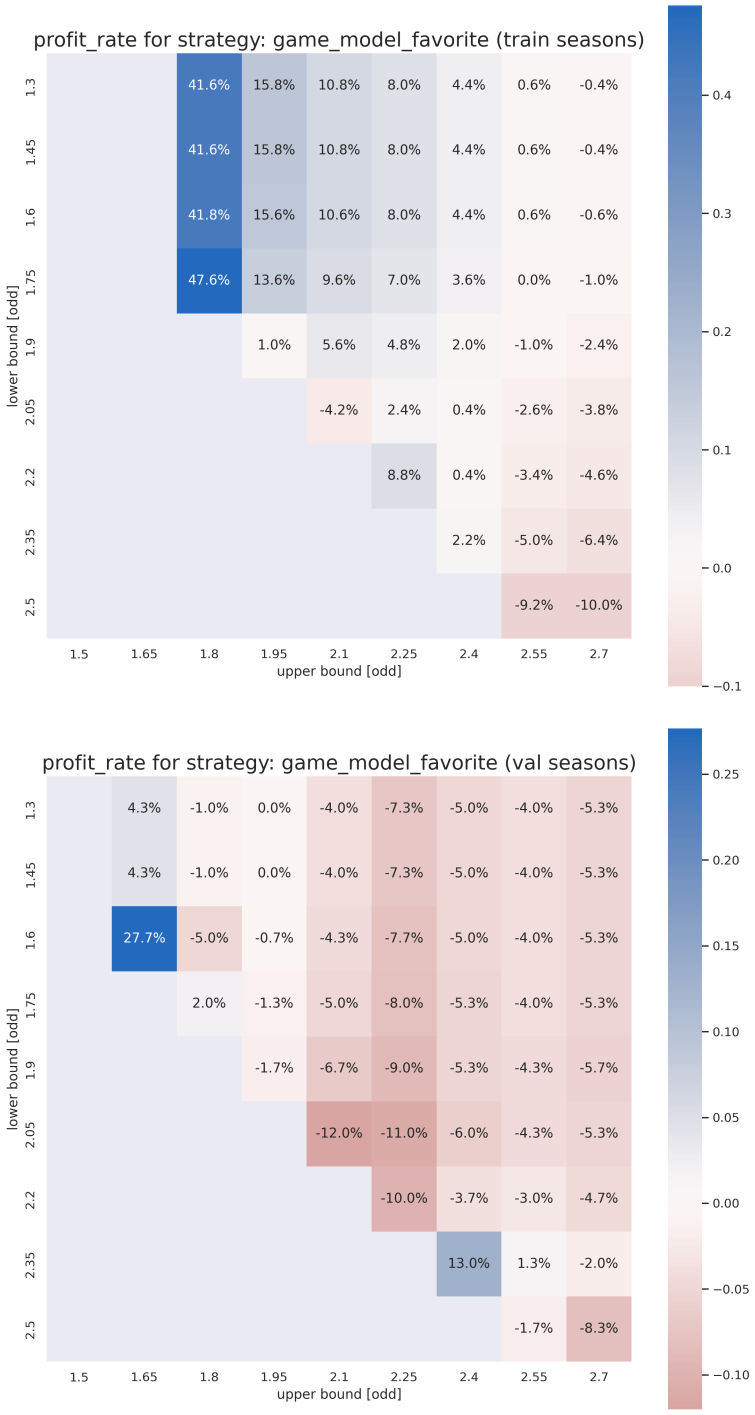
Obrázek 12: Vliv parametru odd_range na profit rate strategie favorit.



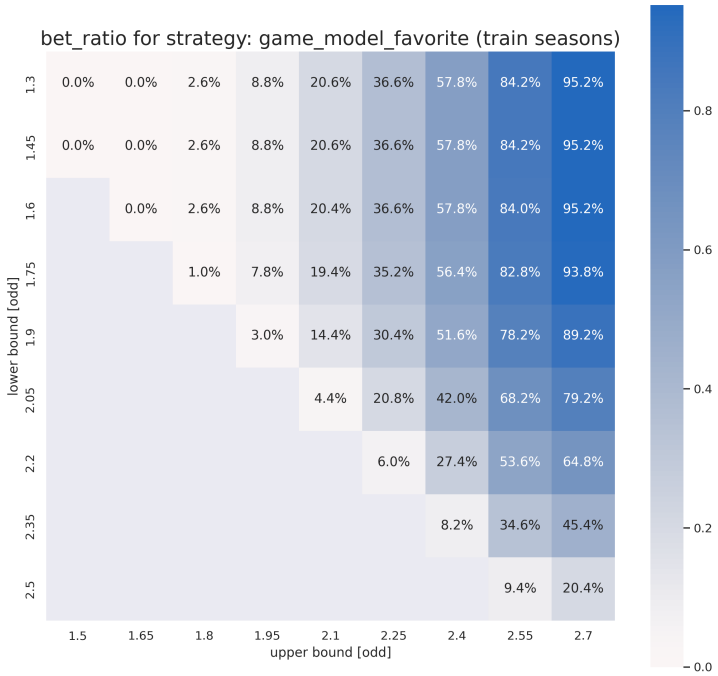
Obrázek 13: Vliv parametru odd_range na bet rate strategie favorit.



Obrázek 14: Vliv parametru odd_range na profit strategie *game_model_favorite*. Je zde patrný rozdíl mezi trénovacími sezónami a validačními.



Obrázek 15: Vliv parametru odd_range na profit rate strategie *game_model_favorite*. Výrazně pozitivní dlaždice jsou ovlivněny velmi malým vzorkem dat.



Obrázek 16: Vliv parametru odd_range na bet rate strategie *game_model_favorite*.