# Zadania - informacje ogólne:

Zadania należy wysyłać poprzez CloudA. Treści zadań są tylko w tym pliku, natomiast w CloudA zdefiniowane są tylko numery zadań, żeby nie powielać niepotrzebnie treści.

Każde zadanie ma określony ostateczny termin jego oddania (podany w CloudA). Termin ten jest zazwyczaj dłuższy, co umożliwia wykonanie i zaliczenie zadania nawet w przypadku nieobecności na zajęciach, kilkudniowej choroby itp. **W związku z tym nie ma możliwości oddawania/nadrabiania zadań po tym terminie za wyjątkiem bardzo wyjątkowych sytuacji (np. dłuższy pobyt w szpitalu). Za nieoddane w terminie zadanie przyznawane jest 0 punktów.**

Jeśli nie jest podane inaczej w treści zadania, w CloudA należy załączyć tekstowy plik *wyniki.txt*, w którym zapisane są wyniki/odpowiedzi **(jeśli odpowiedzi do zadania jest kilka to proszę koniecznie oznaczyć odpowiedzi numerem odpowiedniego podpunktu z zadania, żeby było jasne, do czego jest dana odpowiedź)**. **W przypadku wysyłki zadania przez CloudA bardzo proszę koniecznie wysyłać te i tylko te elementy, które są wyspecyfikowane do oddania w treści zadania ! Nieprawidłowa wysyłka będzie skutkowała odrzuceniem zadania (0 punktów).**

Przy każdym zadaniu poniżej podana jest jego punktacja standardowa (podana też w CloudA) oraz dodatkowe punkty bonusowe. **Punkty bonusowe przysługują tylko w przypadku w pełni poprawnego rozwiązania i oddania zadania na zajęciach, na których zostało one zadane.** Punkty bonusowe są nagrodą za szybkie rozwiązanie zadania i wykraczają poza maksymalną pulę punktów niezbędnych do uzyskania najwyższej oceny zaliczeniowej. W przypadku wysyłki zadania po zajęciach nie ma już możliwości otrzymania tych punktów.

Zadania są przeznaczone do pracy samodzielnej. Plagiaty są bardzo niesprawiedliwe/nieuczciwe wobec osób rozwiązujących zadania w pojedynkę i będą karane (wyzerowanie punktów za zadania dla osoby plagiatującej oraz dającej zadanie).

**Oddanie zadania jeszcze na zajęciach powinno być realizowane poprzez bezpośredni kontakt głosowy i prezentację, chyba że z jakichś powodów ustalimy indywidualnie inny sposób udostępnienia wyników (mail, CloudA itp.).** W szczególności nie należy podczas zajęć wysyłać zadań mailem lub przez CloudA bez wyraźnego ustalenia tego ze mną – takie zadania zostaną potraktowane jak wysłane po zajęciach tzn. nie będzie możliwości otrzymania za nie punktów bonusowych. ***Osoby, które oddały zadania na zajęciach, nie muszą i nie powinny ich już wysyłać przez CloudA. Ponowna wysyłka generuje niepotrzebnie dodatkową pracę po mojej stronie i grozi nieprawidłowym nadpisaniem uzyskanych już punktów wpisanych w dzienniku ocen.***

Reszta szczegółów niezbędnych do wysyłki zadania podawana jest w treści zadań.

**Maksymalna liczba punktów 120 (bez punktów bonusowych). Przeliczenie uzyskanej liczby punktów R na punkty docelowe P z przedziału [0,100]:**

$$P=R*100/120$$

Dla ułatwienia przeliczyłem przedziały w skali oficjalnej [0,100] na przedziały w skali wewnętrznej [0,110] (z zaokrągleniem w dół, żeby było korzystniej dla studentów):

| min | max | ocena |
|---|---|---|
| **0** | **60** | **2** |
| **61** | **72** | **3** |
| **73** | **84** | **3,5** |
| **85** | **93** | **4** |
| **94** | **103** | **4,5** |
| **104** | **112** | **5** |
| **113** | **120** | **6** |

Czyli jak ktoś uzyska np. 116 punktów w skali wewnętrznej to nie musi już tego przeliczać na skalę oficjalną tylko sprawdza powyżej, w jakim przedziale się znajdzie (dla 116 punktów będzie to ostatni przedział czyli ocena 6).

## Zadanie 1

| | |
|---|---|
| **Max. liczba punktów** | 5 |
| **Punkty bonusowe** | 2 |

Na podstawie artykułu Building Book Recommendation System in R zaimplementuj system rekomendacji książek w oparciu o segmentację klientów. Dane wejściowe można pobrać z miejsca wskazanego w artykule: CharlesBookClub.csv lub z katalogu z zadaniami.

*Co należy przesłać:*

- Nr segmentu (1-3), który jest najbardziej obiecujący (zapewnia największy zysk) z punktu widzenia sprzedaży książki *The Art of History of Florence*.
- Wszystkie kombinacje RFM we wskazanym segmencie.
- Lift chart dla zrealizowanego modelu.

*Dodatkowe uwagi/wskazówki:*

- W oryginalnym zbiorze danych wejściowych zamieszczonym w internecie jest kolumna *Related Purchase*, natomiast w kodzie zamieszczonym w artykule nazwa tej kolumny występuje jako *Related.Purchase* . Z niewiadomych powodów (być może zależnych od wersji R i używanych pakietów) nie zawsze powoduje to błąd, ale na wszelki wypadek można na początku zamienić (nawet ręcznie) nazwę tej kolumny na *Related.Purchase* .
- Zamieszczona w artykule funkcja *respondrate* nie działa poprawnie. Zastąp ją następującą funkcją:

```
respondrate<-function(r,f,m,data,x=0,y=0)
{
 x <- sum(data$Rcode==r & data$Fcode==f & data$Mcode==m)
 y <- sum(data$Rcode==r & data$Fcode==f & data$Mcode==m & data$Florence==1)
 if(x > 0)
  y/x
 else
  0
}
```

- Zamieszczony w artykule fragment kodu, zaczynający się od komentarza *#Code to find RFM combination in segment 1*, nie działa poprawnie. Zastąp go następującym kodem:

```
#Code to find RFM combination in segment 1
l<-
cb.train%>%group_by(Rcode,Fcode,Mcode)%>%summarize(rr=respondrate(Rcode,Fcode,Mc
ode,cb.train))
l1<-filter(l, rr > 2*m)
l2<-filter(l, rr > m & rr <= 2*m)

#Print out the RFM combinations in segment 1
for(i in 1:nrow(l1))
{print(l1[i,]$Rcode*100+l1[i,]$Fcode*10+l1[i,]$Mcode)}
```

- Analiza RFM jest jedną z najpopularniejszych metod segmentacji klientów (patrz np.: RFM (market research), How to Identify Your Best Customers Using RFM Segmentation).

## Zadanie 2

| | |
|---|---|
| **Max. liczba punktów** | 5 |
| **Punkty bonusowe** | 2 |

Na podstawie artykułu Building Book Recommendation System in R zaimplementuj system rekomendacji książek w oparciu o regresję logistyczną w dwóch wariantach: 1) model oparty o wszystkie predyktory, 2) model oparty tylko o predyktory R, M, F. Dane wejściowe można pobrać z miejsca wskazanego w artykule: CharlesBookClub.csv lub z katalogu z zadaniami.

*Co należy przesłać:*

- Skumulowany wykres zysków (cumulative gains chart) dla zrealizowanego modelu.
- Lift chart dla zrealizowanego modelu.

*Dodatkowe uwagi/wskazówki:*

- W zamieszczonych w artykule fragmentach kodu dla obu wariantów należy zamienić działającą niepoprawnie linijkę:

  cb_mat<-confusionMatrix(as.factor(cb_pred),cb.test$Florence)

  na:

  cb_mat<-confusionMatrix(as.factor(cb_pred),as.factor(cb.test$Florence))

- Regresja logistyczna (funkcja *glm*) jest używana, gdy zmienna wyjściowa przyjmuje tylko 2 wartości (Florence= 0 lub 1 ).

## Zadanie 3

| | |
|---|---|
| **Max. liczba punktów** | 5 |
| **Punkty bonusowe** | 2 |

Na podstawie artykułu Building Book Recommendation System in R zaimplementuj system rekomendacji książek w oparciu o technikę K-NN (K-Nearest Neighbors). Dane wejściowe można pobrać z miejsca wskazanego w artykule: CharlesBookClub.csv lub z katalogu z zadaniami.

*Co należy przesłać:*

- Skumulowany wykres zysków (cumulative gains chart) dla zrealizowanego modelu.

## Zadanie 4

| | |
|---|---|
| **Max. liczba punktów** | 10 |
| **Punkty bonusowe** | 5 |

Wzorując się na Zadaniu 1 opracuj i zaimplementuj system predykcji zakupów (zmienna Purchase) w oparciu o zbiór danych dotyczących sprzedaży gier i oprogramowania edukacyjnego Tayko Software Cataloger (dostępny również w katalogu z zadaniami jako Tayko.csv) przy użyciu segmentacji.

*Co należy przesłać:*

- Krótki opis wykorzystanego modelu (lista wykorzystanych zmiennych i ich funkcje (wejściowe, wyjściowe, …) itp.).
- Wyniki wraz z krótkim omówieniem i wykresem (lift chart) oraz wskazaniem/charakterystyką najbardziej obiecującego segmentu docelowego.
- Skrypt R z zaimplementowanym modelem.
- Zbiór danych wejściowych (o ile nie przekracza limitu pojedynczej wysyłki).

*Dodatkowe uwagi/wskazówki:*

- Zbiór danych nie zawiera, jak sugeruje opis, pola RFM%. Ponadto wskaźnik Monetary nie jest niestety dostępny w podanym zbiorze (pole Spending nie jest sumą pieniędzy wydanych przez klienta w dłuższym okresie, a jedynie w czasie testowej akcji mailingowej, w której zakupy są celem predykcji, dlatego pole Spending nie może być używane do predykcji w żadnym modelu predykcji, gdyż jego wartość jednoznacznie określa wartość pola Purchase (Spending>0 ⇔ Purchase=1)). Dlatego nie jest możliwe zrobienie pełnej analizy RFM na podanym zbiorze. Jeśli więc chcesz wykorzystać tego typu informacje w swoim modelu to segmentuj na podstawie dwóch pól: last_update_days_ago (Recency przy założeniu, że aktualizacje rekordu klienta są robione głównie ze względu na zakup), Freq (Frequency). Ewentualnie można też oczywiście segmentować po innym sensownym zestawie pól.
- W razie potrzeby (zbyt długie obliczenia lub przekroczenie limitu dostępnej pamięci) zmniejsz / użyj zmniejszonego zbioru wyspecyfikowanych danych wejściowych.

## Zadanie 5

| | |
|---|---|
| **Max. liczba punktów** | 10 |
| **Punkty bonusowe** | 5 |

Wzorując się na Zadaniu 2 opracuj i zaimplementuj system predykcji zakupów (zmienna Purchase) w oparciu o zbiór danych dotyczących sprzedaży gier i oprogramowania edukacyjnego Tayko Software Cataloger (dostępny również w katalogu z zadaniami jako Tayko.csv) przy użyciu regresji logistycznej (tylko w jednym wariancie).

*Co należy przesłać:*

- Krótki opis wykorzystanego modelu (lista wykorzystanych zmiennych i ich funkcje (wejściowe, wyjściowe, …) itp.).
- Skumulowany wykres zysków (cumulative gains chart) dla zrealizowanego modelu.
- Lift chart dla zrealizowanego modelu.
- Skrypt R z zaimplementowanym modelem.
- Zbiór danych wejściowych (o ile nie przekracza limitu pojedynczej wysyłki).

*Dodatkowe uwagi/wskazówki:*

- Patrz Zadanie 4.
-

## Zadanie 6

| | |
|---|---|
| **Max. liczba punktów** | 10 |
| **Punkty bonusowe** | 5 |

Wzorując się na Zadaniu 3 opracuj i zaimplementuj system predykcji zakupów (zmienna Purchase) w oparciu o zbiór danych dotyczących sprzedaży gier i oprogramowania edukacyjnego Tayko Software Cataloger (dostępny również w katalogu z zadaniami jako Tayko.csv) przy użyciu K-NN.

*Co należy przesłać:*

- Krótki opis wykorzystanego modelu (lista wykorzystanych zmiennych i ich funkcje (wejściowe, wyjściowe, …) itp.).
- Skumulowany wykres zysków (cumulative gains chart) dla zrealizowanego modelu.
- Skrypt R z zaimplementowanym modelem.
- Zbiór danych wejściowych (o ile nie przekracza limitu pojedynczej wysyłki).

*Dodatkowe uwagi/wskazówki:*

- Patrz Zadanie 4.

## Zadanie 7

| | |
|---|---|
| **Max. liczba punktów** | 5 |
| **Punkty bonusowe** | 2 |

Na podstawie artykułu How to code a recommendation system in R zaimplementuj system rekomendacji książek w oparciu o filtrowanie oparte na treści (content-based filtering). Dane wejściowe należy pobrać z: BX-CSV-Dump (nie są już dostępne w miejscu wskazanym w artykule) lub z katalogu z zadaniami (BX-CSV-Dump.zip). W przypadku pobrania z podanego adresu należy zamienić wszystkie myślniki (-) na kropki w nazwach kolumn w plikach csv.

*Co należy przesłać:*

- Listę 5 rekomendacji wypisanych przez funkcję *visualizar_recomendacion*.

*Dodatkowe uwagi/wskazówki:*

- Zamieszczona w artykule funkcja *visualizar_recomendacion* nie działa poprawnie (serwer odmawia dostępu do plików graficznych). Zastąp ją następującą funkcją:

```
visualizar_recomendacion = function(recomendation,
                                    recommended_book, image, n_books = 5)
{

 if(n_books > nrow(recomendation)) {n_books = nrow(recomendation)}

 cat("---REKOMENDACJE---", "\n")

 for(i in 1:n_books)
 {
  title <- as.character(recomendation[i,'Book.Title'])
  author <- as.character(recomendation[i,'Book.Author'])
  year <- as.character(recomendation[i,'Year.Of.Publication'])
  publ <- as.character(recomendation[i,'Publisher'])
  category <- as.character(recomendation[i,'category'])
  cat(i, ": ", title, ", ", author, ", ", year, ", ", publ, ", ", category, "\n")
 }
}
```

- Wczytywanie plików zastosowane w artykule:

```
ratings = read.csv(files[1], sep = ";")
…
```

może prowadzić do błędów gdyż jest wrażliwe na kolejność, w jakiej pliki wejściowe umieszczone na liście. Zamiast tego lepiej użyć poleceń:

```
ratings = read.csv("…\\Ratings.csv", sep = ";") # w miejsce 3 kropek należy wpisać ścieżkę do pliku
…
```

- Plik users.csv ze zbioru wejściowego zawiera niestety pewne nieoczywiste błędy, które powodują jego nieprawidłowe wczytanie np. w przy z linii występuje cudzysłów w polu objętym cudzysłowami ("10724";"groningen, \"n/a, netherlands";NULL , "10743";"vienna, \"n/a\", austria";"43"). Powoduje to nieprawidłowe wczytanie kolejnych wierszy pliku. Przed wczytaniem pliku można go więc poprawić pod tym kątem (np. robiąc zamianę \″ n na \′*n*).

## Zadanie 8

| | |
|---|---|
| **Max. liczba punktów** | 5 |
| **Punkty bonusowe** | 2 |

Na podstawie artykułu How to code a recommendation system in R zaimplementuj system rekomendacji książek w oparciu o filtrowanie kolaboratywne oparte na przedmiotach (item-based collaborative filtering). Dane wejściowe należy pobrać z: BX-CSV-Dump (nie są już dostępne w miejscu wskazanym w artykule) lub z katalogu z zadaniami (BX-CSV-Dump.zip). W przypadku pobrania z podanego adresu należy zamienić wszystkie myślniki (-) na kropki w nazwach kolumn w plikach csv.

*Co należy przesłać:*

- Listę 5 rekomendacji wypisanych przez funkcję *visualizar_recomendacion*.

*Dodatkowe uwagi/wskazówki:*

- Patrz Zadanie 7.

## Zadanie 9

| | |
|---|---|
| **Max. liczba punktów** | 5 |
| **Punkty bonusowe** | 2 |

Na podstawie artykułu How to code a recommendation system in R zaimplementuj system rekomendacji książek w oparciu o filtrowanie kolaboratywne oparte na użytkownikach (user-based collaborative filtering). Dane wejściowe należy pobrać z: BX-CSV-Dump (nie są już dostępne w miejscu wskazanym w artykule) lub z katalogu z zadaniami (BX-CSV-Dump.zip). W przypadku pobrania z podanego adresu należy zamienić wszystkie myślniki (-) na kropki w nazwach kolumn w plikach csv.

*Co należy przesłać:*

- Listę 5 rekomendacji wypisanych przez funkcję *visualizar_recomendacion*.

*Dodatkowe uwagi/wskazówki:*

- Patrz Zadanie 7.

## Zadanie 10*

| | |
|---|---|
| **Max. liczba punktów** | 10 |
| **Punkty bonusowe** | 10 |

Wzorując się na Zadaniu 7 opracuj i zaimplementuj system rekomendacji filmów w oparciu o filtrowanie oparte na treści (content-based filtering). Dane wejściowe należy pobrać z Movie Tweetings lub z katalogu z zadaniami (MovieTweetings.zip).

*Co należy przesłać:*

- Skrypt R z zaimplementowanym modelem.
- Zbiór danych wejściowych (o ile nie przekracza limitu pojedynczej wysyłki).
- Listę 5 rekomendacji dla dowolnego użytkownika wypisanych przez funkcję *visualizar_recomendacion*.

*Dodatkowe uwagi/wskazówki:*

- Zwróć uwagę, że zbiór *Movie Tweetings* nie zawiera klasycznych plików CSV, co może być przyczyną problemów z wczytaniem za pomocą *read.csv* itp. Zamianę na pliki CSV najlepiej przeprowadzić następująco: - zamienić rozszerzenia plików dat -> csv (opcjonalne), - w dowolnym edytorze np. Notatniku dodać nazwy kolumn w pierwszej linijce plików (plik movies: *movieId::title::genres*, plik ratings: *userId::movieId::rating::timestamp*), - w dowolnym edytorze zmienić separator :: na separator 1-bajtowy np. ; (uwaga !: przed zamianą należy sprawdzić/wyszukać, czy plik nie zawiera nowego separatora w środku pól np. tytuły filmów zawierają czasem przecinek, więc nie można go użyć jako separatora).
- Podobieństwo filmów można oceniać tylko na podstawie kategorii. W przeciwieństwie do danych z Zadania 7 rozważany tutaj zbiór filmów ma pole kategorii. Niestety to pole może zawierać ciąg różnych kategorii w różnej kolejności, które pasują do danego filmu. Jest to problematyczne dla funkcji oceniających podobieństwo filmów. W związku z tym przed wykonaniem tego zadania zmodyfikuj zbiór wejściowy tak, by plik *items.dat* zawierał dodatkowe kolumny odpowiadające wszystkim możliwym kategoriom filmowym, a wartościami tych kolumn były 0 (film nie należy do danej kategorii) lub 1 (film należy do danej kategorii).
- W razie potrzeby (zbyt długie obliczenia lub przekroczenie limitu dostępnej pamięci) zmniejsz / użyj zmniejszonego zbioru wyspecyfikowanych danych wejściowych (w tym przypadku np. snapshots\10K).

## Zadanie 11

| | |
|---|---|
| **Max. liczba punktów** | 10 |
| **Punkty bonusowe** | 5 |

Wzorując się na Zadaniu 8 opracuj i zaimplementuj system rekomendacji filmów w oparciu o filtrowanie kolaboratywne oparte na przedmiotach (item-based collaborative filtering). Dane wejściowe należy pobrać z Movie Tweetings lub z katalogu z zadaniami (MovieTweetings.zip).

*Co należy przesłać:*

- Skrypt R z zaimplementowanym modelem.
- Zbiór danych wejściowych (o ile nie przekracza limitu pojedynczej wysyłki).
- Listę 5 rekomendacji dla dowolnego użytkownika wypisanych przez funkcję *visualizar_recomendacion*.

*Dodatkowe uwagi/wskazówki:*

- Zamiana plików *Movie Tweetings* na CSV -> patrz Zadanie 10.
- Kolumna *timestamp* w *ratings* powoduje błąd przy wykonywaniu kodu zamieszczonego w artykule (być może z powodu istnienia duplikatów w tej kolumnie). Aby uniknąć problemów z tym związanych, usuń tą kolumnę zaraz na początku kodu odpowiadającego temu zadaniu:

  **ratings <- ratings[,-4]**
  user_item = ratings %>% top_n(10000) %>% pivot_wider(names_from = movie_id,values_from = rating) %>% as.data.frame()

- W razie potrzeby (zbyt długie obliczenia lub przekroczenie limitu dostępnej pamięci) zmniejsz / użyj zmniejszonego zbioru wyspecyfikowanych danych wejściowych.

## Zadanie 12

| | |
|---|---|
| **Max. liczba punktów** | 10 |
| **Punkty bonusowe** | 5 |

Wzorując się na Zadaniu 9 opracuj i zaimplementuj system rekomendacji filmów w oparciu o filtrowanie kolaboratywne oparte na użytkownikach (user-based collaborative filtering). Dane wejściowe należy pobrać z Movie Tweetings lub z katalogu z zadaniami (MovieTweetings.zip).

*Co należy przesłać:*

- Skrypt R z zaimplementowanym modelem.
- Zbiór danych wejściowych (o ile nie przekracza limitu pojedynczej wysyłki).
- Listę 5 rekomendacji dla dowolnego użytkownika wypisanych przez funkcję *visualizar_recomendacion*.

*Dodatkowe uwagi/wskazówki:*

- Zamiana plików *Movie Tweetings* na CSV -> patrz Zadanie 10.
- W razie potrzeby (zbyt długie obliczenia lub przekroczenie limitu dostępnej pamięci) zmniejsz / użyj zmniejszonego zbioru wyspecyfikowanych danych wejściowych.

## Zadanie 13

| | |
|---|---|
| **Max. liczba punktów** | 10 |
| **Punkty bonusowe** | 5 |

Na podstawie artykułu How to build a recommendation engine in R zaimplementuj system rekomendacji filmów przy wykorzystaniu dedykowanego pakietu *recommenderlab*. Dane wejściowe (inne niż użyte w artykule) należy pobrać z Movie Tweetings lub z katalogu z zadaniami (MovieTweetings.zip).

*Co należy przesłać:*

- Skrypt R z zaimplementowanym modelem.
- Zbiór danych wejściowych (o ile nie przekracza limitu pojedynczej wysyłki).
- Listę 5 rekomendacji dla dowolnego użytkownika ze zbioru danych.

*Dodatkowe uwagi/wskazówki:*

- Format plików ze zbioru *Movie Tweetings* różni się od użytych oryginalnie w artykule plików ze zbioru MovieLens, natomiast zawartość jest prawie identyczna. Aby uniknąć większych modyfikacji kodu, wystarczy zamienić pliki *Movie Tweetings* na pliki CSV wg uwagi z Zadania 10.
- W zamieszczonym w artykule fragmencie kodu należy zamienić działającą niepoprawnie linijkę:

  Top_5_df$movieId=as.numeric(levels(Top_5_df$movieId))

  na:

  Top_5_df$movieId=as.numeric(Top_5_df$movieId)

- W razie potrzeby (zbyt długie obliczenia lub przekroczenie limitu dostępnej pamięci) zmniejsz / użyj zmniejszonego zbioru wyspecyfikowanych danych wejściowych (w tym przypadku np. snapshots\10K).

## Zadanie 14

| | |
|---|---|
| **Max. liczba punktów** | 10 |
| **Punkty bonusowe** | 5 |

Na podstawie artykułu Evaluation of Recommender Systems zaimplementuj kod porównujący różne modele rekomendacyjne. Dane wejściowe (inne niż użyte w artykule) należy pobrać z Movie Tweetings lub z katalogu z zadaniami (MovieTweetings.zip).

*Co należy przesłać:*

- Skrypt R z zaimplementowanym kodem porównującym.
- Zbiór danych wejściowych (o ile nie przekracza limitu pojedynczej wysyłki).
- Diagramy *ROC curve* i *Precision-recall* dla pojedynczego modelu.
- Diagramy *ROC curve* i *Precision-recall* dla 6 użytych w artykule modeli (IBCF_cos, IBCF_cor, UBCF_cos, UBCF_cor, SVD, random).

*Dodatkowe uwagi/wskazówki:*

- Format plików ze zbioru *Movie Tweetings* różni się od użytych oryginalnie w artykule plików ze zbioru MovieLens, natomiast zawartość jest prawie identyczna. Aby uniknąć większych modyfikacji kodu, wystarczy zamienić pliki *Movie Tweetings* na pliki CSV wg uwagi z Zadania 10.
- W razie potrzeby (zbyt długie obliczenia lub przekroczenie limitu dostępnej pamięci) zmniejsz / użyj zmniejszonego zbioru wyspecyfikowanych danych wejściowych (w tym przypadku np. snapshots\10K).

## Zadanie 15

| | |
|---|---|
| **Max. liczba punktów** | 10 |
| **Punkty bonusowe** | 10 |

Znajdź samodzielnie odpowiedni zbiór danych (najlepiej zgodnie z zainteresowaniami oraz w zgodzie z licencją na wykorzystanie zbioru) zawierający przynajmniej 1000 rekordów oraz inny niż wykorzystywane na tych zajęciach. Zaimplementuj system rekomendacyjny wykorzystujący ten zbiór, oparty na dwóch różnych algorytmach wybranych z następującego zestawu:

- Content-based filtering
- User-based collaborative filtering
- Item-based collaborative filtering
- SVD (Singular Value Decomposition)
- Sieci neuronowe
- Naiwny klasyfikator bayesowski
- Reguły asocjacyjne
- Model hybrydowy (złożony z dowolnych powyższych algorytmów/modeli).

Poprawnie zaimplementowany model powinien wypisywać dla dowolnie wybranego klienta ze zbioru danych wejściowych 5 rekomendacji wyliczonych przez pierwszy użyty algorytm/model i 5 rekomendacji wyliczonych przez drugi użyty algorytm/model (można to zrealizować w 1 skrypcie R-owym zawierającym oba modele, albo w dwóch oddzielnych skryptach).

*Co należy przesłać:*

a) Krótka charakterystyka wybranego zbioru (link, krótki opis itp.).
b) Krótki tekstowy opis zastosowanych modeli/algorytmów.
c) Skrypt (lub 2 skrypty) R z zaimplementowanym systemem rekomendacyjnym.

*Dodatkowe uwagi/wskazówki:*

- Lista przykładowych linków zawierających ciekawe zbiory danych:
  - https://github.com/caserec/Datasets-for-Recommender-Systems
  - https://github.com/RUCAIBox/RecSysDatasets
  - https://grouplens.org/datasets/
  - https://webscope.sandbox.yahoo.com/catalog.php?datatype=r
  - https://gist.github.com/entaroadun/1653794
  - https://www.kaggle.com/datasets/netflix-inc/netflix-prize-data
  - https://paperswithcode.com/datasets?task=recommendation-systems
  - https://cseweb.ucsd.edu/~jmcauley/datasets.html
  - https://analyticsindiamag.com/10-open-source-datasets-one-must-know-to-build-recommender-systems/
  - https://www.kdnuggets.com/2016/02/nine-datasets-investigating-recommender-systems.html
  - https://www.scipublications.com/journal/index.php/ijmebac/article/view/350/tab2
  - https://www.kaggle.com/