Czynnik ludzki, a statystyki – sztuczna inteligencja do gry Fantasy Premier League

(The human factor versus statistics – artificial intelligence for the Fantasy Premier League)

Dawid Paluszak

Praca inżynierska

Promotor: dr Jakub Kowalski

Uniwersytet Wrocławski
Wydział Matematyki i Informatyki
Instytut Informatyki

27 lutego 2021

Streszczenie

Praca ta opisuje proces tworzenia sztucznej inteligencji do gry przeglądarkowej Fantasy Premier League. Gra polega na jak najlepszym wyborze składu prawdziwych piłkarzy, którzy będą zdobywać dla nas punkty poprzez swoją grę. Przedstawiono trzy modele, które próbują pokonać występujący tu czynnik ludzki. Każdy z modeli, na podstawie statystyk z poprzednich meczów danych zawodników, próbuje przewidzieć jacy piłkarze będą najlepsi w kolejnym tygodniu rozgrywek.

This thesis describes the process of creating artificial intelligence for the Fantasy Premier League browser game. The game is all about choosing the best team of real players who will earn points for us through their game. Three models are presented that try to overcome the human factor involved. Each model, based on the statistics from the previous matches of given players, tries to predict which players will be the best in the next week of games.

Spis treści

Rozdział 1.

Wstęp

Sztuczna inteligencja jest coraz powszechniej spotykanym elementem ludzkiego życia. Rozpoznawanie mowy w różnych urządzeniach elektronicznych, wyszukiwanie najlepszych reklam dla użytkowników, polecanie najbardziej pasujących produktów względem naszej historii wyszukiwań czy nawet samochody sterowane przez sztuczną inteligencję to jedne z wielu zastosowań we współczesnym świecie. Od dawna jest wykorzystywana również w grach komputerowych, gdzie twórcy gier tworzą boty, które będą towarzyszyły graczom w wirtualnym świecie. Młodszym zastosowaniem w grach jest pisanie sztucznych graczy, którzy będą przechodzić grę w jak najbardziej optymalny sposób.

Dobrym gatunkiem gier do takiego zastosowania są turowe gry strategiczne. Na podstawie aktualnej sytuacji w grze, SI może ustalić najoptymalniejszy ruch na kolejną iterację rozgrywki. Najbardziej popularne są tu gry karciane, takie jak Hearthstone. Wykorzystując różne podejścia naukowcy z całego świata próbują stworzyć SI, które podoła każdemu graczowi [1, 2].

1.1. Historia turowych gier strategicznych

Początek turowych gier strategicznych sięga starożytnych czasów, gdzie grano w różne planszowe gry strategiczne. Jednak najbardziej znaną planszową grą planszową są szachy, których początek sięga VI wieku. Za pierwszą grę wideo z tego gatunku uznaje się *Microchess* stworzoną przez *Peter R. Jennings* w 1976. Jednak to druga gra wpłynęła znacznie bardziej na rozwój gatunku. Gra pod tytułem *Empire* z 1977 napisana przez *Walter Bright* była pierwszą grą, która przypomina dzisiejsze turowe strategie ekonomiczne. Była ona inspiracją dla twórców znanej serii *Sid Meier's Civilization*, w której zarządzamy rozwojem miasta, państwa czy całej cywilizacji, walcząc z innymi graczami lub SI.

W latach 90. oraz na początku XXI wieku gatunek ten miał swoje najlepsze lata. Później powstawało coraz mniej gier z tego gatunku. Dziś największą popularnością

cieszą się wieloosobowe gry karciane takie jak *Hearthstone*, *Legends of Runeterra* czy *Magic: The Gathering Arena*.

1.2. Cechy turowych gier strategicznych

Turowe gry strategiczne są bardzo podobne do dzisiejszych strategicznych gier planszowych. Każdy z graczy w swojej turze może zajmować się rozwijaniem swoich posiadłości, ruchem jednostek czy walkami, wedle zasad danej gry. Celem takich gier najczęściej jest pozbycie się wszystkich przeciwników (zwycięstwo militarne) lub wygranie poprzez zebranie większej ilości zasobów (zwycięstwo ekonomiczne).

Istnieje również podgatunek turowych gier strategicznych nazywany „Taktyczną grą fabularną". Ten gatunek charakteryzuje się tym, że historia opowiadana jest jak w typowych grach RPG, a walki odbywają się w sposób turowy, gdzie liczy się strategiczne myślenie.

Rozdział 2.

Fantasy Premier League

2.1. Wstęp

Fantasy Premier League (FPL) [3] jest grą przeglądarkową, w której zadaniem gracza jest wybranie składu piłkarskiego składającego się z rzeczywistych graczy. Jak nazwa wskazuje mecze jak i zawodnicy pochodzą z ligi angielskiej (Premier League jest to oficjalna nazwa ligi angielskiej) z najwyższego szczebla rozgrywek. Pierwszy oficjalny sezon w FPL odbył się w 2002 roku. Gra przez te wszystkie lata przyciągnęła spore grono odbiorców. Na ten moment w sezonie 2020/21 bierze udział prawie 8 milionów graczy.

Wybór składu na kolejne tygodnie rozgrywek jest dosyć skomplikowany. Gra posiada dodatkowe bonusy, których możemy używać oraz wybór zawodników jest w pewien sposób ograniczony (Patrz Sekcja Opis gry). Ja w swojej pracy ograniczę się do wskazania najlepszych zawodników względem ich statystyk z poprzednich meczów.

2.2. Opis gry

Zaczynając grę w FPL, gracz dostaje 100 milionów funtów i za ich pomocą musi wybrać swój początkowy skład składający się z 15 zawodników. Dwóch bramkarzy, pięciu obrońców, pięciu pomocników oraz trzech napastników. Każdy z zawodników ma swoją cenę, w zależności od tego ile zebrał punktów w poprzednim sezonie. Gracz ustala potem swój pierwszy skład, składający się z 11 zawodników (reszta zawodników jest „na ławce") oraz musi wybrać kapitana. Kapitan dostaje podwojoną ilość punktów, jaką udało mu się zdobyć podczas meczu. Następnie zawodnicy rozgrywają swoje mecze i zdobywają dla nas punkty (Patrz Sekcja Punktacja). Podczas rozgrywek nie możemy już wpływać na nasz skład. Po meczach następuje „tura" gracza, w której może wykonać jedną z akcji:

- **Wykonać transfer** – gracz może sprzedać zawodników i kupić nowych. Za pierwszy transfer w turze płacimy tylko cenę zawodnika. Za kolejne transfery w turze płacimy punktami, które zdobywamy podczas gry.
- **Aktywować bonus** – gracz posiada trzy bonusy, które może aktywować raz w ciągu całego sezonu. „Triple Captain", czyli potrojenie punktów za kapitana. „Free Hit", czyli nieskończona liczba transferów na ture. „Bench Boost", czyli do punktujących zawodników zostaną wliczeni zawodnicy z ławki.
- **Zmienić skład** – możemy dowolnie modyfikować nasz skład na kolejne mecze oraz dowolnie zmieniać kapitana.

Dodatkowo raz na pół sezonu możemy za darmo wymienić cały skład. Nazywane jest to „Wildcard".

2.3. Punktacja

Zawodnicy, których wybierzemy do składu wyjściowego zdobywają dla nas punkty. Zawodnik, który nie zagra zostanie zastąpiony zawodnikiem z ławki o ile nadal w naszym składzie wyjściowym będziemy posiadać co najmniej jednego bramkarza, trzech obrońców, dwóch pomocników i jednego napastnika. Punkty są przydzielane za różne aktywności wykonywane przez zawodników. Niektóre z aktywności są punktowane tylko przez konkretne pozycje zajmowane przez zawodników (np. „Strzały obronione" tylko przez bramkarzy) lub różnie punktowanie w zależności od pozycji (Napastnik dostaje 4pkt za gola, a obrońca 6pkt). Punktowanymi aktywnościami są:

- **Czas gry** – zawodnik dostaje punkt za zagranie w meczu i kolejny punkt jeśli zagra co najmniej 60 minut w meczu.
- **Gole** – strzelone bramki przez zawodnika.
- **Asysty** – asysty przy golach strzelonych przez kolegów z zespołu.
- **„Czyste konto"** – jeśli zespół nie stracił bramki to jego zawodnicy defensywni dostają punkty.
- **Gole stracone** – zawodnicy defensywni dostają punkty ujemne jeśli stracili dużo bramek.
- **Gole samobójcze** – za gol do własnej bramki zawodnik dostaje punkty ujemne.
- **Obronione karne** – jeśli bramkarz obroni rzut karny dostaje dodatkowe punkty.
- **Spudłowane rzuty karne** – jeśli zawodnik nie zdobędzie bramki z rzutu karnego dostaje punkty ujemne.

- **Żółte i czerwone kartki** – każdy zawodnik ukarany kartką dostaje punkty ujemne.
- **Strzały obronione** – bramkarze dostają bonusowe punkty za co trzeci obroniony strzał.
- **Bonus** – w grze istnieje też bonus, który dostają trzej najlepsi zawodnicy meczu. Bonus ten jest wyliczany za ogół meczu i ocenia ogólną sprawność zawodnika.

2.4. Ekonomia w FPL

Cena zawodników jak i posiadane pieniądze przez graczy nie są stałą wartością. Fantasy Premier League ma swoja własną ekonomię dzięki, której gracze mogą zarabiać pieniądze. Jest to prosty system znany wszystkim z prawdziwego świata. Im większy popyt na danego zawodnika tym jego cena będzie rosła. Jeśli wielu graczy kupi jakiegoś zawodnika na początku, a okaże się on słaby i zaczną oni go sprzedawać to jego cena również spadnie. Przez to gracze, którzy go posiadają stracą pieniądze kiedy będą próbowali go sprzedać po zaniżonej cenie.

2.5. Czynnik ludzki w FPL

Zawodnicy rozgrywający mecze są to prawdziwi piłkarze. Każdy człowiek może mieć gorszy dzień. To co moje SI próbowało zrobić to było właśnie przewidzenie tego, kiedy jakiś zawodnik będzie miał wzrost czy też spadek formy, patrząc tylko i wyłącznie na statystyki.

FPL ma wprowadzony mechanizm informowania graczy o możliwej lub pewnej absencji zawodnika. Oznacza zawodników, którzy z jakiegoś powodu mają nie zagrać oraz informuje o możliwej absencji zawodnika. Jeśli menadżer jednego z zespołów poinformuje, że ktoś ma kontuzję to gra oznaczy zawodnika jako niegrającego. Jak będzie informacja o tym, że zawodnik się rozchorował to oznaczy go jako prawdopodobnie niegrającego. Często też podawana jest możliwa data powrotu zawodnika.

Ten sezon piłkarski jednak ma w sobie jeszcze jedną przeszkodę. Pandemia wirusa COVID-19 wprowadziła dodatkową zmienną do gry. Zawodnicy są badani bardzo często i jeśli któryś z nich zachoruje to nie ma możliwości gry. Informacja o chorych zawodnikach nie jest publicznie dostępna. Jedynym źródłem informacji są konferencje prasowe menadżerów danych zespołów i czasami nie da się zdobyć informacji kiedy jakiś zawodnik zachorował. Drugim problemem jest tu fakt, że jeśli zachoruje kilku piłkarzy z jednego zespołu to mecz tego zespołu może zostać odwołany i wtedy piłkarze z dwóch zespołów nie grają meczu. To jest czynnik, którego nie da się w żaden sposób przewidzieć.

Rozdział 3.

Tworzenie modeli

3.1. Przygotowanie danych

Fantasy Premier League udostępnia swoje API [4], ale nie ma tam danych historycznych. Jedynie aktualny sezon. Dlatego do swojej pracy użyłem danych archiwizowanych przez jednego z użytkowników FPL [5].

Statystyki piłkarzy używane przez moje modele są to dane, które opisałem w sekcji 2.3 oraz dodatkowo kilka statystyk wprowadzonych przez FPL:

- **Influence** – punkty przyznawane piłkarzowi za to jak duży wpływ ma na rozgrywkę.
- **Creativity** – punkty przyznawane piłkarzowi za to jak dużą kreatywność przedstawia.
- **Threat** – punkty przyznawane piłkarzowi za to, jak dużym zagrożeniem jest dla drużyny przeciwnej.
- **ICT index** – „Influence Creativity Threat Index", czyli zbiorczy wynik dla powyższych statystyk.
- **BPS** – „Bonus Points System", czyli ogólna ocena zawodnika określana przez FPL. Służy do wybrania piłkarzy, którzy dostaną bonusowe punkty.

Formy piłkarzy nie da się określić patrząc tylko na ostatni grany przez nich mecz. Dlatego do swoich modeli używam uśrednionych danych z trzech ostatnich meczów danego zawodnika. Wykluczam również zawodników, którzy nie grają, ponieważ przewaga zawodników, którzy dostają zero punktów za mecz byłaby ogromna.

3.2. Dane wyjściowe

Z założenia moje SI ma wspomóc graczy w wyborze zawodników do swojego składu. Punkty, jakie piłkarze dostają za mecze, jest to dowolna liczba całkowita. Ponieważ występuje tu czynnik ludzki te same dane nie zawsze dają ten sam wynik. Dlatego dla ułatwienia selekcjonowania zawodników podzieliłem ich wynik na 5 grup. Każda z tych grup odzwierciedla, jak duży wpływ dany zawodnik będzie miał na naszą ogólną ilość punktów.

- **„Don't want to have"** – oznacza wynik zawodnika, który wpłynie negatywnie na nasz wynik. Zwykle są to zawodnicy, którzy uzyskują najwięcej ujemnych punktów i przegrywają swoje mecze.
- **„Low impact"** – jest to bardzo niski wynik dodatni. Najczęściej oznacza, że dany zawodnik przegrał mecz. Posiadanie takiego zawodnika sprawi, że spadniemy w rankingu.
- **„Standard score"** – wynik standardowy. Tańsi zawodnicy z tej kategorii są warci posiadania, ponieważ nie kosztują nas dużo, a nie spowodują naszego spadku w rankingu.
- **„Awesome performence"** – zawodnicy z tego przedziału są warci każdej kwoty. Zdobywają wysokie wyniki, które pozwolą nam zdobyć na tyle dużo punktów, że wzniesiemy się w rankingu.
- **„Surperior score"** – jeśli, któryś z zawodników zdobywa na tyle dużo punktów, że trafi do tej kategorii, to posiadanie tego zawodnika jest wymagane. Brak tego zawodnika w naszej drużynie może sprawić, że każdy inny gracz, który go posiada, wyprzedzi nas w rankingu ogólnym.

Użytkownik po uruchomieniu programu z odpowiednio wybranym modelem uzyska plik tekstowy, który będzie miał w sobie nazwiska piłkarzy oraz przewidziane przez dany model wyniki. Pojawią się tam tylko piłkarze, którzy w ostatnim czasie grali swoje mecze.

3.3. Użyte metody

Jak widać, nasze SI wykorzystuje kilkanaście różnych statystyk na wejściu i próbuje sklasyfikować dany obiekt na jedną z pięciu grup, wykorzystując pozyskaną już wiedzę. Jest to typowy problem, z jakim mierzą się różne modele w uczeniu maszynowym. W swojej pracy użyłem trzech różnych modeli:

- **Sieci neuronowych**
- **Regresji liniowej**

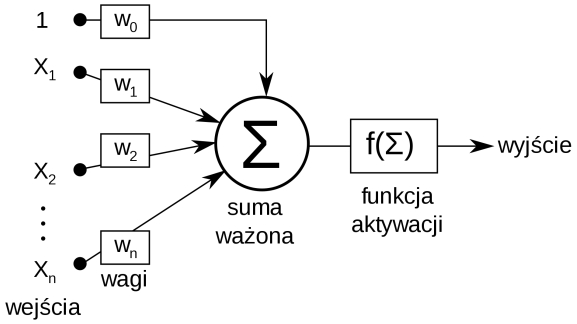
- **Drzew decyzyjnych**

Wszystko napisałem w języku Python, używając dostępnych w nim bibliotek. Głównie używałem PyTorch'a [6] i Scikit-learn [7].

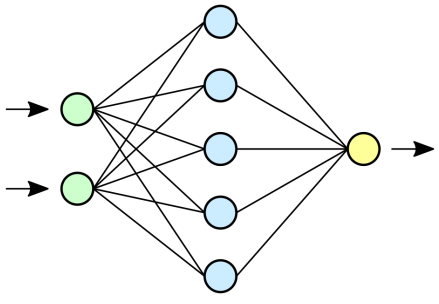
3.4. Sieci neuronowe

Żeby zrozumieć działanie sieci neuronowej, warto najpierw zrozumieć działanie biologicznego neuronu. Neuron jest to najmniejsza część układu nerwowego. Odpowiedzialny jest on za odbieranie bodźców i przekazywanie ich do kolejnych neuronów, aż dojdzie do mózgu, gdzie zostanie on przetworzony jako informacja. Właśnie takie zachowanie chciał odtworzyć w latach 50. *Frank Rosenblatt* tworząc perceptron [8].

Perceptron jest najprostszą siecią neuronową. Składa się on z jednego lub wielu neuronów *McCullocha-Pittsa*. Wskazuje on czy jakiś obiekt, który został podany na wejściu, należy czy też nie do danej klasy. Jak już utworzymy taki perceptron, to musimy najpierw nauczyć go, jak ma reagować na otrzymywane dane. Robimy to poprzez trenowanie go przykładowymi danymi i odpowiednim modyfikowaniem jego parametrów, tak by otrzymać oczekiwane przez nas rezultaty. Jeśli ustawimy wiele perceptronów w kilka warstw, stworzymy perceptron wielowarstwowy.



Rysunek 3.1: Neuron McCullocha-Pittsa [9]



Rysunek 3.2: Sieć neuronowa [11]

Perceptron wielowarstwowy [10] jest najpopularniejszym typem sieci neuronowych. Najczęściej składa się z jednej warstwy wejściowej, kilku warstw ukrytych i jednej warstwy wyjściowej. Każdy węzeł takiej sieci neuronowej łączy się z każdym węzłem z kolejnej warstwy. Wszystkie takie połączenia posiadają swoje wagi. Do uczenia takiej sieci wykorzystywana jest propagacja wsteczna. Po otrzymaniu wyniku, obliczany jest błąd, który propagacja wsteczna minimalizuje za pomocą odpowiedniego zmieniania wag.

Tworząc swoją sieć neuronową, metodą prób i błędów próbowałem ustalić jej najlepsze parametry. Ostateczna sieć miała 4 warstwy. Posiadały kolejno 17, 64, 64 oraz 5 neuronów. Zastosowałem również technikę dropout. Stworzyłem też drugą sieć, która bazuje na pierwszej, ale ma dodatkową warstwę na końcu. Warstwa ta ma imitować zachowanie regresji liniowej. Zamienia wartości z pięciu neuronów z ostatniej warstwy poprzedniego modelu na jeden.

3.5. Regresja liniowa

Regresja liniowa [12] jest metodą, która próbuje przybliżyć oczekiwane wyniki, za pomocą linii. Ogólnym wzorem reprezentującym ten model jest:

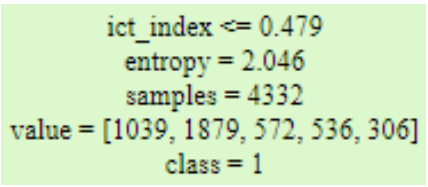
$$y_i = \alpha_0 + \alpha_1 x_{i1} + \alpha_2 x_{i2} + ... + \alpha_n x_{in} + \epsilon_i$$

- y_i - są to wyniki, które regresja próbuje przybliżyć.
- α_i - są to kolejne stałe, które model modyfikuje w taki sposób, by jak najlepiej przybliżyć wynik.
- x_{ik} - kolejne dane wejściowe do naszego modelu.
- ϵ_i - jest to błąd przybliżenia, który otrzymamy podczas estymacji wyniku.

Mój model za dane wejściowe bierze statystyki otrzymane przez piłkarza i próbuje przewidzieć otrzymany przez niego wynik, minimalizując otrzymany błąd.

3.6. Drzewa decyzyjne

Drzewa decyzyjne są to struktury, które pozwalają na podstawie dostępnych informacji określić, jaki będzie wynik. Drzewa, które służą do klasyfikacji, najczęściej są drzewami binarnymi. W węzłach takiego drzewa znajdują się „pytania", które model zadaje naszemu wejściu (np. Czy dany piłkarz strzelił więcej niż dwa gole?). Drzewo się rozrasta do momentu, w którym będzie w stanie określić odpowiedź lub też gdy osiągnie maksymalną wysokość. W liściach takiego drzewa znajdują się przewidywane przez model odpowiedzi.



Rysunek 3.3: Przykładowy węzeł z mojego drzewa

W przykładowym węźle, który widnieje na rysunku po lewej, widać, że dzieli on po „ict index". Widać, że do danego węzła dotarło 4332 piłkarzy i, że większość z nich została sklasyfikowana jako 1, czyli „low impact". W momencie kiedy piłkarz „idąc" daną ścieżką dojdzie do liścia, to mój model zwraca przewidywany wynik, jaki widnieje w tym liściu.

Rozdział 4.

Rezultaty

4.1. Sieci Neuronowe

Sieć neuronowa, w mojej pracy wypadła najgorzej. Uzyskałem tu w najlepszym momencie skuteczność na poziomie 34%. Sieć neuronowa z dodatkową warstwą imitującą regresję liniową wypadła gorzej niż normalna. Wszelkie moje próby zwiększenia skuteczności poprzez modyfikowanie parametrów sieci, czy też zmienianie całego modelu nie dawały lepszych skutków.

Problem, który można zauważyć w mojej sieci, to nadmiar danych, w których piłkarz zdobywa małą ilość punktów. Po wynikach, jakie dostajemy od sieci, można zauważyć, że mój model stwierdza, że najlepszą decyzją będzie zwrócenie zawsze tego samego wyniku. Jest to bardzo niefortunny wynik. Gracz, który by korzystał z takiego modelu, realnie nie zyskuje nic. Próbowałem wyrównać dane i dać sieci do nauki po równo z każdego z wyników, ale wtedy danych jest znacznie mniej i wyniki, jakie daje sieć, nie są lepsze od losowego wybierania wyniku.

4.2. Regresja liniowa

Regresja liniowa wypadła trochę lepiej niż sieć neuronowa uzyskując średnie wyniki około 40% skuteczności. Natomiast tu można zauważyć, że regresja wskazuje zawodników, którzy rzeczywiście zagrali bardzo dobrze, jako tych najlepszych. Sieć neuronowa przypisywała wszystkim te same wyniki. W regresji jest zauważalny podział na kilka klas. Z oczu gracza, który próbuje wybrać odpowiedni skład na kolejny zestaw meczy, jest to znaczne ułatwienie.

4.3. Drzewa decyzyjne

Drzewa decyzyjne w mojej pracy wypadły najlepiej. Po wielu próbach znalezienie optymalnych parametrów udało mi się w najlepszym momencie uzyskać skuteczność około 48%. Taki wynik przy problemie, w którym przez czynnik ludzki istnieje spora losowość, jest całkiem dobrym osiągnięciem.

Moje drzewa miały ustaloną maksymalną głębokość równą piętnaście. Dodatkowo ustawiłem, by moje drzewa dzieliły węzły tylko jeśli sto rekordów napłynęło do nich oraz minimalną ilość próbek, jakie muszą znajdować się w liściu, również ustawiłem na sto.

Prawdopodobnie przez to, że drzewo dzieli względem otrzymanych danych, (np. po strzelonych bramkach przez piłkarza) to podział zawodników na klasy będzie lepszy niż w poprzednich dwóch metodach.

4.4. Wybór losowy

Zupełnie niedoświadczony gracz, który nie zna się na piłce nożnej będzie zmuszony do wyboru losowego. Taki wybór będzie miał skuteczność 20%, ponieważ klasyfikujemy zawodników na pięć grup. Każdy z wymienionych wyżej modeli, osiąga lepszą skuteczność niż model losowy, co świadczy o tym, że znalazły one lepsze rozwiązanie danego problemu. Dlatego też te modele, dla początkujących graczy, mogą okazać się przydatne, ponieważ nie będą oni zmuszeni do całkowicie losowego wyboru swojego pierwszego zespołu.

Warto jednak zaznaczyć, że żaden z modeli nie byłby w stanie konkurować z najlepszymi graczami świata. Dzieje się tak, ponieważ modele korzystają tylko i wyłącznie ze statystyk. Gracze z najwyższych miejsc z rankingu stosują dodatkowe kryteria, oglądają konferencję prasowe i szukają w internecie informacji o formie zawodników.

Rozdział 5.

Wnioski i plany na przyszłość

5.1. Podsumowanie

Moje wyniki pokazują, że używając tylko statystyk, nie jesteśmy w stanie przewidzieć tego, jakie wyniki osiągną realni piłkarze. Losowość jaką wprowadza czynnik ludzki nie jest prosta do przewidzenia wyłącznie za pomocą liczb. Widać również, że modele z tygodnia na tydzień dają różną skuteczność spowodowaną tym, że ludzka forma nie jest deterministyczna. Występuje tu kilka problemów. Oto kilka z nich razem z możliwymi rozwiązaniami:

- **Nadmiar słabo grających zawodników** – nawet posiadając bardzo dużo danych (ja użyłem danych od sezonu 2016/17, czyli ponad 4 lata rozgrywek z ligi angielskiej) nie byłem w stanie zmusić moich modeli, by przewidywały dobrze wszystkich dobrze grających zawodników. Widać, że każdy z modeli faworyzuje nadanie piłkarzom słabych wyników, ponieważ jest to najbardziej prawdopodobny wynik. Próbowałem tak przygotować dane by każdego z wyników było po równo, ale wtedy danych jest mało. Możliwym rozwiązaniem jest skorzystanie ze znacznie bardziej bogatego API, gdzie będzie więcej danych.

- **Możliwa absencja zawodnika przez wypadek losowy** - zdarza się, że dany zawodnik może nie zagrać, bo na treningu przed meczowym doznał kontuzji, albo trener zespołu stwierdzi, że dany zawodnik musi odpocząć, albo zdarzył się jakiś inny wypadek losowy. Możliwości nieobecności piłkarza jest wiele. W tym sezonie również przez wirusa COVID-19. Często jedynym źródłem informacji o takich absencjach są konferencje. To jest problem, którego sztuczna inteligencja nie rozwiąże. Możliwym rozwiązaniem byłoby stworzenie aplikacji, w której użytkownik informowałby SI o możliwej absencji zawodnika.

- **Czynniki zewnętrzne wpływające na grę zawodnika** - istnieją również dodatkowe informacje, które najlepsi gracze FPL wykorzystują do przewidywania tego, jak pójdzie danym piłkarzom. Małe odstępy między meczami po-

wodują zmęczenie zawodników. Absencja kluczowego zawodnika wpływa na cały zespół i słabiej grają wtedy jego koledzy z zespołu. Dobra forma drużyny przeciwnej sprawia, że dany zawodnik będzie miał trudniejszy mecz. Te, oraz wiele innych dodatkowych informacji tego typu jest wykorzystywanych przez najlepszych graczy FPL. Są to problemy, które da się rozwiązać pisząc kod, ale konieczna jest konsultacja z najlepszymi graczami w celu ustalenia co jest ważne przy wybieraniu składu.

5.2. Plany na przyszłość

Pierwszym krokiem do rozwoju projektu byłoby znalezienie nowego API z danymi piłkarzy. Powiększenie ilości danych wpłynie pozytywnie na wyniki. W dalszych krokach warto rozważyć inne modele od zaproponowanych przeze mnie. Możliwe, że inne metody dadzą lepsze wyniki. Następnie należałoby rozwinąć samą aplikację. Wprowadzenie dodatkowych zmiennych, które wchodziłyby w skład statystyk służących predykcji, da modelom więcej danych, względem których wykonywać będą prognozy. Dodatkowo można przygotować interfejs użytkownika, aby aplikacja stała się prostsza w użyciu. Ostatnim krokiem byłoby stworzenie bota, który sam próbowałby grać w FPL, zamiast tylko wspomagać gracza.

Bibliografia

[1] Andre Miguel Leitao Santos. *Monte Carlo Tree Search Experiments in Hearthstone.* 2017. https://www.researchgate.net/profile/Andre_Santos62/publication/317904983_Monte_Carlo_Tree_Search_experiments_in_Hearthstone/links/5977b48aa6fdcc30bdbadc2a/Monte-Carlo-Tree-Search-experiments-in-Hearthstone.pdf.

[2] Łukasz Grad. *Helping AI to Play Hearthstone using Neural Networks.* 2017. https://annals-csis.org/proceedings/2017/drp/pdf/561.pdf.

[3] Fantasy Premier League. https://fantasy.premierleague.com.

[4] Fantasy Premier League API. https://fantasy.premierleague.com/api/bootstrap-static/.

[5] GitHub użytkownika *vaastav* z archiwalnymi statystykami z FPL. https://github.com/vaastav/Fantasy-Premier-League.

[6] Strona główna PyTorch. https://pytorch.org.

[7] Strona główna SKLearn. https://scikit-learn.org/stable/index.html.

[8] Perceptron. https://pl.wikipedia.org/wiki/Perceptron.

[9] Neuron McCullocha-Pittsa. https://pl.wikipedia.org/wiki/Neuron_McCullocha-Pittsa.

[10] Perceptron wielowarstwowy. https://pl.wikipedia.org/wiki/Perceptron_wielowarstwowy.

[11] Sieć neuronowa. https://pl.wikipedia.org/wiki/Sie%C4%87_neuronowa.

[12] Regresja liniowa. https://pl.wikipedia.org/wiki/Regresja_liniowa.