Zeszyty Naukowe WWSI, No 20, Vol. 13, 2019, s. 29-40
DOI: 10.26348/znwwsi.20.29

Hybrydowy system rekomendacji planów treningowych

Maciej Kaczanowski[1]

Instytut Systemów Informatycznych, Wydział Cybernetyki,
Wojskowa Akademia Techniczna, Warszawa, Polska

Abstrakt

Hybrydowe systemy rekomendacji łączą zalety metod stosowanych powszechnie w rekomendacji. Głównym celem tego artykułu jest przedstawienie zastosowania uczenia maszynowego do budowy hybrydowego silnika rekomendacji. Uczenie maszynowe jest poddziedziną sztucznej inteligencji, która wykazuję obiecujące rezultaty w klasyfikacji, predykcji, wykrywaniu anomalii i rekomendacji. W tym artykule zaproponowano koncepcję spersonalizowanego modelu systemu rekomendacji opartego na parametrach i planach treningowych sportowców. Badania przeprowadzono w środowisku chmurowym Microsoft Azure Machine Learning Studio na zbiorze danych wygenerowanym na podstawie danych referencyjnych.

Słowa kluczowe **–** uczenie maszynowe, sztuczna inteligencja, nauka o danych, hybrydowe systemy rekomendacji, Microsoft Azure Machine Learning, język programowania Python

[1] E-mail: maciej.kaczanowski@wat.edu.pl

Zgłoszono do druku 10 marca 2019 r.

Wprowadzenie

W dzisiejszym świecie nowe technologie wspierają bezpośrednio lub pośrednio cały proces rywalizacji sportowej. Aby konkurować z zawodnikami znajdującymi się w czołówkach różnorakich rankingów: krajowych, kontynentalnych, światowych, a czasami nawet o mniejszym zasięgu, nieuniknione jest wzmacnianie rozwoju sportowca osiągnięciami medycyny oraz techniki.

Najbardziej utalentowani i pracowici sportowcy bez dostępu do najnowszych, precyzyjnych narzędzi informatycznych mogą nie mieć równych szans z konkurentami wspomagającymi komputerowo swój rozwój. Jednym ze sposobów wsparcia procesu decyzyjnego wyboru planu treningowego może być system rekomendacji.

Zadaniem systemu rekomendacji jest przetworzenie informacji w celu dostarczenia użytkownikom pomocnych dla nich wskazówek, w postaci naturalnej, tzn. zrozumiałej dla człowieka, dotyczących produktów lub usług, które mogą być dla nich w przyszłości użyteczne. W omawianym przypadku, gdy użytkownikiem systemu jest sportowiec, to rekomendowanym produktem będzie plan treningowy dostosowany do jego potrzeb.

1. Metody rekomendacji

Rekomendacje wykorzystujące techniki pamięciowe opierają się na danych i relacjach pomiędzy nimi. Ten typ systemów używa metod takich, jak: wspólna filtracja (ang. *collaborative filtering*), filtrowanie w oparciu o informacje o obiektach (ang. *content-based*), opartych na wiedzy (ang. *knowledge-based*). Cechą, która łączy te metody, to badanie podobieństw i odmienności użytkowników i obiektów. W zależności od cech obiektów/obserwacji, sposoby obliczania współczynników podobieństwa i odmienności dzieli się na:

- porządkowe;
- jakościowe;
- binarne;
- skategoryzowane;

W metodzie wspólnej filtracji zakłada się, że preferencje użytkownika można prognozować na podstawie jego i innych opinii, wyrażonych za pomocą ocen. Druga

metoda swoje działanie opiera na metadanych obiektów. Rekomenduję ona użytkownikowi obiekty podobne do tych, które w przeszłości wysoko ocenił. Bardziej złożone systemy rekomendacji wykorzystują trzecią metodę, która bazuję na specjalistycznej wiedzy o preferencjach użytkownika dotyczących konkretnych cech obiektów. Podejście bazujące na atrybutach profilu użytkownika, np. na geolokalizacji, nazywa się demograficznym.

Każda z tych metod zawiera szereg zalet i wad, więc ich użycie należy dopasować do konkretnej sytuacji. Koncepcja, która łączy wszystko, co najlepsze w tych metodach, nazywa się mieszaną albo hybrydową.

2. Rekomendacje treningu dla sportowców

System do rekomendacji treningu sportowców powinien realizować następujące funkcjonalności:

Definiowanie profilu sportowca – wyniki rekomendacji dostosowane są do danej osoby na podstawie jej profilu, który może zawierać wiek, płeć, poziom zaawansowania, historię treningu, dane wydolnościowe, psychologiczne, antropometryczne itp.

Umożliwianie pomocy innym – sportowcy i trenerzy powinni mieć możliwość wspierania swoją wiedzą mniej doświadczone osoby. Wiedza ta mogłaby posłużyć do oceny planów treningowych, a nawet ich prawidłowej klasyfikacji pod względem poziomu trudności i dostosowania do aktualnego stanu formy sportowca. Może to stanowić dobry sposób na dostarczanie danych do systemu.

2.1. Modelowanie

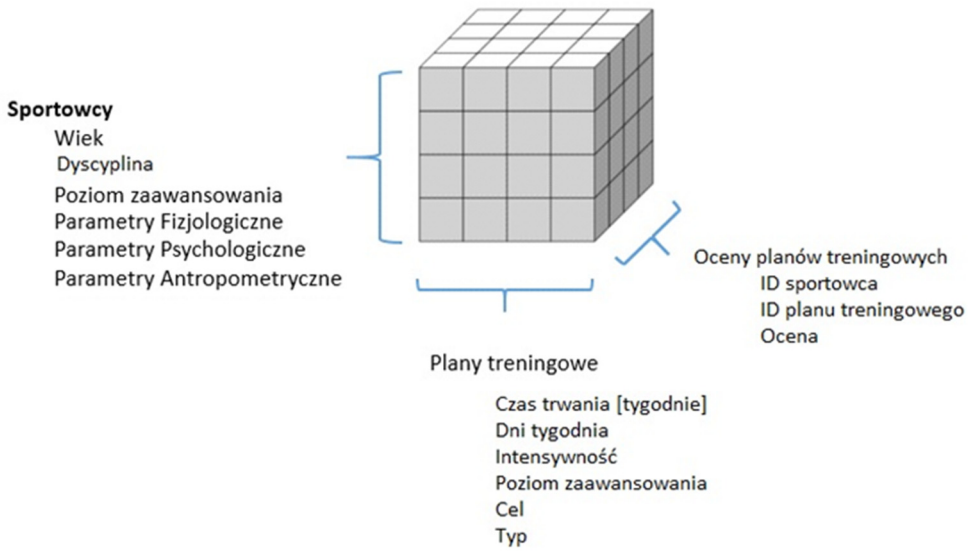
Oznaczmy zbiór sportowców za pomocą S, a zbiór planów treningowych za pomocą P. Oceny wystawione planom treningowym przez sportowców w systemie należą do zbioru O $\{o(s_i,p_i)\}$, a możliwe wartości ocen zawierają się w W, np. $W=\{1..5\}$. Każdy sportowiec może wystawić tylko jedną ocenę O_{sp} dla planu treningowego $p \epsilon P$. Plany treningowe ocenione przez sportowców s_1 i s_2 zawieraj się w $P_{s1} \cap P_{s2}$, a użytkownicy, którzy ocenili p_1 i p_2 w $U_{p1} \cap U_{p2}$. Celem silnika rekomendacji jest predykcja $f(s_i,p_i)$ oceny sportowca s_i dla nowego planu treningowego p_i, którego w przeszłości nie ocenił.

2.2 Silnik rekomendacji

System hybrydowy na potrzeby niniejszego opracowania został zbudowany w oparciu o model uczący się, przy wykorzystaniu Azure ML Studio i języka programowania Python. Do trenowaniu modelu użyto trzy zestawy danych, które zostały wygenerowane na potrzeby przeprowadzenia eksperymentu.

Zestawy danych:

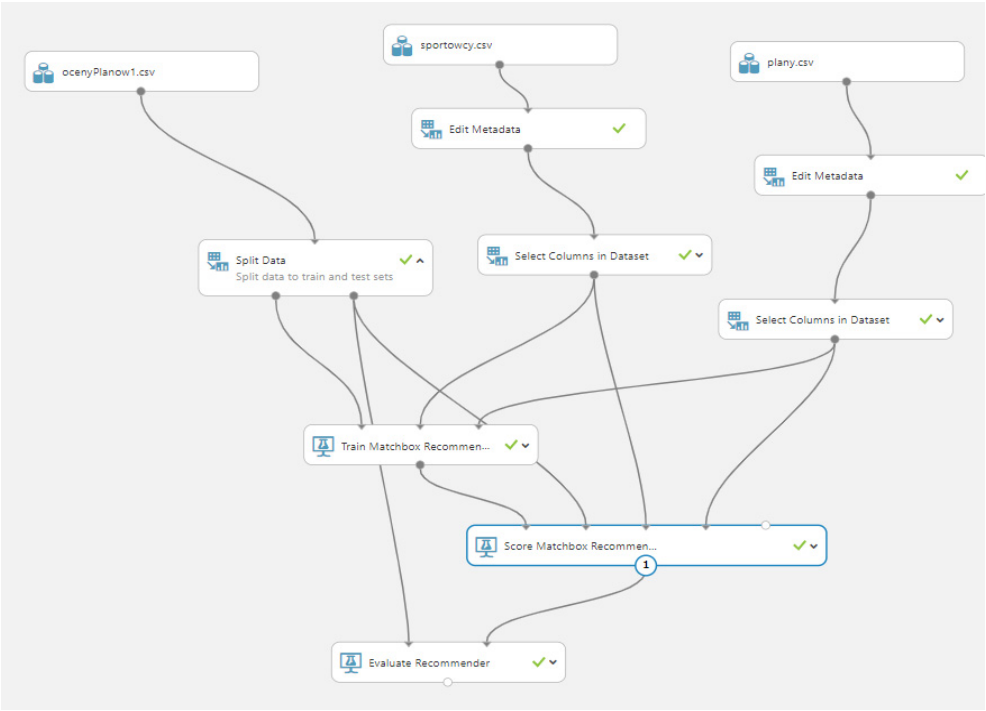
- Oceny – składający się z trzech kolumn: id sportowca, id planu treningowego, ocena (1-5).
- Sportowcy – wiek, dyscyplina, poziom zaawansowania wyrażony w latach uprawiania sportu, parametry fizjologiczne, psychologiczne oraz antropometryczne.
- Plany treningowe – czas trwania, liczba dni w tygodniu, intensywność, poziom zaawansowania, cel, typ. Każdy plan treningowy składa się z różnych treningów, których dane przechowywane są w bazie danych.



Rysunek 1. Zbiór danych użyty do eksperymentów

Zastosowany algorytm w modelu to „Matchbox Recommender" [1], [2]. Algorytm bazuję na klasyfikatorze bayesowskim, który osiąga dobre rezultaty dla danych

wielowymiarowych. Załóżmy, że system otrzymuję krotkę (x, y, ϕ, r), gdzie $x \in R^n$ – dane użytkowników, $y \in R^m$ – dane obiektów, $r \in R$ – oceny, $\phi \in R^f$ – kontekst ocen. Wektor cech użytkownika oznaczmy przez $s = Ux$ (U to macierz $K \times n$ w której elementy to u_{ki}), wektor obiektów $t = Vy$ (V macierz $K \times m$), a odchylenie $b = \phi^T w$ (w to ukryty zestaw wag). Często b to kombinacja cech opisowych użytkownika x i obiektu y, $b = x^T u + y^T v$, u i v to obciążenia. Ocena r jest wyliczana w sposób następujący $p = (r|s, t, b) = N(r|s^T t + b, \beta^2)$, β – odchylenie standardowe szumu informacyjnego. Ocena jest proporcjonalna do długości wektorów cech $|s|$ i $|t|$ tak samo jak i do cosinusa kąta pomiędzy nimi. W przypadku, gdy nie mamy danych użytkowników i obiektów to algorytm wykorzystuję tylko wspólną filtrację. Po wytrenowaniu modelu i skonfigurowaniu serwisu internetowego (ang. *Web Service*) użyto go w aplikacji napisanej w języku Python.



Rysunek 2. Model uczący hybrydowego silnika rekomendacji

3. Pomiar odległości i identyfikacja odmienności

Wiele algorytmów rekomendacji bazuję na pomiarze odległości pomiędzy użytkownikami oraz rekomendowanymi przedmiotami, aby wyznaczyć ich stopień podobieństwa. Obiekty mające cechy porządkowe przyjmujące wartości liczbowe można przedstawić w n-wymiarowym układzie ujednoliconych współrzędnych, a następnie wyznaczyć ich podobieństwa i odmienności poprzez pomiar odległości.

Najprostszym pod względem obliczeniowym jest pomiar odległości euklidesowej:

$$d(x,y) = \sqrt{\sum_{k=1}^{n}(x_k - y_k)^2}. \tag{1}$$

gdzie x_k i y_k– to atrybuty obiektów x i y, a n to liczba atrybutów obiektów (wymiarów). Jest to szczególny przypadek odległości Minkowskiego [3]:

$$d(x,y) = \left(\sum_{k=1}^{n}|x_k - y_k|^r\right)^{\frac{1}{r}}. \tag{2}$$

W powyższej formule r oznacza stopień odległości. I tak, jeżeli przyjąć wartość $r = 1$, odległość Minkowskiego staje się tzw. odległością miejską (albo odległością Manhattan). Dla $r = 2$ równanie (2) sprowadza się do postaci (1) i opisuje odległość euklidesową. Jeżeli założyć, że $r \to \infty$, z równania (2) uzyskuje się odległość Czebyszewa:

$$d(x,y) = \lim_{r\to\infty}\left(\sum_{k=1}^{n}|x_k - y_k|^r\right)^{\frac{1}{r}} = \max_k|x_k - y_k|. \tag{3}$$

Jeszcze inna metoda pomiaru to odległość Mahalanobisa:

$$d(x,y) = \sqrt{(x-y)C^{-1}(x-y)^T}, \tag{4}$$

gdzie C jest macierzą kowariancji.

Dosyć często w pracach badawczych wykorzystuję się cosinusową miarę podobieństwa wektorów. Cosinus kąta pomiędzy wektorami znajduje się jako stosunek iloczynu skalarnego tych wektorów do iloczynu ich długości:

$$\cos(x,y) = \frac{x\circ y}{\|x\|\cdot\|x\|}. \tag{5}$$

We wzorze (5) oznaczenia ‖x‖ i ‖y‖ reprezentują normy euklidesowe (długości) wektorów x i y.

W przypadku zmiennych losowych *x* i *y*, jako miarę ich podobieństwa stosuje się współczynnik korelacji Pearsona r_{xy}:

$$r_{xy} = \frac{cov(x,y)}{\sigma_x \cdot \sigma_y}, \tag{6}$$

gdzie $cov(x, y)$ oznacza kowariancję zmiennych x i y, zaś σ_x i σ_y odpowiednio, oznacza ich odchylenia standardowe.

W celu sprowadzenia pomiarów odległości i odmienności do przedziału <0,1> dokonuje się normalizacji ocen w zakresie cech porównywanych obiektów/obserwacji. Do normalizacji można wykorzystać czynnik maksymalnej odległości, który można wyznaczyć w danej przestrzeni.

W przypadku obserwacji/obiektów, których atrybuty mają wartości binarne {0,1}, można posłużyć się parametrami porównawczymi przedstawionymi w tabeli 1.

Tabela 1. Parametry porównawcze obiektów X i Y

	Obiekt Y		
Obiekt X		0	1
	0	N	NP
	1	PN	P

Poszczególne elementy tej tabeli mają następujące interpretacje:

P – liczba cech przyjmujących wartość 1 dla obydwu obiektów/obserwacji;

N – liczba cech przyjmujących wartość 0 dla obydwu obiektów/obserwacji;

PN – liczba cech przyjmujących wartość 1 dla obiektu X, a 0 dla Y;

NP – liczba cech przyjmujących wartość 0 dla X i 1 dla Y.

Przy ich pomocy można oszacować różne miary podobieństwa lub odległości (niepodobieństwa) [4]:

- **Współczynnik Jaccarda:**

$$WspJac = \frac{P}{P+PN+NP}. \tag{7}$$

- **Odległość Jaccarda:**

$$dJac = \frac{PN+NP}{P+PN+NP}. \tag{8}$$

- **Współczynnik prostego dopasowania/ prostego porównania zbiorów:**

$$WPD = \frac{P+N}{P+PN+NP+N}. \tag{9}$$

- **Odległość Hamminga:**

$$H(X,Y) = \sum_{k=1}^{n} h(i) = PN + NP. \tag{10}$$

gdzie $h(i) = \{0,1\}$, 0 – jeśli $x_k = y_k$, 1 – w przeciwnym przypadku.
Przykład: X = 10011, Y = 00001, n = 5 to H(X,Y) = 2

4. Walidacja modelu

Należy pamiętać, że model stanowi uproszczoną postać obserwacji. Przed doborem zbioru uczącego i testowego powinniśmy przyjąć pewne założenia. Zgodnie z twierdzeniem o nieistnieniu darmowych obiadów (ang. *No Free Lunch Theorem*) [5], nie istnieje żaden model, który będzie działał lepiej dla pewnej klasy problemów. Każdy z przygotowywanych modeli dla konkretnych przypadków trzeba testować i szczegółowo oceniać, aby wybrać dla nich najlepszą konfigurację – przygotowanie danych, algorytm uczący oraz odpowiedni dobór hiperparametrów, czyli parametrów algorytmu uczącego pozwalających sterować procesem uczenia.

W praktyce przetestowanie wszystkich konfiguracji jest niewykonalne, dlatego na początku eksperymentów przyjmuję się założenia i sprawdza do nich dopasowane konfiguracje. Można rozróżnić kilka sposobów walidacji modelu, w zależności od sposobu podziału dostępnych danych na treningowe i testowe:

Prosta walidacja – polega na podzieleniu zbioru na dane uczące oraz dane testowe. Zbiór testowy zawiera zazwyczaj nie więcej niż 30% wszystkich obiektów. Model budowany jest na podstawie danych treningowych, a sprawdzany danymi testowymi.

Walidacja krzyżowa K-krotna (ang. *K-fold cross validation*) – wszystkie dane dzielone są na K rozłącznych, równych objętościowo zbiorów. Proces trenowania zostaje przeprowadzony K razy, gdzie jeden wyłączony z K zbiorów pełni rolę zbioru testującego, zaś pozostałe łącznie stanowią zbiór trenujący. Każdorazowo przeprowadzane są testy, a ostateczny wynik jest uśredniany.

Dzięki takiemu podejściu każdy obiekt treningowy zostanie dokładnie raz przetestowany, a z drugiej strony wszystkie dane mogą zostać użyte do budowy modelu.

Jeżeli OT jest testowym zbiorem danych ocen planów treningowych, a $|OT|$ jest licznością tego zbioru, to do walidacji modelu można zastosować następujące metryki [6]:

- MAE (ang. *Mean Absolute Error*)

$$MAE(f) = \frac{1}{|OT|} \sum_{o_{sp} \in OT} |f(s_i, p_i) - o_{sp}| \tag{11}$$

- RMSE (ang. *Root Mean Squared Error*)

$$RMSE(f) = \sqrt{\frac{1}{|OT|} \sum_{o_{sp} \in OT} \left(f(s_i, p_i) - o_{sp}\right)^2}. \tag{12}$$

4.1 Miary rankingu najlepszych n rekomendacji (ang. *top-N*)

Do znalezienia rankingu najlepszych rekomendacji można stosować różnie zdefiniowane miary [7]:

- DCG (ang. *Discounted Cumulative Gain*) DCG mierzy użyteczność obiektów na każdej pozycji listy rankingowej. Obiekty na początku listy są ważniejsze od pozostałych. Wartość tej metryki mieści się w przedziale <0,1>, a im bliżej wynik znajduję się 1 tym model jest lepszy.

$$DCG(O, s) = \sum_p \frac{r_{sp}}{disc(p)}, \quad disc(p) = \begin{cases} 1, & p \leq 2 \\ log_2 p, & p > 2 \end{cases} \tag{13}$$

- DCGperfect (czasem nazywany IDCG) – na początku sortuje się otrzymane oceny, a następnie korzysta się z tego samego wzoru, co DCG.
- NDCG (ang. *Normalized Discounted Cumulative Gain*):

$$NDCG = \frac{DCG}{DCG_{perfect}} \tag{14}$$

W środowisku Azure pojawiają się jeszcze dwa oznaczenia metryki NDCG w zależności od wykorzystywanego pomiaru odległości:

a) L1Sim NDCG – odległość Manhattan (2).
b) L2 Sim NDCG – odległość euklidesowa (1).

W Azure ML do oceny algorytmu rekomendacji wykorzystuję się blok „Score Matchbox Recommender". W zależności od wyboru rodzaju rekomendacji wyliczane są w nim automatycznie odpowiednie metryki. Uzyskane wartości metryk dla testowanego modelu przedstawiono poniżej.

a) Predykcja ocen
   MAE: 1,69
   RMSE: 2,1
b) Rekomendacja planów treningowych
   NDCG: 0,827
c) Podobne plany treningowe
   L1 Sim NDCG = 0,35
   L2 Sim NDCG = 0,33.

Wartość L1 i L2 podobnych planów treningowych wyliczana jest w sposób następujący [8]:

- Znajdywani są wszyscy użytkownicy, którzy ocenili dwa obiekty będące w tej iteracji przedmiotem zainteresowania.
- Tworzone są dwa wektory dla obiektów.
- Obliczane są odległości pomiędzy tymi wektorami L1 (2) oraz L2 (1).
- Obliczane są L1 Sim NDCG oraz L2 Sim NDCG używając wartości zysku wszystkich powiązanych obiektów.
- Wyznaczane jest NDCG dla wszystkich interesujących obiektów w zbiorze danych.

d) Podobni sportowcy – obliczenia wykonywane są analogicznie do podobnych planów treningowych.
   L1Sim NDCG = 0,171
   L2 Sim NDCG = 0,163.

Podsumowanie

Hybrydowy algorytm rekomendacji jest efektywnym narzędziem do rozwiązywania złożonych problemów w sytuacjach, gdzie pojedyncze metody uzyskują słabe wyniki. Podejście to rozwiązuję problem zimnego startu, który występuję, gdy

pojawia się nowy użytkownik i system nie ma podstaw do wyznaczenia jego sąsiedztwa, a zatem nie potrafi dostarczyć mu rekomendacji.

W artykule przedstawiono zastosowanie hybrydowego systemu rekomendacji do wspomagania sportowców przy wyborze odpowiedniego dla nich planu treningowego. Wszystkie eksperymenty przeprowadzono w środowisku Microsoft Azure Machine Learning Studio. Dane do eksperymentów wygenerowano za pomocą autorskich skryptów Python bazując na danych referencyjnych [9].

Bibliografia

[1] D. Stern, R. Herbrich, Th. Graepel, *Matchbox: Large Scale Online Bayesian Recommendations*, International World Wide Web Conference, 2009

[2] *Github* [Online], https://github.com/MicrosoftDocs/azure-reference-other [02.03.2019]

[3] A. Ameljańczyk, *Metryki Minkowskiego w tworzeniu uniwersalnych algorytmów rankingowych*, „Biuletyn WAT", Vol. LXIII, Nr 2, 2014

[4] A. Ameljańczyk, *Modele podobieństwa w komputerowych systemach diagnostyki medycznej i ich wpływ na wiarygodność rozpoznania*, Seminarium GM, 2014

[5] D.H. Wolpert, W.G. Macready, *No Free Lunch Theorems for Optimization*, "IEEE Transactions on Evolutionary Computation" Vol. 1, No. 1, 1997

[6] *Prediction Accuracy Metrics*, Coursera [Online], https://www.coursera.org/lecture/recommender-metrics/prediction-accuracy-metrics-LGLb3 [02.03.2019]

[7] *Rank-Aware Top-N Metrics*, Coursera [Online], https://www.coursera.org/lecture/recommender-metrics/rank-aware-top-n-metrics-Wk98r [02.03.2019]

[8] *Evaluate Recommender*, Microsoft documents [Online], https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/evaluate-recommender [02.03.2019]

[9] D.A. Santos, J.A. Dawson, C.N. Matias, P.M. Rocha, C.S. Minderico, D.B. Allison, L.B. Sardinha, A.M. Silva, *Reference Values for Body Composition and Anthropometric Measurements in Athletes*, "PLOS ONE", 2014

Training plans hybrid recommender system

Abstract

Hybrid recommendation systems combine the advantages of commonly used methods in recommendations. This main objective of this article is to present application of machine learning to build a hybrid recommendation engine. Machine learning is subdomain of artificial intelligence that show promising results in classification, prediction, anomaly detection and recommendations. This paper proposed a personalized recommendation system model based on athletes parameters and training plans. The researches were carried out in the cloud environment Microsoft Azure Machine Learning Studio on football data set.

Keywords – machine learning, artificial intelligence, data science, hybrid recommender, Microsoft Azure Machine Learning, Python programming language