Politechnika Warszawska
Wydział Matematyki i Nauk Informacyjnych

PRACA DYPLOMOWA MAGISTERSKA
NA KIERUNKU MATEMATYKA

STATYSTYCZNE METODY REGRESJI PORZĄDKOWEJ

Autor:
Marta Sommer

Promotor:
prof. nzw. dr hab.
Przemysław Grzegorzewski

Warszawa, czerwiec 2015

.....

podpis promotora

.....

podpis autora

Spis treści

Wstęp

Regresja porządkowa (ang. *ordinal regression*) jest jednym z działów uczenia maszynowego. Od problemu klasycznej regresji różni ją to, że zmienna odpowiedzi jest dyskretna, natomiast od problemu klasyfikacji to, że zmienna odpowiedzi ma pewien naturalny porządek. Regresja porządkowa zajmuje się zatem uczeniem i oceną jakości predyktora, który modeluje zmienną uporządkowaną i skończoną. Problem regresji porządkowej rozwija się dość szybko m.in. dlatego, że ma on bardzo wiele zastosowań, choćby w systemach rekomendacji, czy bardzo popularnych wyszukiwarkach internetowych. Prześledźmy to na konkretnym przykładzie. Wyobraźmy sobie sytuację, że chcielibyśmy określić, w jakim stopniu danemu człowiekowi spodoba się sprzedawany przez nas produkt. Mamy do dyspozycji zbiór treningowy składający się z wektora zmiennej objaśniającej $\mathbf{x} = (x_1, \ldots, x_d)$, gdzie x_i są różnymi cechami określającymi daną osobę (np. płeć, wiek, wykształcenie, ...). Cechy te – podobnie jak w przypadku zwykłej regresji – mogą być zarówno ciągłe, jak i dyskretne. Mamy również dostęp do zmiennej objaśnianej $\mathbf{y} = (y_1, \ldots, y_r)$, będącej wektorem zero-jedynkowym, wskazującym która klasa została przypisana danemu rekordowi. W naszym przykładzie, zmienną odpowiedzi mogłyby być na przykład: *zdecydowanie mi się nie podoba*, *nie podoba mi się*, *nie mam zdania*, *podoba mi się*, *zdecydowanie mi się podoba*. Widać wyraźnie, że są one uporządkowane.

Najprostszym podejściem do tego typu problemu byłoby zignorowanie kolejności zmiennej odpowiedzi i potraktowanie go, jak zwykłą klasyfikację. W takim przypadku tracimy jednak pewną informację, która prawdopodobnie mogłaby przyczynić się do poprawy naszego klasyfikatora. Idąc w drugą stronę, można potraktować nasz problem, jak zwykłą regresję, zamieniając zmienną odpowiedzi na pewną zmienną ciągłą i to ją modelując, a następnie z powrotem dyskretyzować. Pojawia się tu jednak problem, jak optymalnie zrobić taką transformację, uwzględniając chociażby fakt, że nasza odpowiedź niekoniecznie jest monotoniczna (tzn. np. różnica między *nie podoba mi się* a *nie mam zdania* wcale mnie musi być taka sama, jak między *podoba mi się* a *zdecydowanie mi się podoba*).

Możemy wyróżnić dwa główne nurty w regresji porządkowej:

- prognoza konkretnej obserwacji (nacisk kładziony jest tu na wyznaczenie konkretnego $\mathbf{y}$ dla konkretnego $\mathbf{x}$ np. czy potencjalnemu klientowi spodoba się dany produkt),
- uszeregowanie kilku obserwacji (celem nie jest poznanie estymacji konkretnej zmiennej odpowiedzi, ale takie uszeregowanie kilku rekordów, by te najbardziej preferowane znalazły się na samej górze, a te najmniej na samym dole np. w jakiej kolejności powinny wyświetlić się znalezione strony w wyszukiwarce).

W mojej pracy zajmować się będę przede wszystkim pierwszym punktem, lecz nakreślę też kilka podejść dotyczących drugiego.

Rozdział 1

Opis teoretyczny dostepnych metod

W tym rozdziale opracuję kilka znanych i opisanych w literaturze podejść do regresji porządkowej. Można je podzielić na kilka grup:

- korzystające z dostępnych metod klasyfikacji, m.in.:
  - metoda zaproponowana przez E. Franka i M. Halla,
- modyfikujące dostępne metody klasyfikacji, m.in.:
  - SVM,
  - sieci neuronowe,
  - procesy gaussowskie,
- metody stworzone specjalnie dla regresji porząckowej, m.in.:
  - model proporcjonalnych szans.

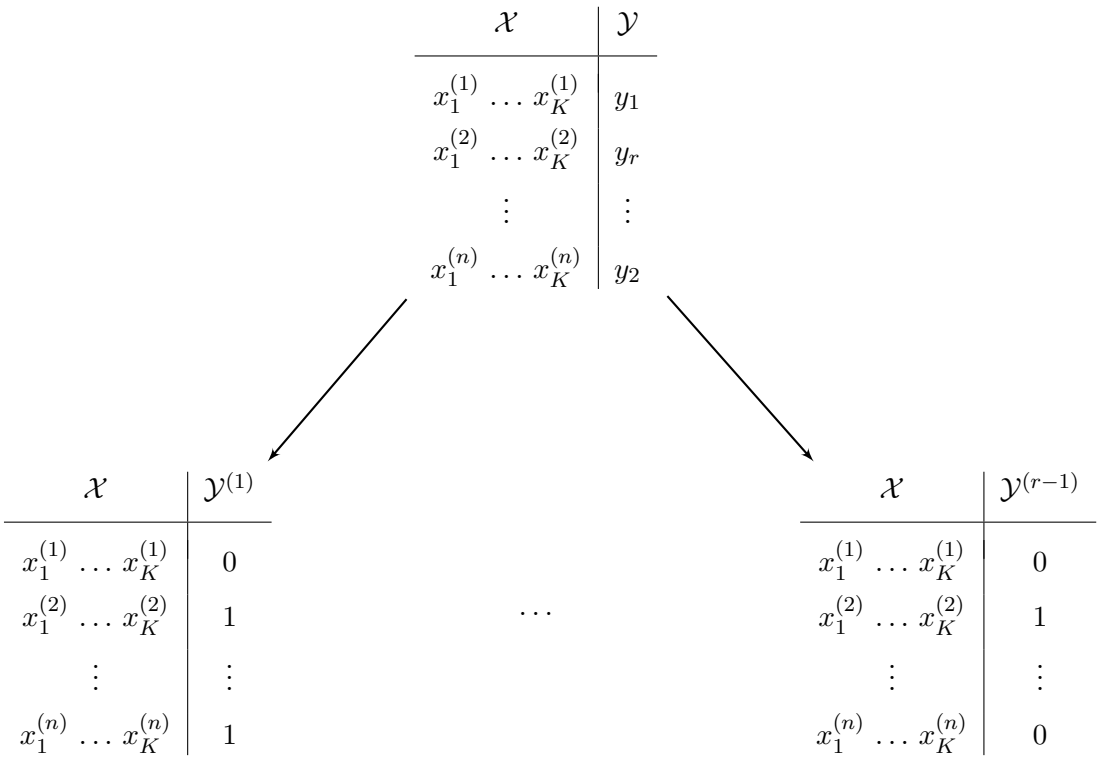
1.1. Metoda zaproponowana przez E. Franka i M. Halla

Podejście Franka i Halla (por. [1]) do zagadnienia regresji porządkowej opiera się nie na stworzeniu nowego modelu, ale na odpowiednim przedefiniowaniu zbioru danych, a następnie na sprowadzeniu zadania do problemu zwykłej klasyfikacji z dwoma klasami. Przekształcamy zatem r–klasowy model regresji porządkowej do $(r-1)$ dwuklasowych problemów klasyfikacji.

Na wejściu otrzymujemy n par $(\mathbf{x}, y)$, gdzie $\mathbf{x} = (x_1, \ldots, x_K)^T$ jest wektorem cech, a $y \in \{y_1, \ldots, y_r\}$ reprezentuje klasę, do której należy dana obserwacja. Zakładamy rosnący porządek na zbiorze $\{y_1, \ldots, y_r\}$, tzn. $y_1 \prec \ldots \prec y_r$. Algorytm do budowy modelu wygląda następująco:

1. Modyfikujemy zbiór uczący (otrzymując $r-1$ nowych zbiorów uczących).
2. Dla każdego nowo uzyskanego zbioru danych dopasowujemy zwykły model klasyfikacyjny (np. drzewo) taki, który zwraca prawdopodobieństwa przynależności do klas.

3. Robimy predykcję dla nowej obserwacji.



Rysunek 1.1: Modyfikacja przykładowego zbioru uczącego.

Ad. 1)

Chcemy otrzymać $r-1$ nowych zbiorów o zero-jedynkowej zmiennej odpowiedzi. W jaki sposób to zrobić? Macierz atrybutów pozostaje bez zmian, zmienia się jedynie wektor zmiennej odpowiedzi według zasady:

$$\begin{aligned} y_i^{(1)} &= \mathbb{I}\{y_i > y_1\} \\ &\vdots \\ y_i^{(r-1)} &= \mathbb{I}\{y_i > y_{r-1}\} \end{aligned}$$

Ad. 3)

Dla nowego wektora atrybutów $\mathbf{x}$ robimy predykcję na $r-1$ modelach uzyskanych w punkcie 2). Zwracamy jednak nie predykcję klasy, ale prawdopodobieństwo przynależności do klasy pierwszej. Uzyskujemy w ten sposób $r-1$ następujących prawdopodobieństw:

$$\begin{aligned} \mathbb{P}(y &> y_1) \\ &\vdots \\ \mathbb{P}(y &> y_{r-1}). \end{aligned}$$

Nas natomiast interesują prawdopodobieństwa:

$$\begin{aligned} \mathbb{P}(y &= y_1) \\ &\vdots \\ \mathbb{P}(y &= y_{r-1}). \end{aligned}$$

Łatwo otrzymamy korzystając z następującego wzoru łańcuchowego:

$$\begin{aligned} \mathbb{P}(y = y_1) &= 1 - \mathbb{P}(y > y_1) \\ &\vdots \\ \mathbb{P}(y = y_i) &= \mathbb{P}(y > y_{i-1}) - \mathbb{P}(y > y_i) \qquad \text{dla} \quad i = 2, \ldots, r-1 \\ &\vdots \\ \mathbb{P}(y = y_r) &= \mathbb{P}(y = y_{r-1}). \end{aligned}$$

Ostatecznie, nowej obserwacji przypisujemy klasę, której prawdopodobieństwo ($\mathbb{P}(y = y_i)$) było największe.

1.2. Sieci neuronowe

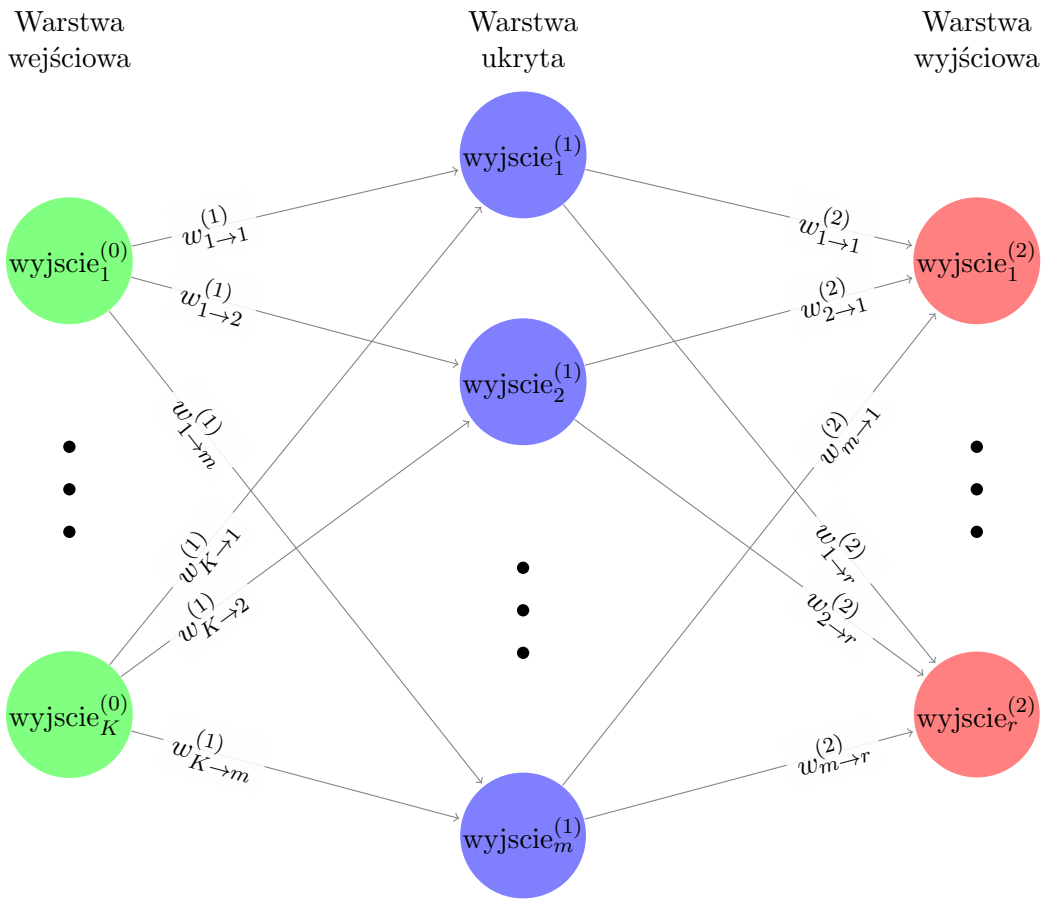
Sieci neuronowe to bardzo proste i szeroko stosowane narzędzie zarówno w problemach regresji, jaki i klasyfikacji. Znalazło ono również swoje zastosowanie w regresji porządkowej (por. [2]).

Standardowo, na wejściu otrzymujemy zbiór uczący w postaci n par $(\mathbf{x}, \mathbf{y})$, gdzie $\mathbf{x} = (x_1, \ldots, x_K)^T$ jest wektorem cech, a $\mathbf{y} = (y_1, \ldots, y_r)^T$ wektorem zero-jedynkowym reprezentującym klasę, do której należy dana obserwacja. Zakładamy rosnący porządek na zbiorze $\{y_1, \ldots, y_r\}$, tzn. $y_1 \prec \ldots \prec y_r$.

W przeciwieństwie do zwykłej klasyfikacji, nasza sieć neuronowa będzie zakładać porządek zmiennej odpowiedzi. W jaki sposób? Podobnie, jak w metodzie proporcjonalnych szans, jako wektor wyjściowy, zamiast wektora $\mathbf{y} = (0, 0, \ldots, 1, \ldots, 0)^T$, mającego jedynkę na i–tym miejscu, jeśli obserwacja należała do i–tej klasy, rozważać będziemy wektor $\mathbf{y} = (1, 1, \ldots, 1, \ldots, 0)^T$, mający jedynki na miejscach od pierwszego do i–tego.

Otrzymujemy w ten sposób sieć neuronową o K neuronach w warstwie wejściowej, z których każdy reprezentuje inną cechę z wektora $\mathbf{x}$, jednej (bądź więcej) warstwie ukrytej o m neuronach i warstwie wyjściowej zawierającej r neuronów, które reprezentują odpowiedź $\mathbf{y}$ w formie opisanej powyżej. Za funkcję przejścia przyjmiemy funkcję sigmoidalną $f(x) = \frac{1}{1+e^{-x}}$, dobrze reprezentującą przynależność do danej klasy jako prawdopodobieństwo. Może się zdarzyć, że wyjściowy wektor nie będzie ciągiem malejącym (co przeczył intuicji), jednak nie jest to konieczne do robienia predykcji.

Uczenie sieci neuronowej będzie się odbywało algorytmem propagacji wstecznej z kwadratową funkcją straty (można też użyć jakiejś innej np. entropii). Algorytm wygląda następująco:



Rysunek 1.2: Przykładowa sieć neuronowa.

1. Wybieramy małe wagi początkowe oraz niewielki współczynnik $\eta > 0$.
2. Losujemy parę $(\mathbf{x}, \mathbf{y})$ ze zbioru uczącego.
3. Przebiegamy sieć w przód.
4. Przebiegamy sieć w tył (licząc błąd dla każdego neuronu).
5. Zmieniamy wagi.
6. Dopóki nie osiągniemy zadowalająco niskiego błędu, wracamy do punktu 2).

Ad. 3)

Dla każdego neuronu obliczamy wartość wejściową ze wzoru:

$$wejscie_j^{(i)} = \sum_{k:\ \exists\ w_{k\to j}^{(i)}} \left(w_{k\to j}^{(i)} \cdot wyjscie_k^{(i-1)} \right),$$

gdzie $wyjscie_i^{(0)} = x_i$. A następnie wyjściową:

$$wyjscie_j^{(i)} = f\left(wyjscie_j^{(i)} \right).$$

Ad. 4)

Dla warstwy wyjściowej błąd ma postać:

$$\delta_j = -2 \cdot wyjscie_j^2 \cdot (1 - wyjscie_j)^2 \cdot (y_j - wyjscie_j),$$

zaś dla warstw ukrytych:

$$\delta_j^{(i)} = wyjscie_j^{(i)} \cdot \left(1 - wyjscie_j^{(i)}\right) \cdot \sum_{k:\ \exists\ w_{j\to k}^{(i+1)}} \left(w_{j\to k}^{(i+1)} \cdot \delta_k^{(i-1)}\right).$$

Ad. 5)

Modyfikacja wag przebiega następująco:

$$w_{k\to j}^{(i)^{(new)}} = w_{k\to j}^{(i)^{(old)}} - \eta \cdot \delta_j^{(i)} \cdot wyjscie_k^{(i-1)}.$$

Predykcja opiera się już tylko na przejściu algorytmu w przód z nowymi obserwacjami wejściowymi $\mathbf{x}$ i ustaleniu progu (najczęściej równego $0,5$), klasyfikującego neuron wyjściowy jako jedynkę. Skanujemy wektor wyjściowy zaczynając od y_1 i kończymy, gdy pierwszy raz natkniemy się na 0. Przypisujemy obserwacji taką klasę, ile wynosił znaleziony przez nas ciąg jedynek.

Literatura

[1] Frank E., Hall M., A simple approach to ordinal classification, *Proceedings of the European Conference on Machine Learning*, Freibourg, Niemcy, 2001, str. 146–156.

[2] Cheng J., Wang Z., Pollastri G., A neural network approach to ordinal regression, *Neutral Networks*, Hong Kong, 2008.

Marta Sommer
Nr albumu 237503

Warszawa, 29 kwietnia 2015

Oświadczenie

Oświadczam, że pracę magisterską pod tytułem „Statystyczne metody regresji porządkowej", której promotorem jest prof. nzw. dr hab. Przemysław Grzegorzewski wykonałam samodzielnie, co poświadczam własnoręcznym podpisem.

.....
Marta Sommer