Wstęp

Jednym z typowych zastosowań metod uczenia maszynowego są zagadnienia analizy zachowań, w tym analizy logów. Zagadnienie to może dotyczyć zarówno problemu analizy logów urządzeń w celu znalezienia typowych występujących w postaci sekwencji ciągów zdarzeń prowadzonych do określonych awarii, bądź też analizy logów zachowań użytkowników portali internetowych. Wydobyta w ten sposób wiedza może przynieść szereg istotnych informacji użytkownikom systemu lub przede wszystkim kadrze zarządzającej.

Przykładowo analiza logów z serwerów WWW polegające na prześledzeniu zachowań różnych użytkowników może znaleźć zastosowanie w:

- umożliwia zbadanie typowych zachowań klientów portali internetowych, wyznaczając typowe grupy zachowań użytkowników
- umożliwia analizę sekwencji odwiedzanych stron internetowych, co może pozwolić i ułatwić ewentualne problemy pozycjonowania określonych treści
- automatyczną kategoryzację użytkowników pozwalając na przydzielenie użytkownika do określonych grup w celu poprawnego „targetowania" treści reklamowych

Powyżej przedstawiono jedynie wybrane zagadnienia związane z zagadnieniem analizy logów. Jednak na poprawne działanie systemów składa się wiele etapów przetwarzania danych, co będzie tematem niniejszego laboratorium. Drugim elementem podjętym w laboratorium będą zagadnienia wizualizacji danych i automatycznego generowania raportów.

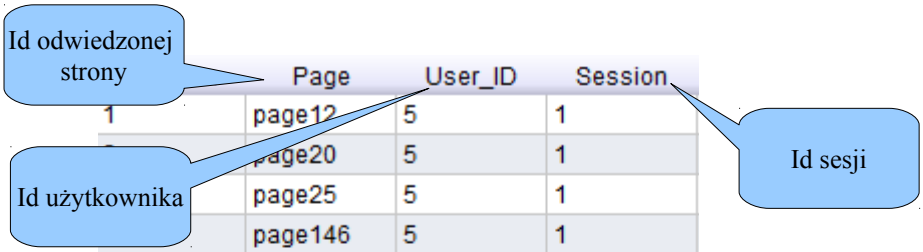
Automatyczne przetwarzanie logów + zadania

Logi zwykle przedstawiane są w postaci tablicy typu klucz wartość, gdzie kluczem jest np. *id_użytkownika, id_sesji a wartością id_strony* odwiedzonej przez tego użytkownika, lub też, w przypadku logów urządzeń są to *id_typu_urządzenia, id_urządzenia, id_zdarzenia.*

Przetwarzanie tego typu danych przez algorytmy inteligencji obliczeniowej wymaga wstępnego przedstawienia ich do postaci tabeli w której wiersze reprezentują pojedyncze sesje danego użytkownika (urządzenia), a kolumny odpowiadają odwiedzinym przez tego użytkownika stronom internetowym (możliwym błędom).

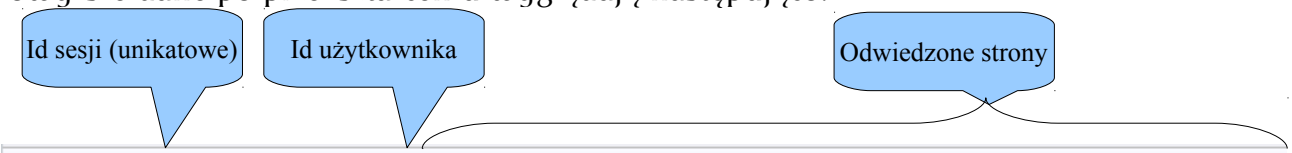
Przykład:

Oryginalne dane:



	Page	User_ID	Session
1	page12	5	1
	page20	5	1
	page25	5	1
	page146	5	1
5	page11	14	2
6	page121	14	2
7	page125	14	2
8	page11	14	3
9	page121	14	3
10	page125	14	3
11	page11	14	4
12	page121	14	4
13	page125	14	4
14	page11	14	5
15	page19	14	5
16	page62	14	5
17	page68	14	5
18	page11	14	6

Powyższe dane po przekształceniu wyglądają następująco:

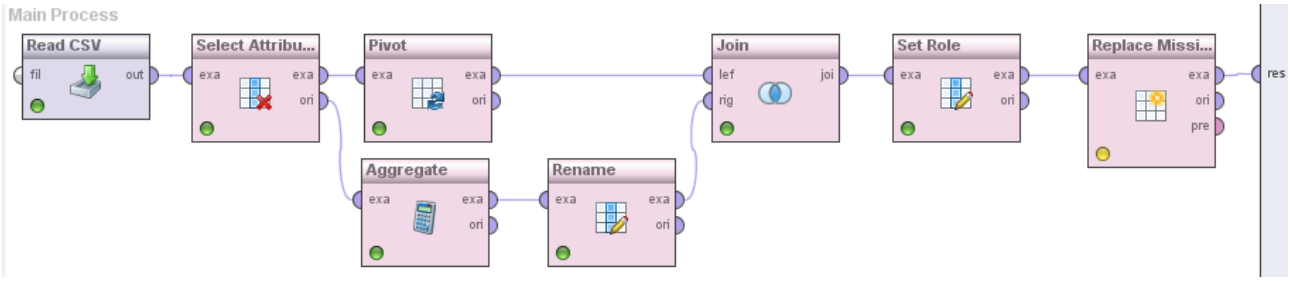


Row No.	Session	User_ID	Visited_page12	Visited_page146	Visited_page20	Visited_page25	Visi
1	1	5	1	1	1	1	0
2	2	14	0	0	0	0	1
3	3	14	0	0	0	0	1
4	4	14	0	0	0	0	1
5	5	14	0	0	0	0	1
6	6	14	0	0	0	0	1
7	7	14	0	0	0	0	1

Rys 1: Tabela danych przygotowana do analizy

Zad 1 Transformacja danych

Stwórz proces dokonujący konwersji danych opisanej powyżej. Dla ułatwienia skorzystaj z poniższego przykładowego procesu:



w pierwszym kroku dane wczytywane są z pliku, następnie odfiltrowywane są dane tak, aby były jedynie niezbędne atrybuty do dokonania konwersji (ze zbioru danych usuwana jest kolumna *user_id*) na tak przygotowanych danych dokonywana jest

konwersja. Problemem jest jednak to, iż w wyniku tracona jest informacja o użytkownikach przypisanych do poszczególnych sesji. Dlatego też konieczne jest połączenie (doklejenie) do wynikowego zbioru danych kolumny *user_id*. W tym celu na wstępie należy zredukować i pozostawić jedynie unikatowe wartości *user_id*. Czynność tą można wykonać na kilka sposobów, przykładowo wykorzystując operator Aggregate, dokonując agregacji zmiennej *user_id i session,* lub też korzystając z operatora *Remove Duplicates.* Następnie po wykonaniu operacji *Join* dobrze jest oznaczyć atrybuty *session* oraz *user_id* jako atrybuty specjalne. Ostatnim krokiem jest usunięcie wartości brakujących. W rezultacie przeprowadzonych operacji uzyskujemy wynik jak na Rys 1.

Wynikowy proces w postaci pliku XML umieść w sprawozdaniu.

Uwaga: wynikowy zbiór danych (po wszystkich transformacjach) powinien składać się z 14422 wektorów oraz 139 atrybutów regularnych i 2 specjalnych

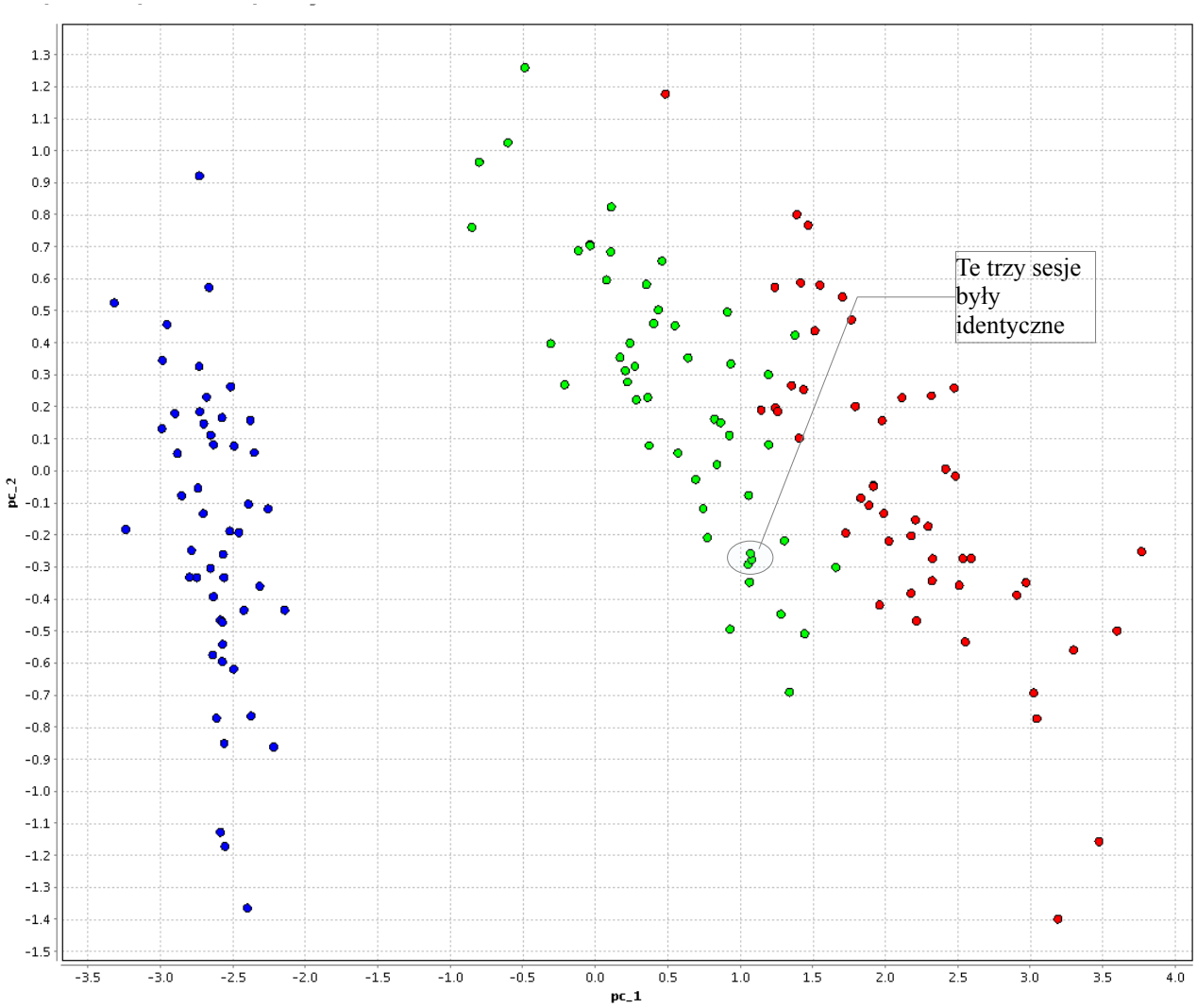
Zad. 2 Wizualizacja danych

Jednym z typowych narzędzi wykorzystywanych do analizy logów jest ich wizualizacja, czyli zapewnienie możliwości wizualizacji informacji zawartych w logach w postaci wykresu 2D lub 3D tak aby użytkownik mógł naocznie stwierdzić jak wygląda rozkład danych z logów. Innymi słowy chodzi o to, aby na wykresie 2D można było w postaci punktów nanieść każdą z grup zdarzeń (pojedynczy wektor/wiersz uzyskany w zadaniu 1). W nawiązaniu do zadania 1, pojedynczy punkt będzie wówczas reprezentował pojedyncza sesję użytkownika. Dodając do wykresu kolorowanie punktów zależne od ID użytkownika można naocznie stwierdzić na ile podobne do siebie są zachowania poszczególnych użytkowników. Patrz poniższy rysunek Rys 2.

Aby uzyskać opisany rezultat w RapidMinerze należy do wyjścia opisanego w zadaniu 1 zbioru danych podpiąć operator PCA (ang. Principle Component Analysis), który odpowiedzialny jest za przeprowadzenie redukcji wymiarowości. Redukcja wymiarowości to przejście z dowolnej n-wymiaorowej przestrzeni (tutaj 139 wymiarowej gdyż każdy punkt/wektor opisany jest przez 139 wartości) do przestrzeni mniej wymiarowej np. 2D tak aby zapewnić minimalizację błędu odwzorowania, tzn aby zminimalizować utratę informacji (zawsze redukując wymiarowość musimy się liczyć z nieodwracalną stratą informacji). Po dodaniu operatora ustaw *dimensionality reduction* na *fixed number* oraz *number of components* ustaw na 2.

Uwaga: często może dojść do sytuacji w której wiele punktów (wiele sesji) jest względem siebie identycznych. Aby móc je zobaczyć należy dodać szum o niewielkiej amplitudzie. Powyższe realizujemy na wykresie zmieniając wartość parametru *Jitter*

W sprawozdaniu umieść uzyskane wyniki oraz pokaż główne obszary w których wystąpiły identyczne sesje.



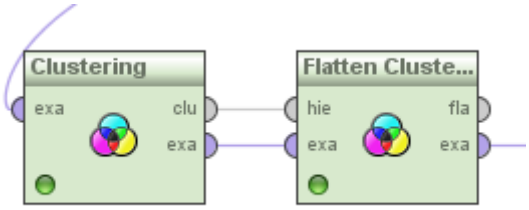
Rys 2: Wynik wizualizacji danych dot. logów

Zad 3. Grupowanie logów

Jedną z metod mających zastosowanie jest grupowanie danych. Zgodnie z przedstawionymi wcześniej założeniami grupowanie danych można przeprowadzić z wykorzystaniem wielu różnych metod i o ile zbiór danych nie jest zbyt duży sugerowanym rozwiązaniem jest zastosowanie grupowania hierarchicznego (Agglomerative Clustering). Zastosowanie metod grupowania pozwoli na odkrycie typowych grup użytkowników, którzy cechują się podobnymi zachowaniami (odwiedzają podobne strony). Taka informacja może być wykorzystana np. do wyświetlania reklam odpowiednich dla danej grupy użytkowników.

Należy jednak uważać na to by liczba wygenerowanych klastrów była dostatecznie duża co zapewni homogeniczność użytkowników wewnątrz klastra (w powyższym zadaniu jest to około 30 klastrów)

W tym celu dodaj operator *K-means* Clustering, (jest on wykorzystany ze względu na to iż jest znacznie szybszy niż Agglomerative Clustering, jednakże należy jest on znacznie mniej dokładny) i ustaw liczbę grup na 10.



Aby móc zaobserwować uzyskany wynik wyjście procesu klasteryzacji podłącz do operatora redukcji wymiarowości (PCA), skonfigurowanego jak w poprzednim zadaniu a następnie całość podłącz do wyjścia.

Narysuj wykres typu *Scatter* i pokoloruj punkty wg klastrów. W sprawozdaniu umieść wykres wynikowy oraz zrzut ekranu z procesu. Zastanów się i spróbuj odpowiedzieć na pytanie dlaczego punkty które wydaje się że leżą daleko tworzą klastry, a inne, które leżą blisko siebie należą do innych klastrów. Z czego to może wynikać. Które rozwiązanie jest bardziej poprawne – czy uzyskane na wykresie, czy wygenerowane przez algorytm klasteryzacji.

Przyjrzyj się jednemu z klastrów, w tym celu dodaj filtr wektorów – operator *Filter examples* i ustaw go tak aby odfiltrował dane z jednego klastra (najlepiej z klastra leżącego po prawej stronie na dole). W tym celu dla RapidMiner 5 ustaw *condition class* na *attribute_value_filter* i ustaw *parameter string* na wartość *cluster* = ***numer_klastra*** gdzie ***numer_klastra*** oznacza nazwę danego klastra np. *cluster_3*. Następnie podłącz filtr kolumn tak aby pozostawić jedynie te kolumny w których nie występują wartości 0 dla całej kolumny. W tym celu wstaw operator *Select attributes* i ustaw *attribute filter type* na *numerical_value_filter* oraz ustaw *numerical condition* wstawiając wartość „= 0" oraz zaznacz opcję *inverse selection*. W efekcie wszystkie kolumny zawierające dla każdego wiersza wartość 0 zostaną usunięte.

W sprawozdaniu pokaż przykładowe wektory należące do pojedynczego klastra oraz opisz ile wektorów do danego klastra należy i z ilu niezerowych atrybutów się składa.

Zad 4 Analiza wektorów odstających

Często dokonując analizy logów chcemy wyznaczyć przypadki nietypowe czyli takie które w nietypowy sposób korzystają z portalu internetowego, lub jeśli są to logi urządzenia to chcielibyśmy wyznaczyć nietypowe zachowania urządzenia. Powyższa analiza ma szczególne znaczenie w przypadku analizy danych będących analizą logów pochodzących z systemów bezpieczeństwa, gdyż analiza przypadków nietypowych pozwala zwykle na odkrycie nietypowych zachowań użytkowników – czyli zachowań użytkowników które odbiegają od normalnej codziennej pracy.

W celu analizy tak postawionego zadania zastąp operator *Clustering* operatorem *Detect Outlier (Distances)*. Jego zadaniem jest analiza odległości pomiędzy poszczególnymi przypadkami ze zbioru danych i wyznaczenie *n* najbardziej nietypowych przypadków, gdzie wartość *n* oznaczana jest jako parametr modelu *number of outliers*. Domyślnie wartość ta określona jest na 10. W wyniku zastosowania tego operatora na jego wyjściu zostanie uzyskany oryginalny zbiór danych zawierający dodatkową kolumnę *outlier* przyjmującą wartości *true* dla tych wektorów które traktowane są jako wartości nietypowe.

Następnie odfiltruj ze zbioru danych wszystkie wektory nietypowe za pomocą operatora *Filter examples* a następnie w sprawozdaniu pokaż których użytkowników uznano za nietypowych i jakie były numery sesji dla tych użytkowników.