Uniwersytet Warszawski
Wydział Matematyki, Informatyki i Mechaniki

Xxx Yyy
Nr albumu: 000000

Sugerowanie wyboru ścieżki kształcenia zintegrowane z USOS

Praca licencjacka
na kierunku INFORMATYKA

Praca wykonana pod kierunkiem
dra Roberta Dąbrowskiego
Pion Zastępcy Kanclerza ds. Informatycznych

??? 2015

Oświadczenie kierującego pracą

Potwierdzam, że niniejsza praca została przygotowana pod moim kierunkiem i kwalifikuje się do przedstawienia jej w postępowaniu o nadanie tytułu zawodowego.

Data Podpis kierującego pracą

Oświadczenie autora (autorów) pracy

Świadom odpowiedzialności prawnej oświadczam, że niniejsza praca dyplomowa została napisana przeze mnie samodzielnie i nie zawiera treści uzyskanych w sposób niezgodny z obowiązującymi przepisami.

Oświadczam również, że przedstawiona praca nie była wcześniej przedmiotem procedur związanych z uzyskaniem tytułu zawodowego w wyższej uczelni.

Oświadczam ponadto, że niniejsza wersja pracy jest identyczna z załączoną wersją elektroniczną.

Data Podpis autora (autorów) pracy

Streszczenie

Praca poświęcona jest systemowi Hermes, który ma na celu rekomendację przedmiotów dla studentów. Znajduje się w niej opis funkcjonalności, architektury, implementacji, zastosowanych algorytmów oraz organizacji pracy nad projektem.

Słowa kluczowe

słowa kluczowe

Dziedzina pracy (kody wg programu Socrates-Erasmus)

11.0 Matematyka, Informatyka:
11.3 Informatyka

Klasyfikacja tematyczna

Information systems
Information systems applications
Decision support systems
Data analytics

Tytuł pracy w języku angielskim

Recomendation of educational path integrated with USOS

Spis treści

Rozdział 1

Wstęp

1.1. Opis Projektu

Głównym celem projektu Hermes było stworzenie serwisu internetowego wspierającego studentów w procesie zoptymalizowania wyboru przedmiotów. Serwis ma za zadanie umożliwić studentom lepsze planowanie ścieżki studiów i kariery zawodowej poprzez proponowanie przedmiotów, które mogą pasować do ich upodobań i predyspozycji.

Dodatkowym spełnionym wymaganiem projektu jest oferowanie usług przewidywania dla konkretnych studentów ocen z przedmiotów, których jescze nie ukończyli lub bądź nie podjęli.

Serwis oferuje również wsparcie dla uniwersytetu w postaci przewidywania ilości studentów którzy zapiszą się na konkretny przedmiot.

W celu obliczania predykcji system wykorzystywać będzie uczenie maszynowe na statystykach wszystkich studentów Uniwersytetu, a także ewentualnie wybrane przedmioty i uzyskane oceny przez studenta proszącego o propozycję.

Projekt został zrealizowany na zlecenie działu sieci komputerowych Uniwersytetu Warszawskiego.

1.2. od Autorów

Wybraliśmy ten projekt, ponieważ sami jesteśmy studentami i jesteśmy świadomi trudności związanych z wyborem przedmiotów. Oferta programowa Uniwersytetu Warszawskiego jest ogromna. W chwili pisania tej pracy po wyszukaniu w systemie usos zwróconych jest ponad 20000 przedmiotów! Dodatkowo opisy często są niejasne, szczątkowe, bądź brakuje syllabusu. Czasem tematyka przedmiotu obieralnego jest na tyle odmienna od dotychczasowego materiału poznanego przez studenta, że może on nie być świadomym, czy dany materiał odpowiada jego predyspozycjom i preferencjom.

Pragniemy zaadresować te problemy za pomocą systemu Hermes. Pisząc go, przyświecała nam idea stworzenia drogowskazu dla studentów, którzy nie są pewni, w jakim kierunku powinni się specjalizować. Wierzymy, iż system dzięki udanym sugestiom zwiększy liczbę studentów, którzy będą naprawdę zainteresowani tematyką przedmiotu, a zmniejszy liczbę studentów zapisanych na przedmioty zupełnie niedopasowane do ich predyspozycji, co często kończy się słabą oceną bądź nawet brakiem zaliczenia. Dzięki temu skorzystają na tym studenci, którzy z większym prawdopodobieństwem zapiszą się na pasujące do ich osiągnięć i predyspozycji przedmioty, z których zdobędą możliwie wiele interesującej ich wiedzy. Zmniejszy się również dla nich ryzyko zapisania się na przedmiot, który w ogóle nie pasuje do ich zainteresowań. Zyska również Uniwersytet, którego studenci będą bardziej zadowoleni z oferty programowej, wzrośnie jakość wiedzy absolwentów oraz lepiej zostanie wykorzystany potencjał studentów. Pomoże to uczelnii być lepiej postrzeganą, bardziej pożądaną przez przyszłych studentów a także zwiększć jej prestiż.

Tomasz Grabowski
Adam Markiewicz
Albert Rozmus
Krzysztof Rutkowski
Wiktor Zuba

Rozdział 2

Funkcjonalności

Przy wejciu na stronę systemu użytkownik otrzymuje stronę startową która jest jednocześnie interfejsem.

Ekran startowy strony

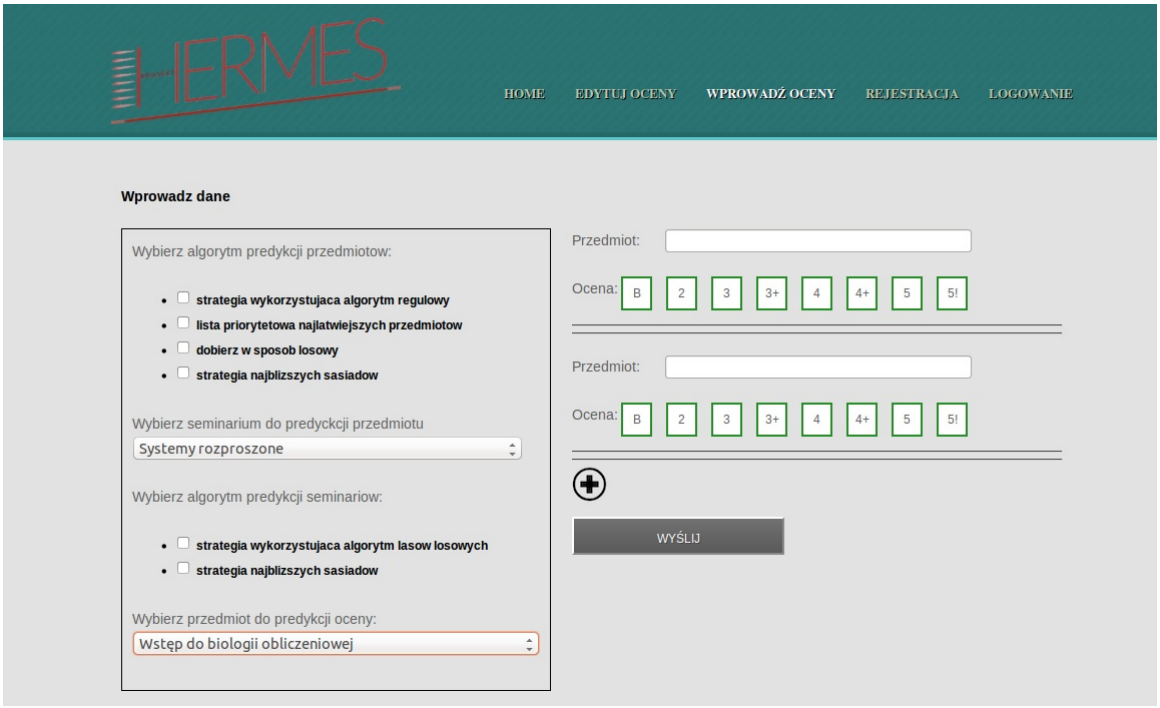
Poniższe podrozdziały opisują funkcjonalności zależne od wybranej opcji.

2.1. Funkcjonalności dla Studentów

W tym podrozdziale znajduje się opis funkcjonalności dostępnych dla użytkownika anonimowego - w domyśle studenta.

W przypadku wyboru zakładki "wprowadź oceny" student uzyskuje możliwość wprowa-

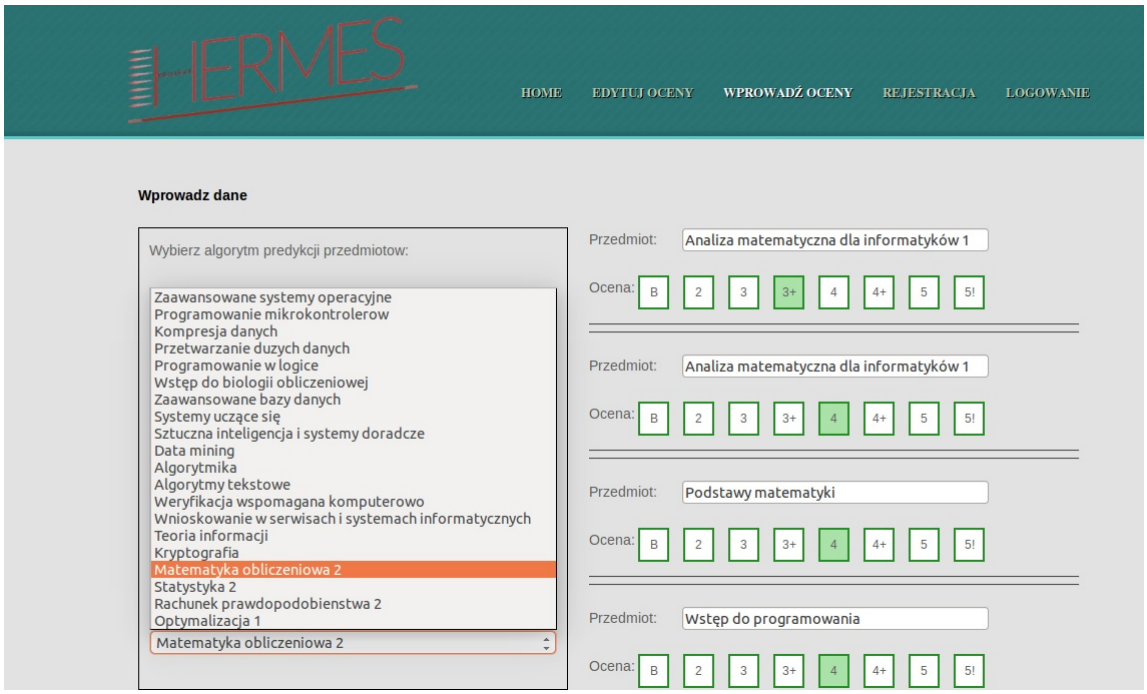
dzenia danych z przedmiotu w celu uzyskania nastepujących predykcji: predykcja oceny z przedmiotu, rekomendacja seminarium oraz rekomendacja przedmiotu. Intefejs ten umozliwia dodawanie dowolnej liczby przedmiotów za pomocą przycisku "plus". Dodatkowo przy wpisywaniu nazwy przedmiotu pojawia się podpowiedź sugerująca istniejące nazwy przedmiotów z bazy pasujące do wpisanego wzorca.



interfejs po wyborze opcji "wprowadź oceny"

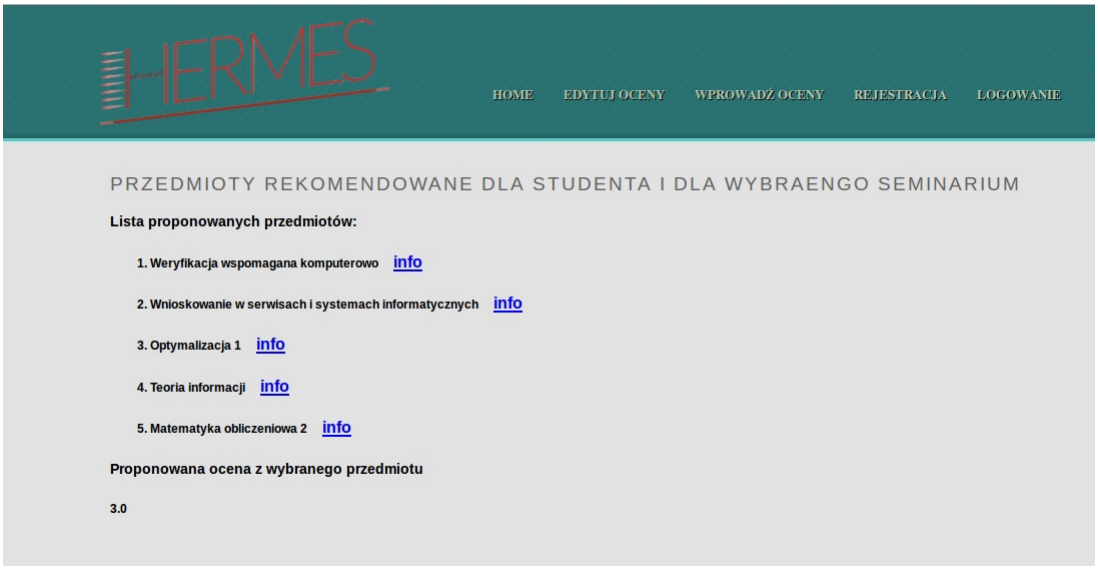
2.1.1. Predykcja ocen

Użytkownik wybiera predykcję oceny z przedmiotu za pomocą opcji "wybierz przedmiot do predykcji oceny"



wybór przedmiotu do predykcji

Po jego wybraniu i uzycia opcji wyślil użytkownik otrzymuje wynik w postaci następującej strony:

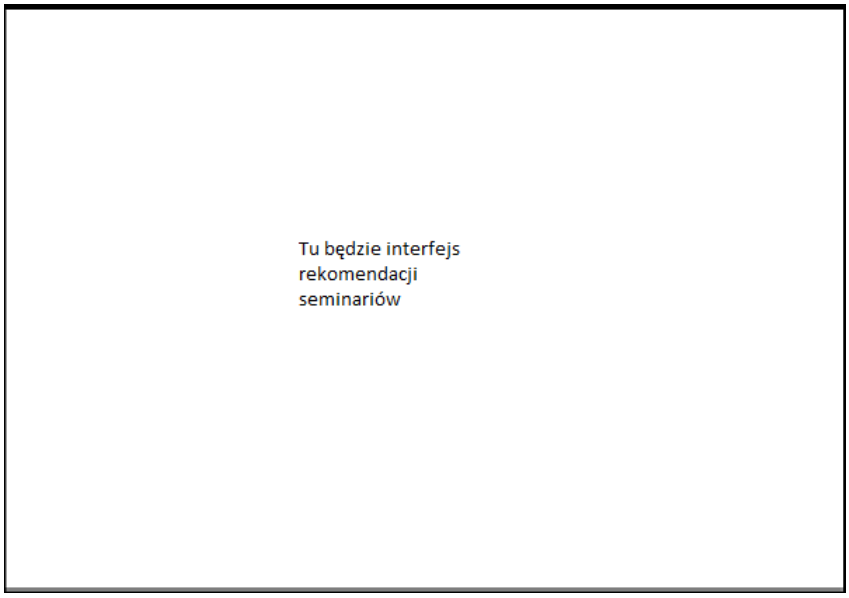


Przykładowy rezultat zapytania o predykcję oceny

2.1.2. Remomendacja seminariów

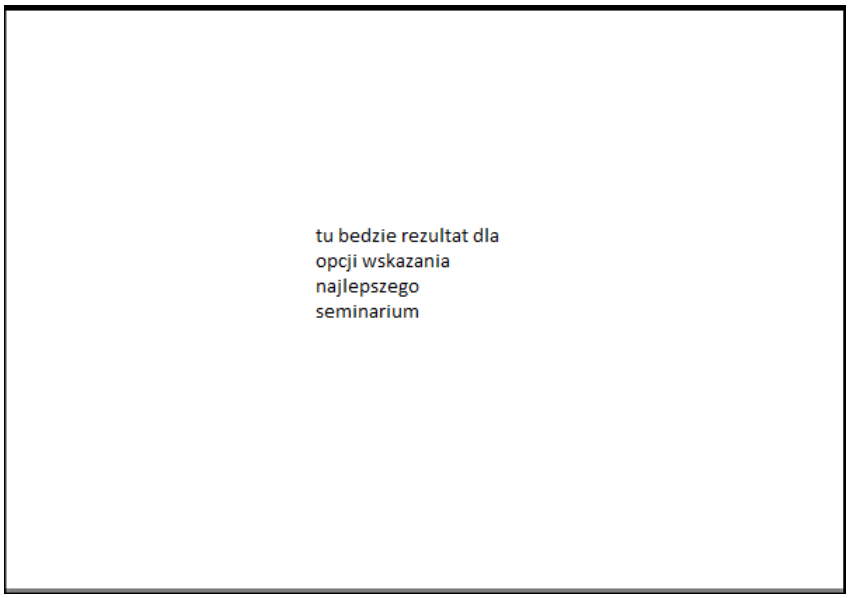
TODO nie działa na moim kompie predykcja seminariów.
Po wybraniu opcji rekomendacji seminariów student staje przed wyborem dwóch opcji :

wybierz najlepsze bądź pokaż ranking.

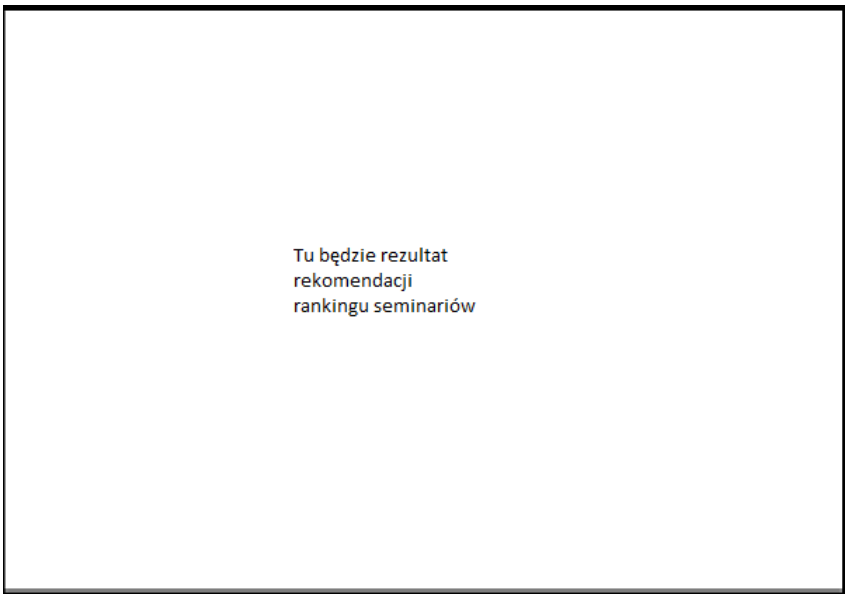


Interfejs rekomendacji seminariów

W obu wypadkach system korzysta z pobranego za pomocą USOSApi programu studiów studenta. Jeżeli student ma więcej niż 1 aktywny kierunek studiow, system prosi go o wybór programu, dla którego predykcja seminariów go interesuje. W zależności od wybranej opcji student otrzymuje jedno seminarium które wg systemu najbardziej pasuje do studenta badź też ranking seminariów od najbardziej pasującego do najmniej wg predykcji systemu.



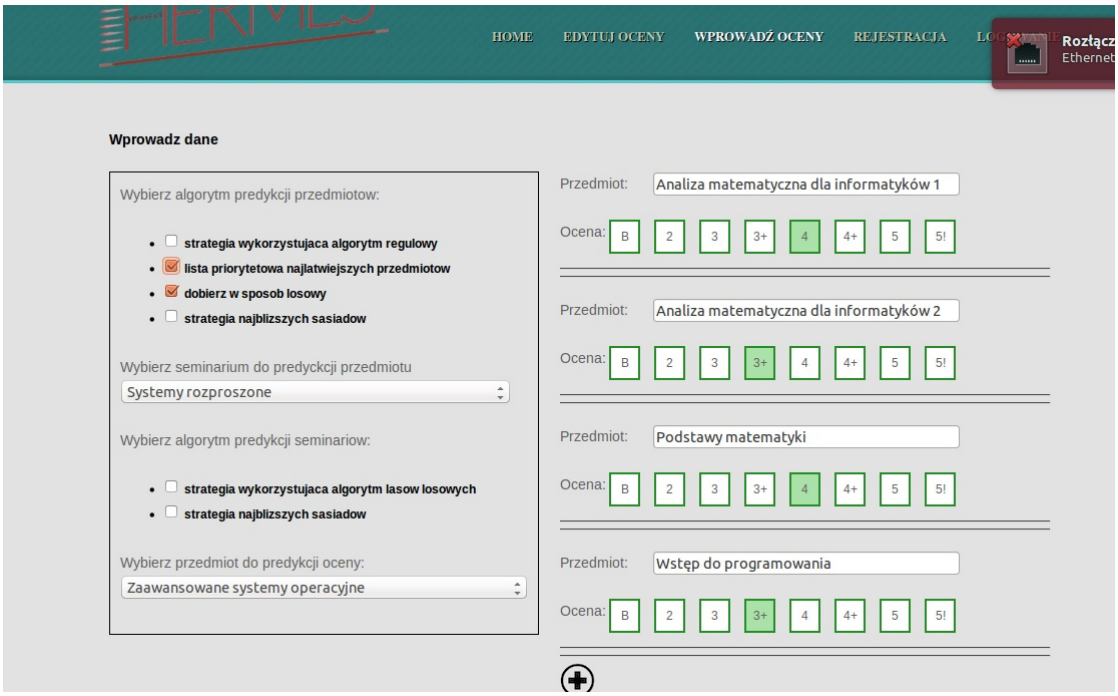
Przykładowy rezultat prośby zarkeomendowanie najlepszego seminarium



Przykładowy rezultat prośby o obliczenie rankingu seminariów

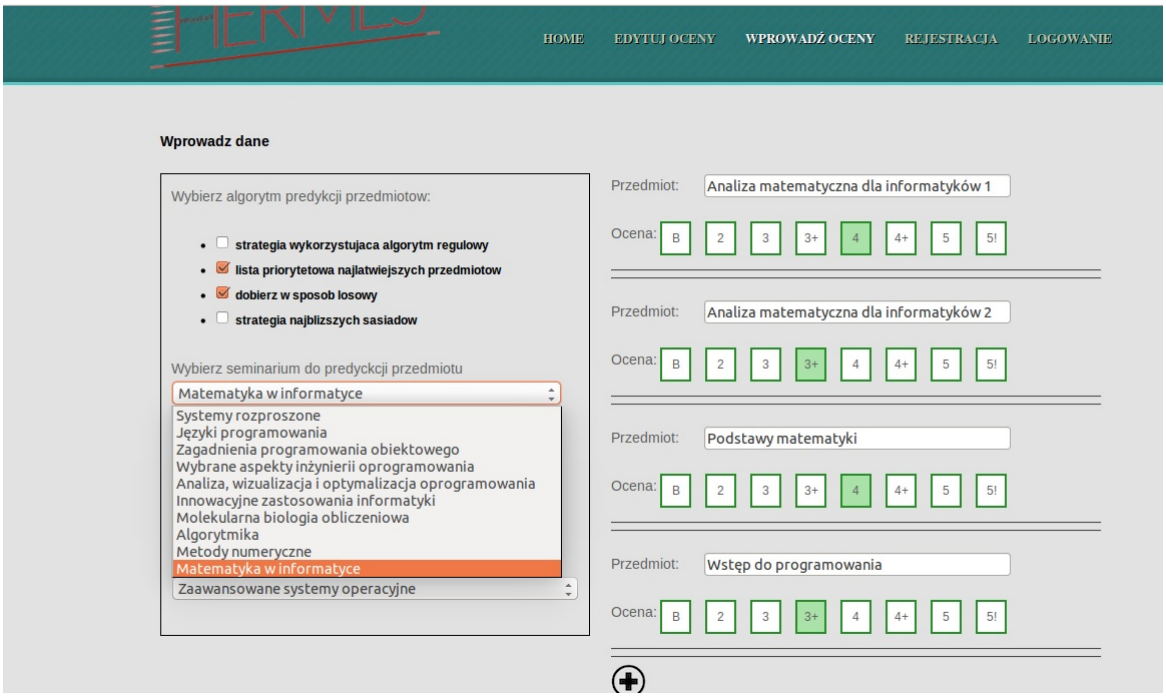
2.1.3. Rekomendacja przedmiotów

Po wybraniu opcji rekomendacji przedmiotów student otrzymuje wybór kilku opcji. Może wybrać opcje "znajdź najłatwiejsze", "szczęśliwy traf", "ranking wg algorytmu najbliższych sąsiadów" oraz "znajdź wg algorytmu regułowego". Możliwy jest także wybór rankingu przedmiotów według wybranego przez siebie seminarium (tzn przedmioty które są najbardziej z nim związane).



Interfejs użytkownika dla opcji rekomendacji przedmiotów - wybór algorytmów

Dodatkowo możliwy wybór seminarium do predykcji.

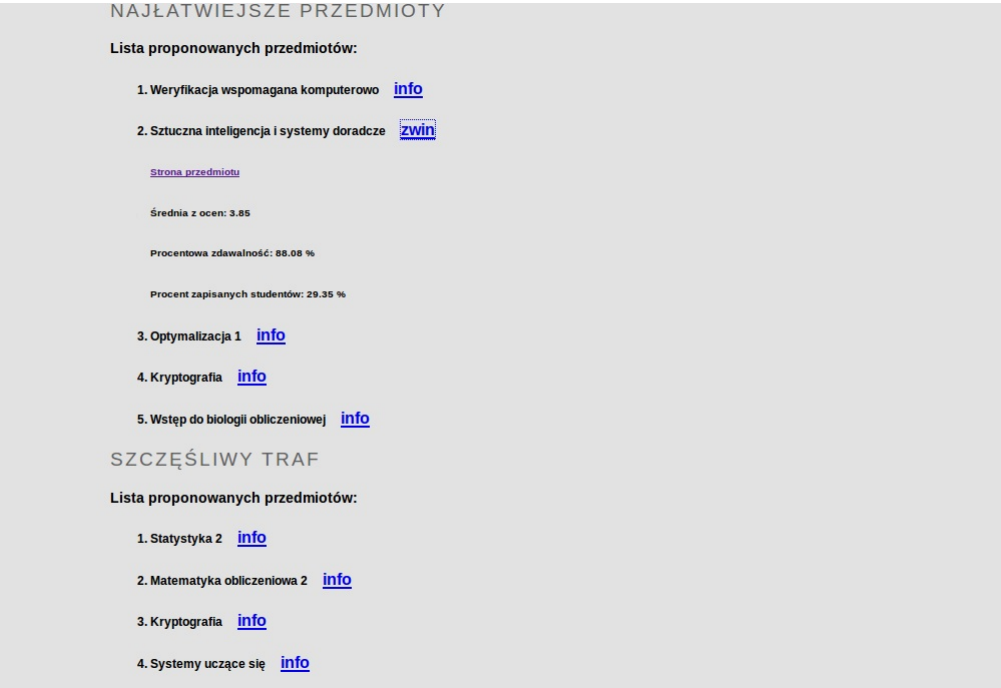


Wybór seminarium na rekomendację przedmiotów związanych z seminarium

Jako wynik każdej z tych rekomendacji student otrzymuje ranking 5 przedmiotów które wg danego algorytmu mają największą szansę pasować do zainteresowań studenta.

Przykładowy rezultat dla rekomendacji przedmiotów wg zainteresowań studenta

Każdy przedmiot po rozwinięciu opisu zawiera proste statystyki : ogólną zdawalność, średnią z ocen, ogólny procent studentów uczęszczających na ten przedmiot czy link do strony przedmiotu w systemie USOS.



Opis przedmiotu po rozwinięciu (tu Sztuczna Inteligencja)

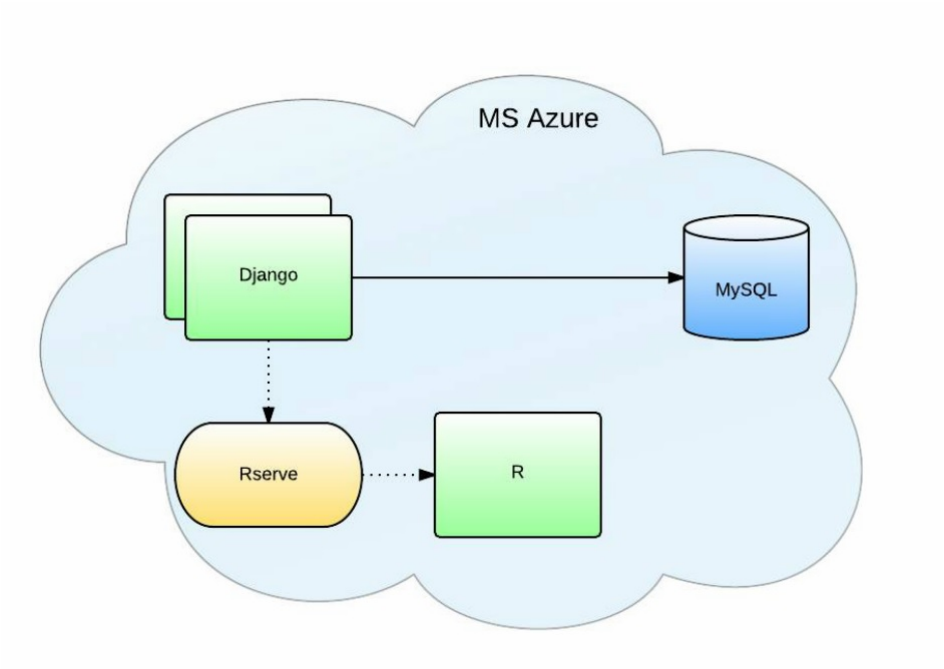
2.2. Funkcjonalności dla Administracji

Domyślnie miała być jeszcze funkcjonalność dla administracji, tzn predykcja popularności przedmiotów, niestety zabrakło nam czasu by ją zaimplementować.

Rozdział 3

Architektura

3.1. Schemat Architektury



Architektura systemu Hermes

W architekturze i logice naszego systemu wyróżniamy następujące komponenty:

- **Chmura (MS Azure)** - Sercem systemu jest główny serwer wraz z innymi usługami znajdujący się w chmurze internetowej. Znajduje się na niej serwer bazy danych, serwer WWW oraz serwer usług analitycznych.
- **RDB (MySQL)** - Relacyjna baza danych zawierająca statystyczne dane dotyczące zdawalności przedmiotów przez studentów, ich zapełnienia, popularność itd. Stanowi bazę do tworzenia predykcji.
- **Sysytem R (R)** - język i zestaw bibliotek uczenia maszynowego wykorzystywany w predykcjach.

- **Rserve (Rserve)** - serwer pośredniczący między stroną WWW a systemem R. Strona komunikuje się z R poprzez komunikaty wysyłane do Rserve.
- **Strona WWW (Django)** - interfejs za pomocą którego użytkownik może przesyłać prośby o wykonanie udostępnianych przez system rekomendacji.
- **Serwer WWW (Django)** - udostępnia użytkownikom stronę internetową, w naszym systemie pośredniczy między interfejsem użytkownika a bazą danych. Pośredniczy również w komunikacji z serwerem analitycznym.

3.2. Schemat współdziałania komponentów architektury

Schemat działania i komunikacji między poszczególnymi komponentami wygląda następująco:

1. Na samym początku system R dokonuje obliczeń i wstępnego przetworzenia zebranych danych budując modele i funkcje, według których będa obliczane predykcje.
2. Poprzez Rserve modele są zapamiętane w systemie.
3. Po wykonaniu przetworzenia danych i obliczenia statystyk aktywuje się serwer WWW i system staje się dostępny dla użytkowników.
4. Użytkownik wchodzi na stronę i otrzymuje formularz logowania. Możliwy jest również tryb pracy dla użytkownika anonimowego.
5. Serwer WWW po odebraniu prośby o rekomendacje przesyła żądanie do Rserve z poleceniem obliczenia predykcji, wykorzystując się przy tym odebrane wcześniej dane użytkownika.
6. Serwer WWW odbiera rezultat zapytania i wyświetla go użytkownikowi za pośrednictwem strony WWW.

Rozdział 4

Technologia

Technologie wykorzystywane w naszym systemie:

Microsoft Azure - Azure jest komercyjną platformą obsługiwaną przez Microsoft. Udostępnia ona usługi związane z chmurą internetową (tzw cloud-computing). W naszym systemie znajduje sie na niej serwer WWW a także serwer bazy danych.

MySQL - Jeden z najpopularniejszych serwerów relacyjncyh baz danych. Znajduje się w nim relacyjna baza danych zawierająca dane niezbędne do stworzenia modeli uczenia maszynowego.

R - język i zestaw bibliotek używany w celu analizy danych a także obliczania predykcji dla studentów bądź uniwersytetu.

- **Python oraz Django** - technologie, za pomocą których tworzymy webowy interfejs użytkownika.

- **USOS Api** - API udostępniane przez system USOS. Nasz system wykorzystuje go w celu zebrania danych zalogowanego użytkownika niezbędnych do rekomendacji.

Rozdział 5

Zastosowane Algorytmy

Przy tworzeniu algorytmów wykorzystaliśmy pakiet statystyczny R. Gwarantował on dużą kontrolę nad algorytmami, których używaliśmy. W naszej pracy pojawlły się wszystkie najważniejsze rodzaje algorytmów używanych w eksploracji danych:

- klasyfikacja
- poszukiwanie reguł asocjacyjnych
- grupowanie

Najwięcej pojawiło się zastosowań pierwszej z metod. Czasem używaliśmy też podejścia statystycznego, nie ucząc jakiegoś modelu, lecz podejmując decyzje na podstawie statystyk. W zastosowanych algorytmach często używalismy odpowiednio zdefiniowanej funkcji odległości między obiektami w naszym modelu. Obiektem był tu wektor n-elementowy ocen studenta pomnożonych o 2 oraz 11 w przypadku oceny 5! a także 0, gdy student nie uczestniczył w przedmiocie, dla dwóch takich wektorów definiujemy miarę odległości między studentami $v = (v_1, .., v_n)$ i $w = (w_1, .., w_n)$:

$$dist(v, w) = \sum_{k=1}^{n} \begin{cases} 10 & \text{gdy } (v_k = 0) \oplus (w_k = 0) \\ |w_k - v_k| & \text{wpp} \end{cases}$$

kod w R:

```
dist <- function(v,w) {
  sum <- 0
  for (i in 1:length(v)) {
    if (xor((v[i] == 0),w[i] == 0)) {
      sum <- sum + 10
    }
    else {
      sum <- sum + abs(v[i] - w[i])
    }
  }
  return(sum)
}
```

dist.R

Podana funkcja mocniej wyraża fakt, że studenci brali różne przedmioty niż to że mieli różne oceny. Uzywaliśmy jej w algorytmach opartych na metodzie k najbliższych sąsiadow, której

używalismy w wielu predykcjach. W naszym systemie mieliśmy kilka rodzajów predykcji i rekomendacji:

- proponowanie przedmiotów
- predykcja seminarium
- predykcja ocen

5.0.1. Algorytmy użyte przy proponowaniu przedmiotów

Przy proponowaniu przedmiotów obieralnych użylismy 4 strategie:

- wyliczanie reguł asocjacyjnych
- strategia najbliższych sąsiadów
- przydział w oparciu o statystyki
- przydział losowy

Reguły asocjacyjne wyliczyliśmy za pomocą algorytmu apriori dostępnego w pakiecie 'arules'. Algorytm brał na wejściu koszyki przedmiotów, czyli zestawy przedmiotów obieralnych, które wzięli poszczególni studenci, zaś na wyjściu zwracał reguły postaci 'jeśli student wziął przedmioty $p_1, .., p_k$ to prawdopodobnie weźmie też przedmiot p_l'. To 'prawdopodobnie' zależy od miar *support* i $confidence$ podawanych na wejściu algorytmu. Miary *support* i $confidence$ ustawiliśmy tak, by reguł nie bylo za duzo, ale też byly sensowne.
Strategia najbliższych sąsiadów została użyta w oparciu o miarę *dist* przedstawioną wyżej. Polegała ona na tym, że spośród pewnej liczby najbliższych studentowi, któremu chcemy coś zaproponować, sasiadów, bierzemy przedmioty, które oni brali. W tym podejściu próbujemy zaklasyfikować studenta do podobnej grupy i przydzielić mu to co brali 'podobni' studenci. Kod w R:

```
recomNearestSub <- function(k, student) {
  n <- dim(data)[1]
  A <- matrix(0,n,2)
  for (i in 1:n) {
    A[i,1] <- dist(student,data[i,])
    A[i,2] <- i
  }
  bestNb <- A[order(A[,1])[1:k],2]
  studNotChosen <- which(student == 0)
  recom <- c()
  for (i in 1:length(bestNb)) {
    nbSub <- which(data[bestNb[i],31:50] > 0)
    recom <- unique(c(recom, nbSub[nbSub %in% studNotChosen]))
  }
  return(recom)
}
```

nearestSub.R

Przydział w oparciu o statystyki to propozycja przedmiotów, które prawdopodobnie byłoby studentowi najłatwiej zaliczyć. Statystykę 'łatwości' przedmiotu stanowiła średnia ważona supportu, średniej ocen i zdawalności przedmiotu.
Przydział losowy polegał na braniu z jenostajnym rozkładem prawdopodobieństwa przedmiotów, których student jeszcze nie wziął.

5.0.2. Algorytmy używane przy predykcji seminariów

Przy proponowaniu seminariów użylismy 2 strategii:

- klasyfikator najbliższych sąsiadów
- klasyfikator lasów losowych

Klasyfikator najbliższych sąsiadow, tak jak w przypadku proponowania przedmiotów, również korzystał ze zdefniowanej funkcji odleglości i ze zbioru decyzji dla najbliższych sąsiadów, czyli seminarium, które wybrali najbliżsi sąsiedzi, jako proponowane wybierał mode, czyli najczęściej występujące wśród sąsiadów seminarium.
Klasyfikator lasów losowych budowaliśmy w oparciu o pakiet 'randomForest'. Jest to klasyfikator, który buduje sie w oparciu o klasyfikacje na pewnej liczbie drzew decyzyjnych, następnie agregując decyzje z tych drzew, najczęściej przez głosowanie. Dawał on gorsze wyniki niz klasyfikator najbliższych sąsiadow. Na danych testowych,skuteczność klasyfikatora knn wynosiła 0.45 podczas gdy skuteczność lasów losowych tylko 0.26. Jednak zaletą tego klasyfikatora jest to, że buduje on już wcześniej model i decyzja dla danych wejściowych studenta obliczana jest bardzo szybko, w przeciwieństwie do klasyfikatora knn. Kod w R:

```
library(randomForest)
classifierf<-randomForest(data[,1:50], data[,51])
predictRf <- function(student) {
  return(round(predict(classifierf, student)))
}
```

rf.R

5.0.3. Algorytmy używane przy predykcji ocen

Przy predykcji ocen użyliśmy wyłącznie klasyfikatora najbliższych sąsiadów. W przypadku wyboru algorytmu uczenia nadzorowanego, musielibyśmy trenować nasz klasyfikator dla każdego przedmiotu, co przy dużej liczbie przedmiotów jest zadaniem dość karkołomnym. Dlatego zdecydowaliśmy się na algorytm uczenia nienadzorowanego, który dobrze się sprawdzał w poprzednich predykcjach. Tutaj algorytm wybiera 100 najbliższych sasiadów i wybiera tych, którzy wzięli dany przedmiot, następnie przelicza średnią ocen wybranych studentów z tego przedmiotu i zwraca ją jako wynik klasyfikacji.

Rozdział 6

Organizacja pracy oraz podział obowiązków

6.1. Praca nad systemem

Za namową zamawiającego na początku zdecydowaliśmy się zrealizować projekt za pomocą technologii koncernu Microsoft. Otrzymaliśmy od niego licencję Bizspark która dała nam licencję na swobodne wykorzystywanie produktów Microsoftu przez 2 lata. Dodatkowo Bizspark umożliwiał korzystanie z usługi Azure w zakresie abonamentu w wysokości 150 euro na miesiąc. Dlatego wybraliśmy chmurę Azure jako nasz serwer. Planowaliśmy dodatkowo zrealizować obliczanie predykcji za pomocą Sql Server Analysis Services a frontend za pomocą .NET.

Z projektem jednak wiązały się dość znaczące problemy. Pierwszym z nich była zmiana technologii użytej w celu obliczania predykcji, na którą zdecydowaliśmy się w lutym. Przed rozpoczęciem pracy nad projektem nikt z nas nie znał możliwości Sql Server Analysis Services (SSAS) ani nie tworzył strony internetowej w .NET. O ile z .NET nie było znaczących problemów, usługi analityczne Sql Servera stanowiły dla nas barierę nie do przejścia. Ze względu na dość kiepsko udomkumentowną technologię SSAS, brak obiecanych szkoleń z tej technologii oraz brak zajęć na uniwersytecie jej poświęconym a także małej elastyczności algorytmów tej usługi podjęliśmy decyzję o rezygnacji z tego narzędzia. Zdecydowaliśmy się na wykorzystanie języka skryptowego R, z którym dobrze zaznajomione były 2 osoby w zespole. Zmieniliśmy także realizację frontendu z .NET na Django ponieważ główny powód wyboru .NET - pluginy dedykowane dla SSAS, przestał być ważny dla projektu. Z Django obeznane były wszystkie osoby w zespole, a dodatkowym atutem tego wyboru był jeden członek zaspołu posiadający doświadczenie zawodowe w pisaniu aplikacji w Pythonie.

Kolejnym znaczącym problemem przy realizacji projektu była mocno utrudniona praca związana z analizą danych i dobieraniem optymalnych algorytmów. Zamawiający obiecał nam dostarczenie zaszumionych danych z systemu USOS. Czekaliśmy na nie, ponieważ trudno było samemu wymyślić i wygenerować nietrywialne korelacje, zbliżone do rzeczywistości.

Próbka takich danych mocno pomogłaby nam z doborem algorytmów predykcyjnych. Jednak zaszumienie okazało się w praktyce niemożliwe a "zwykłych" danych nie mogliśmy otrzymać z powodu ustawy o ochronie danych osobowych. O tych ograniczeniach praktycznie dowiedzieliśmy się dopiero w kwietniu, co mocno nam popsuło pracę nad algorytmami predykcyjnymi i analizą ich jakości która niestety nie została wykonana na danych produkcyjnych.

6.2. Organizacja Pracy

Naszym repozytorium był git. Drzewo projektu w repozytorium prezentuje się następująco:

TODO : tu będzie drzewko naszego repo.

Przydział zadań do osób i planowanie pracy zostały zrealizowane za pomocą portalu Redmine.

Praca na początku projektu przebiegała dość chaotycznie, z długimi przerwami. Dużą blokadą były dla nas problem z efektywną pracą nad predykcjami z SSAS oraz brak danych bliższych rzeczywistości utrudniający nam wyjście poza teoretyczne rozważanie o algorytmach. Jednak pewne prace były wzwiązku z tym systemem wykonywane, zbudowaliśmy pewną strukturę danych, wygenerowaliśmy dane testowe i próbowaliśmy generowac predykcję. Z wszystkich tych rzeczy (nie licząc części danych testowych) zrezygnowaliśmy w dalszej fazie realizacji projektu.

Realną pracę wychodzącą poza studiowanie dokumentacji i metodę prób i błędów prowadząca do nikąd zaczęliśmy dopiero po decyzji o zmianie technologii w lutym. Praca odbywała się wtedy w cotygodniowych iteracjach, a znacząco przyśpieszyła w kwietniu po ostatecznej decyzji o braku możliwości otrzymania jakichkolwiek danych przypominających rzeczywiste. Niestety ze względu na niezbyt długi pozostały wówczas czas nie udało nam się zrealizować wszystkich funkcjonalności, w szczególności integracji z USOS (przez brak czasu ale także utrudnioną współpracę z zamawiającym).

6.3. Podział Obowiązków

TODO (przy działającym serwisie rzetelnie opiszę wkład osób)
W projekcie Hermes podział pracy był następujący:

- **Tomasz Grabowski** :
  - Pisanie licencjatu : jego redakcja, formatowanie, wstawienie zrzutów ekranu. Napisanie wszystkich rozdziałów poza rozdziałem 5 (autorstwa Krzystofa Rutkowskiego) oraz tym podrozdziałem (każdy sam opisywał swój wkład).
  - Pomoc przy tworzeniu frontendu, w szczególności: edycja szablonu wyświetlającego wyniki predykcji oraz rekomendacji, obliczanie i wyświetlanie statystyk każdego przemiotu (po uzyciu w tym szablonie opcji info), realizacja tego elementu w Django (kod w języku Python).
  - Generowanie danych testowych i przypadków testowych .

  - Funkcjonalne testowanie systemu.
  - We wstępnej wersji projektu (gdy jeszcze korzystaliśmy z Sql Server Analysis Services) tworzenie przypadków oraz danych testowych a także projektowanie kostki OLAP i sposobu wykorzystania modułu Data Mining w celu obliczenia rekomendacji.

- **Adam Markiewicz** - Administrowanie serwerem umieszczonym na Azure, pomoc przy tworzeniu frontendu.
- **Albert Rozmus** - Pomoc przy tworzeniu frontendu i serwera WWW, prezentacja.
- **Krzysztof Rutkowski** - Tworzenie algorytmów uczenia maszynowego służących predykcji i rekomendacji, komunikacja funkcji i modeli uczenia maszynowego z systemem.
- **Wiktor Zuba** - Tworzenie danych testowych do bazy oraz ich relacyjnego modelu.

Plakat został zrealizowany przez osobę wynajętą przez nas w tym celu.

Rozdział 7

Podsumowanie

Projekt napotkał na duże przeszkody w trakcie realizacji. Problemy z zamawiającym, z otrzymaniem sensownych danych testowych oraz z nauką Sql Server Analysis Services a także skutecznym wykorzystaniem jego algorytmów z dość znaczącymi limitacjami mocno wpłynęły na czas spędzony na efektywnej pracy nad projektem. Z ubolewaniem przyznajemy, iż realną pracę przynoszącą progres a nie błądzenie w ślepych zaułkach, książkach i dokumentacjach zaczęliśmy dopiero w marcu.

W celu realizacji projektu i usprawnienia pracy musieliśmy podjąć decyzję o zmianie technologii. Umożliwiło nam to mieć większy wpływ na tworzone algorytmy predykcyjne oraz sposób ich działania. Dużym plusem tej zmiany była możliwosc wykorzystania solidnej znajomosc teorii systemów decyzyjnych oraz jej zastosowania w R przez członków zespołu. Zmiana na Django także była skutkiem lepszej znajomości Pythona przez zespół. Uważamy, iż zmiany okazały się w końcowym rozrachunku pozytywne, ponieważ udało nam się zaimplementować system który zawiera większość planowanych na początku funkcjonalności. Dodatkowo, dzięki zastosowanym technologiom, w miarę łatwe jest rozwijanie systemu. W celu zaimplementowania nowych typów predykcji wystarczy napisać nowy skrypt w R i załączyć go do systemu oraz zintegrować z frontendem.

Problemy przy pracy nad projektem niestety uniemożliwiły spełnienie naszych wszystkich założeń. Zabrakło nam czasu na zaimplementowanie funkcjonalności dla pracowników predykujących popularność wybranych przedmiotów wśród studentow. Z przyczyn niezależnych od nas nie udało nam się również przetestować napisanych przez nas algorytmów na prawdziwych danych, przez co nie mieliśmy możliwości praktycznej weryfikacji predykcji.

Mamy jednak nadzieję, iż pomimo tych trudności system sprawdzi się na prawdziwych danych i okaże się realną, efektywną pomocą dla studentów.

Bibliografia

[AZR] Microsoft *dokumentacja techniczna Azure*
http://azure.microsoft.com/en-us/documentation/

[PTN] Python Software Foundation *dokumentacja techniczna języka Python*
https://docs.python.org/2.7/

[DGO] Django Software Foundation *dokumentacja techniczna Django*
https://docs.djangoproject.com/en/1.8/

[R] R Development Core Team *dokumentacja techniczna języka R*
http://cran.r-project.org/manuals.html

[RSV] Simon Urbanek *dokumentacja techniczna serwera R - modułu RServe*
http://www.rforge.net/Rserve/doc.html