Uczenie preferencji I

1 Wstęp

Uczenie preferencji (ang. preference learning) jest dziedziną z pogranicza uczenia maszynowego oraz wielokryterialnego wspomagania decyzji (ang. multiple criteria decision analysis). Jego zadaniem jest modelowanie preferencji użytkownika (decydenta) za pomocą metod uczenia maszynowego. Głównymi wyróżnikami modelowania preferencji jest możliwosć wyjaśnienia decyzji oraz zinterpretowania modelu. Ponadto metody te muszą zachować zgodność rozwiązania z ograniczeniami podanymi przez decydenta. Oznacza to, że jeżeli dla decydenta dana cecha w obiekcie jest preferowana to model powinien to odzwierciedlić w rozwiązaniu.

Takimi ograniczeniami mogą być między innymi:

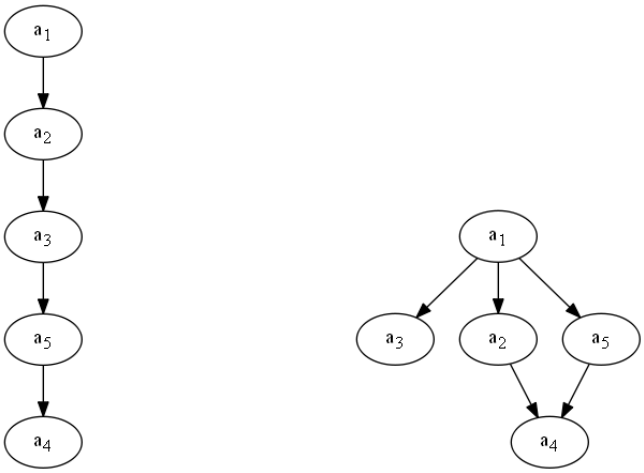
- **monotoniczność** - niektóre cechy mogą mieć jasno wskazany kierunek preferencji i np. rozwiązanie gdzie niższa jakość obiektu lub wyższa cena wpływa pozytywnie na wyższy ranking jest zazwyczaj rozwiązaniem błędnym.
- **progi** - użytkownik może podać pewien zestaw progów, które odzwierciedlają jego sposób myślenia, np. pewne przedziały wartości są dla niego nierozróżnialne. Innym przykładem może być również sytuacja, gdy różnica pomiędzy dwoma obiektami na jakiejś cesze jest wystarczająco duża, to taka cecha ma znaczący wpływ na rozwiązanie. W takiej sytuacji oczekiwane rozwiązanie nie może podać obiektu gorszego na tej cesze jako lepsze.
- **zależności między cechami** - decydent może zdefiniować, że pewne grupy cech, jeżeli występują razem, dają pozytywny lub negatywny wpływ na jego preferencje.

- **zrozumiałość modelu** - otrzymany model musi być zrozumiały dla decydenta i zgodny z jego sposobem myślenia (decydent zazwyczaj nie wykonuje w umyśle bardzo skomplikowanych obliczeń w przestrzeni wielowymiarowej).

Główne problemy rozwiązywane przez preference learning są problemami rankingu:

- **Label ranking** - rozszerzenie problemu wieloetykietowalnej klasyfikacji (multi label classification); dla każdego obiektu tworzony jest ranking etykiet, które są do niego przypisane
- **Instance ranking - problem sortowania** - przypisanie wariantu do jednej z predefiniowanych klas, które są ułożone względem preferencji. Pomiędzy klasami istnieją relacje: bardziej preferowana - mniej preferowana klasa.
- **Object ranking** - typowy problem rankingu, czyli nie posiadamy predefiniowanych klas i tworzymy ranking wariantów.
- **Systemy rekomendacyjne** - ranking obiektów dla każdego użytkownika.

W zależności od zastosowań możemy mieć do czynienia z rankingiem zupełnym (Rysunek 1 (a)), czyli między każdą parą obiektów możemy powiedzieć który obiekt jest bardziej preferowany. Jeżeli między parą obiektów może istnieć nieporównywalność, czyli sytuacja, w której nie jesteśmy w stanie stwierdzić na podstawie posiadanej informacji czy zachodzi relacja preferencji w którąkolwiek stronę to mówimy o rankingu częściowym (Rysunek 1 (b)).



(a) (b)

Rysunek 1: Ranking zupełny (a) i częściowy (b)

2 Miary oceny

W celu analizy otrzymanych rozwiązań konieczne jest wprowadzenie odpowiednich miar oceny, które będą w stanie porównać rankingi między sobą. Załóżmy, że r jest funkcją
Błąd rankingowy dla rankingów r i $\hat{r}$ i n obiektów a_i:

- odległość Spearmana - kwadrat różnic pozycji w obu rankingach dla wszystkich obiektów: $D_S(r, \hat{r}) = \sum_{i=1}^{n}(r(a_i) - \hat{r}(a_i))^2$.
- znormalizowana odległość Spearmana: $\frac{3D_S(r,\hat{r})}{n(n^2-1)}$
- odległość tau Kendalla - liczba par obiektów dla których ranking się różni: $D_\tau(r, \hat{r}) = |\{(i, j)|r(a_i) < r(a_j) \wedge \hat{r}(a_i) > \hat{r}(a_j)\}|$
- znormalizowana odległość tau Kendalla: $\frac{2D_\tau(r,\hat{r})}{n(n-1)}$
- Area under the ROC curve (AUC)
- Odległości ważone $D_w(r, \hat{r}) = \sum_{i=1}^{n} w_i d_{a_i}(r, \hat{r})$
  - różnice dla wysokich pozycji rankingu docelowego są bardziej istotne: $w_i = \frac{1}{log(r(a_i)+1)}$

Uwagi dla sortowania z dwoma klasami:

- AUC jest równe: 1 - znormalizowana odległość tau Kendalla
- dokładność (ang. accuracy) jest równe 1 - znormalizowana odległość Spearmana

3 Metody uczenia preferencji

3.1 Metody bazujące na funkcji użytecznosci

Jednym ze sposobów na stworzenie rankingu jest wykorzystanie metod bazujących na funkcji użyteczności U (ang. utility functions, score functions). Obiekt a_i przewyższa a_j ($a_i \prec a_j$) jeżeli $U(a_i) > U(a_j)$. Jako funkcję użyteczności można przyjąć dowolną metodę uczenia maszynowego, która spełnia ograniczenia podane we wstępie. Takimi metodami mogą być boosted trees, które implementują ograniczenia monotoniczności chech:

- **XGBoost**
- LightGBM
- CatBoost
- Scikit-Learn histogram gradient booster

W ramach zajęć zostanie przedstawione działanie algorytmu XGBoost dla problemu sortowania z 2 klasami dla problemu Lectures Evaluation, które znajduje się w notebooku: xgboost.ipynb.

3.2 Metody bazujące na uczeniu relacji preferencji

Innym sposobem jest stworzenie metod, które przyjmują na wejście parę obiektów, porównują je i zwracają informację o zachodzeniu bądź nie relacji preferencji. Następnie takie porównania parami są agregowane do rankingu.

Przykładem takiej metody jest metoda **RankSVM**, która bazuje na metodzie SVM zamieniając problem rankingu na problem klasyfikacji. Metoda działa w 2 etapach:

1. Należy obliczyć róznicę w ocenach dla każdej pary wariantów które pochodzą z róznych klas.

2. Na powstałych różnicach wykonujemy metodę SVM.

Przykład wykorzystania metody RankSVM został przedstawiony w notebooku: RankSVM.ipynb.

3.3 Metody bazujące na lokalnej agregacji preferencji

Te metody bazują na założeniu, iż obiekty podobne do siebie powinny być w podobny sposób preferowane przez decydenta. Jednym z przykładów metod uczenia maszynowego, które może zostać wykorzystane w tego typu problemie jest metoda **KNN**.
Przykład wykorzystania metody KNN został przedstawiony w notebooku: Knn.ipynb.

Literatura

[1] Hüllermeier, E., & Fürnkranz, J. (2012). Preference Learning: A Tutorial Introduction.

[2] Fürnkranz J., Hüllermeier E. (2011) Preference Learning