Multivariate Adaptive Regression Splines (MARS)

Paweł Teisseyre*

10 marca 2015

Poniższy dokument zawiera opis metody MARS oraz opis implementacji, która jest wzorowana na pakiecie `earth` w R (`http://www.milbo.org/doc/earth-notes.pdf`).

1 Oznaczenia

- n, liczba obserwacji
- p, liczba zmiennych
- $\boldsymbol{y} = (y_1, \ldots, y_n)'$, wektor zmiennej zależnej
- $\bar{\boldsymbol{y}}$, średnia wektora $\boldsymbol{y}$
- X, macierz zmiennych niezależnych o wymiarach $n \times p$
- $x_1, \ldots, x_p$, zmienne niezależne (kolumny macierzy X)
- $x_{j,i}$, i-ta współrzędna zmiennej j
- B_j, j-ta przekształcona zmienna
- $B_{j,i}$, i-ta współrzędna zmiennej przekształconej j
- $tr(A)$, ślad macierzy A
- A', transpozycja macierzy A
- $||x||$, norma euklidesowa wektora x (pierwiastek z sumy kwadratów współrzędnych)
- Funkcja "zawiasowa" (ang. hinge function):

$$(x-t)_+ = \begin{cases} x-t & \text{jeżeli } x \geqslant t \\ 0 & \text{wpp} \end{cases}$$

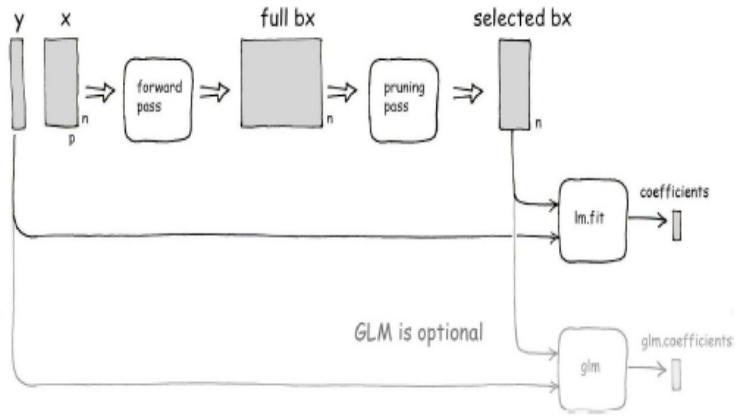
2 Opis metody

Metoda MARS (Multivariate Adaptive Regression Splines) została zaproponowana w pracy [1]. Praca [2] zawiera uzupełnienie dotyczące szczegółów implementacyjnych. Największą zaletą metody MARS jest możliwość znajdowania przekształceń oryginalnych zmiennych oraz interakcji między zmiennymi. Procedura składa się z trzech kroków:

*teisseyrep@ipipan.waw.pl

1. Krok FORWARD polega na iteracyjnym tworzeniu nowych zmiennych, w oparciu o zmienne utworzone w poprzednich iteracjach. W każdym kroku dodajemy dwie nowe zmienne. Procedura jest zatrzymywana jeżeli jest spełniony jeden z warunków stopu. W wyniku działania procedury FORWARD powstaje model, który ma zwykle zbyt wiele zmiennych i jest nadmiernie dopasowany do danych treningowych. Konieczna jest więc eliminacja zmiennych.
2. Krok BACKWARD polega na krokowej eliminacji zmiennych znalezionych w kroku FORWARD. W tym kroku znajdujemy model (t.j. podzbiór zmiennych) który minimalizuje pewną funkcję kryterialną (np. GCV- uogólniona kroswalidacja).
3. BUDOWA MODELU w oparciu o znaleziony w kroku 2 podzbiór zmiennych. W tym kroku możemy użyć dowolnego modelu regresji lub klasyfikacji.

Schemat procedury jest pokazany na rysunku 1. Krok FORWARD jest znacznie bardziej wyma-



Rysunek 1: Schemat procedury mars (źródło: dokumentacja pakietu `earth`).

gający obliczeniowo niż krok BACKWARD.

2.1 Krok FORWARD

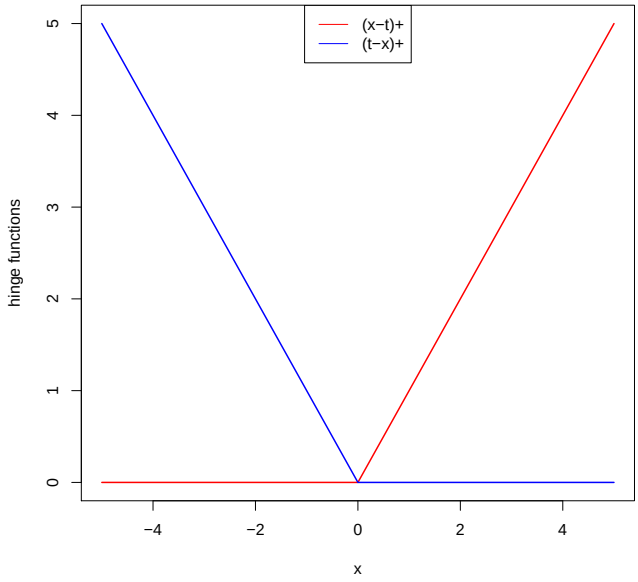
Rozważmy iterację numer I. Niech $B = [B_1, \ldots, B_M]$ będzie macierzą o wymiarach $n \times M$, która zawiera w kolumnach zmienne $B_1, \ldots, B_M$ wyznaczone w poprzednich iteracjach $1, \ldots, I-1$. Funkcje $B_1, \ldots, B_M$ nazywamy "rodzicami". Mamy, że $M = 2I - 1$. Pierwsza kolumna macierzy B jest zawsze stała i równa 1, t.j. $B_1 = 1$. W iteracji I znajdujemy dwie nowe zmienne ("potomków"), które mają postać:

$$B_l \cdot (t - x_v)_+$$

oraz

$$B_l \cdot (x_v - t)_+,$$

gdzie $l \in \{1, \ldots, M\}$ jest wybranym rodzicem, x_v jest wybraną zmienną objaśniającą, t jest wybranym progiem dla zmiennej x_v. Rysunek 2 pokazuje funkcje zawiasowe dla zmiennej o zakresie $[-5, 5]$ i $t = 0$.



Rysunek 2: Funkcje zawiasowe.

W każdej iteracji musimy wybrać optymalne wartości l, v, t. Optymalne wartości parametrów znajdujemy poprzez minimalizację następującej funkcji:

$$(l^*, v^*, t^*) = \arg \min_{l,v,t,a_1,\ldots,a_{M+2}} RSS(l, v, t, a_1, \ldots, a_{M+2}),$$

gdzie:

$$RSS(l, v, t, a_1, \ldots, a_{M+2}) = \sum_{i=1}^{n} \left[y_i - \sum_{j=1}^{M} a_j B_j - a_{M+1} B_l \cdot (t - x_v)_+ - a_{M+2} B_l \cdot (x_v - t)_+ \right]^2. \tag{1}$$

Zamiast funkcji (1) będziemy rozważać funkcję

$$RSS(l, v, t, a_1, \ldots, a_{M+2}) = \sum_{i=1}^{n} \left[y_i - \sum_{j=1}^{M} a_j B_j - a_{M+1} B_l \cdot x_v - a_{M+2} B_l \cdot (x_v - t)_+ \right]^2. \tag{2}$$

Okazuje się, że minimalizacja funkcji (2) jest równoważna minimalizacji funkcji (1). W obu przypadkach minimalna wartość funkcji będzie taka sama, choć oszacowania współczynników $a_1, \ldots, a_{M+2}$ mogą być różne. Zauważmy, że dla ustalonych l, v, t musimy rozwiązać problem liniowy z wektorem odpowiedzi $\boldsymbol{y}$ i macierzą eksperymentu o wymiarach $n \times (M + 2)$ postaci

$$\tilde{B} = [B_1, \ldots, B_M, B_l \cdot x_v, B_l \cdot (x_v - t)_+].$$

Dla efektywnej implementacji, istotne jest to, że tylko ostatnia kolumna macierzy $\tilde{B}$ zależy od t. To pozwala na szybkie rozwiązanie problemów liniowych dla różnych wartości t (przy zadanych l, v). Poniższy schemat 1 pokazuje przebieg kroku FORWARD. $B_{l,i}$ to i-ta obserwacja l-tej zmiennej przekształconej.

Algorithm 1: Schemat kroku FORWARD

Initialize: $B_1 = 1$, $RSS^* = Inf$, $I = 1$, $M = 1$
repeat
  for $l \leftarrow 1$ **to** M **do**
    for $v \leftarrow 1$ **to** p **do**
      for $t \in \{x_{v,i} : B_{l,i} > 0\}$ **do**
        $RSS_{act} = \min_{a_1,\ldots,a_{M+2}} RSS(l, v, t, a_1, \ldots, a_{M+2})$
        if $RSS_{act} < RSS^*$ **then**
          $RSS^* \leftarrow RSS_{act}$
          $t^* \leftarrow t$
          $v^* \leftarrow v$
          $l^* \leftarrow l$
        end
      end
    end
  end
  $B_{M+1} \leftarrow B_{m^*}(t^* - x_{v^*})_+$
  $B_{M+2} \leftarrow B_{m^*}(x_{v^*} - t^*)_+$
  $I \leftarrow I + 1$
  $M \leftarrow M + 2$
until *convergence is achieved*;

Opisana powyżej procedura zostaje zatrzymana, jeżeli jest spełniony jeden z poniższych warunków:

1. osiągnięta została maksymalna liczba iteracji n_k (gdzie n_k to parametr)
2. $R_I^2 > 0.999$, gdzie:
$$R_I^2 := 1 - RSS^*/SST$$
to wpółczynnik determinacji w iteracji I, natomiast:
$$SST := \sum_{i=1}^{n}(y_i - \bar{\boldsymbol{y}})^2$$
3. $|R_I^2 - R_{I-1}^2| < 0.001$, dla $I > 1$

2.2 Krok BACKWARD

Przypuśćmy że w kroku FORWARD wyznaczyliśmy zmienne $B_1, \ldots, B_{M_{max}}$. Chcemy znaleźć podzbiór $m \subset \{B_1, \ldots, B_{M_{max}}\}$, który minimalizuje uogólnione kryterium kroswalidacyjne:
$$GCV(m) := \sum_{i=1}^{n}\left[\frac{y_i - \hat{y}_i(m)}{1 - \frac{C(m)}{n}}\right]^2,$$
gdzie:

- $\hat{y}_i(m)$ to predykcja dla i-tej obserwacji, obliczona na podstawie modelu m
- $|m|$ to liczba (niestałych) zmiennych w modelu m
- $C(m) = 1 + |m| + pen \cdot |m|$ (pen to dodatkowy parametr kary za liczbę zmiennych w modelu m), B_m to macierz, której kolumny stanowią zmienne odpowiadające podzbiorowi m.

Obliczenie GCV dla wszystkich podzbiorów m zbioru $\{1, \ldots, M_{max}\}$ jest zbyt kosztowne obliczeniowo. Dlatego stosujemy standardową procedurę krokowej eliminacji zmiennych. W pierwszym kroku dopasowujemy model przy użyciu wszystkich zmiennych. W kolejnym kroku usuwamy zmienną, której pominięcie powoduje największy spadek GCV. Procedurę kontynuujemy aż do momentu, w którym usunięcie którejkolwiek ze zmiennych nie przynosi poprawy (t.j. nie obniża GCV). Alternatywnie można użyć innych funkcji kryterialnych (np. AIC/BIC i innych metod przeszukiwania podzbiorów, np. metody selekcji "wprzód").

3 Implementacja

3.1 Dopasowanie modeli liniowych dla różnych t

Poniżej przedstawiamy jak w sposób efektywny wykonać wewnętrzną pętlę w kroku FORWARD. Przypuśćmy że l oraz v są zadane. Pokażemy, jak dopasować szybko modele liniowe (2), dla różnych wartości t. Rozważamy problem liniowy z wektorem odpowiedzi $\boldsymbol{y}$ i macierzą eksperymentu o wymiarach $n \times (M + 2)$ postaci
$$\tilde{B} = [B_1, \ldots, B_M, B_l \cdot x_v, B_l \cdot (x_v - t)_+].$$
Rozwiązaniem problemu jest wektor $a \in R^{M+2}$, który spełnia równanie:
$$Va = c, \tag{3}$$
gdzie $V = B'B$ to macierz symetryczna o wymiarach $(M + 2) \times (M + 2)$, natomiast $c = B'y$ to wektor o $M + 2$ współrzędnych. Uklad równań (3) rozwiązujemy w 3 krokach:

- Wykonaj rozkład Cholesky'ego macierzy $V = LL'$, gdzie L jest macierzą trójkątną dolną (KROK WYMAGAJĄCY OBLICZENIOWO, M^2 operacji)
- Rozwiąż układ równań $Lb = c$ i wyznacz b (M operacji)
- Rozwiąż układ równań $L'a = b$ i wyznacz a (M operacji)

Dwa ostatnie układy można rozwiązać szybko ponieważ macierze po lewej stronie są trójkątne.

W praktyce, może się zdażyć, że kolumny macierzy $\tilde{B}$ będą liniowo zależne. Dlatego macierz V zastępuje się macierzą $V + \epsilon I$, gdzie I jest macierzą identycznościową, natomiast ϵ jest parametrem (w naszej implementacji $\epsilon = 10^{-3}$).

Zauważmy, że w symetrycznej macierzy V, jedynie ostatnia kolumna i ostatni wiersz zależy od t (ostatni wiersz jest równy ostatniej kolumnie). Z kolei, w wektorze c, jedynie ostatnia współrzędna c_{M+2} zależy od t. W następnych paragrafach pokażemy:

- jak szybko obliczyć ostatnią współrzędną wektora c i ostatni wiersz (kolumnę) macierzy V dla różnych wartości t,
- jak wykonać aktualizację rozkładu Cholesky'ego,
- jak wykonać aktualizację RSS.

Implementacja dopasowania modeli liniowych dla różnych węzłów znajduje się w funkcji `fit_lm_knots`.

3.2 Aktualizacja c

Niech $t_1 \leqslant t_2 \leqslant \ldots \leqslant t_N$ będą węzłami dla których chcemy obliczyć c_{M+2}. Poniższa procedura pozwala wyznaczać wartości c_{M+2} dla kolejnych węzłów, w porządku malejącym. Korzystając z własności funkcji zawiasowych, c_{M+2} (dla ostatniego węzła) możemy zapisać jako

$$c_{M+2}(t_N) = \sum_{i=1} y_i B_{l,i} (x_{v,i} - t_N)_+ = \sum_{x_{v,i} \geqslant t_N} y_i B_{l,i} x_{v,i} - t_N \sum_{x_{v,i} \geqslant t_N} y_i B_{l,i}.$$

W kolejnym kroku możemy wyznaczyć c_{M+2} dla przedostatniego węzła:

$$c_{M+2}(t_{N-1}) = c_{M+2}(t_N) + \sum_{t_{N-1} \leqslant x_{v,i} < t_N} x_{v,i} y_i B_{l,i} - t_{N-1} \sum_{t_{N-1} \leqslant x_{v,i} < t_N} y_i B_{l,i} + (t_N - t_{N-1}) \sum_{x_{v,i} \geqslant t_N} y_i B_{l,i}.$$

Analogicznie, korzystając z ostatniej równości, wyznaczamy wartość c_{M+2} dla kolejnych węzłów: $t_{N-2}, \ldots, t_2, t_1$. Implementacja tego kroku znajduje się w funkcji `dotprod1`.

3.3 Aktualizacja V

Zapiszmy macierz V w formie blokowej

$$V = \begin{bmatrix} V_1 & v \\ v' & v_2 \end{bmatrix},$$

gdzie V_1 to macierz $(M+1) \times (M+1)$, która nie zależy od t, v to wektor o $M+1$ współrzędnych, v_2 to liczba. Przypomnijmy, że w powyższej macierzy jedynie wektor v oraz wyraz v_2 zależą od t. Na początek zauważmy że każdą współrzędną wektora v można obliczyć dla różnych węzłów w dokładnie taki sam sposób jak c_{M+2}. Wystarczy we wzorach na c_{M+2} zamienić y_i na $B_{k,i}$, dla $k = 1, \ldots, M+1$.

Pozostało pokazać jak wyznaczyć v_2 dla różnych t. Niech $t_1 \leqslant t_2 \leqslant \ldots \leqslant t_N$ będą węzłami dla których chcemy obliczyć v_2. Poniższa procedura pozwala wyznaczać wartości v_2 dla kolejnych węzłów, w porządku malejącym. Korzystając z własności funkcji zawiasowych, v_2 (dla ostatniego węzła) możemy zapisać jako

$$v_2(t_N) = \sum_{i=1} B_{l,i}^2 (x_{v,i} - t_N)_+ = \sum_{x_{v,i} > t_N} B_{l,i}^2 x_{v,i}^2 - 2t_N \sum_{x_{v,i} > t_N} B_{l,i}^2 x_{v,i} + t_N^2 \sum_{x_{v,i} > t_N} B_{l,i}^2.$$

W kolejnym kroku możemy wyznaczyć v_2 dla przedostatniego węzła:

$$v_2(t_{N-1}) = v_2(t_N) + \sum_{t_{N-1} \leqslant x_{v,i} < t_N} B_{l,i}^2 x_{v,i}^2 - 2t_{N-1} \sum_{t_{N-1} \leqslant x_{v,i} < t_N} B_{l,i}^2 x_{v,i} + t_{N-1}^2 \sum_{t_{N-1} \leqslant x_{v,i} < t_N} B_{l,i}^2 + 2(t_N - t_{N-1}) \sum_{x_{v,i} \geqslant t_N} B_{l,i}^2 x_{v,i} + (t_{N-1}^2 - t_N^2) \sum_{x_{v,i} \geqslant t_N} B_{l,i}^2.$$

Analogicznie, korzystając z ostatniej równości, wyznaczamy wartość v_2 dla kolejnych węzłów: $t_{N-2}, \ldots, t_2, t_1$. Implementacja tego kroku znajduje się w funkcji `dotprod2`.

3.4 Aktualizacja rozkładu Cholesky'ego

Zapiszmy macierz V w formie blokowej

$$V = \begin{bmatrix} V_1 & v \\ v' & v_2 \end{bmatrix},$$

gdzie V_1 to macierz $(M+1)\times(M+1)$, która nie zależy od t, v to wektor o $M+1$ współrzędnych, v_2 to liczba. Przypuśćmy, że mamy wyznaczony rozkład Cholesky'ego macierzy $V_1 = L_1 L_1'$ i chcemy wyznaczyć rozkład Cholesky'ego macierzy V, t.j. chcemy znaleźć macierz trójkątną dolną L taką, że $V = LL'$. Macierz L wyznaczamy jako

$$L = \begin{bmatrix} L_1 & 0 \\ l_1' & l_2 \end{bmatrix},$$

gdzie wektor l_1 znajdujemy rozwiązując układ $L_1 l_1 = v$ ($M+1$ operacji, bo L_1 jest dolnotrójkątna), natomiast wartość l_2 wyznaczamy ze wzoru

$$l_2 = \sqrt{v_2 - ||l_1||^2}.$$

Zastosowanie powyższych operacji powoduje, że kosztowny obliczeniowo rozkład $V_1 = L_1 L_1'$ wykonujemy tylko raz, a następnie korzystając z powyższych formuł wyznaczamy $V = LL'$, dla różnych wartości t. Implementacja tego kroku znajduje się w funkcji `chol_update`.

3.5 Aktualizacja RSS

W powyższych paragrafach pokazaliśmy jak, w sposób efektywny, rozwiązać układ $Va = c$ i wyznaczyć a, dla różnych wartości t. Poniżej pokażemy jak obliczyć RSS (potrzebne w każdym kroku Algorytmu 1) dla różnych t. Niech $\hat{a}$ będzie rozwiązaniem $Va = c$. Możemy zapisać:

$$RSS(l, v, t, \hat{a}) = \boldsymbol{y}' \tilde{B} \hat{a},$$

gdzie

$$\tilde{B} = [B_1, \ldots, B_M, B_l \cdot x_v, B_l \cdot (x_v - t)_+].$$

Zauważmy, że tylko ostatnia współrzędna wektora $\boldsymbol{y}'\tilde{B}$ zależy od t i wobec tego tylko ta ostatnia współrzędna wymaga aktualizacji dla każdego t. Pozostałe współrzędne $\boldsymbol{y}'\tilde{B}$ wystarczy obliczyć tylko raz.

3.6 Wybór "rodziców" w procedurze MARS

Zauważmy, że w kroku FORWARD, w każdej kolejnej pętli musimy sprawdzać coraz większy zbiór "rodziców" B. To powoduje, że kolejne pętle są coraz wolniejsze. Aby rozwiązać ten problem, stosuje się następujące rozwiązanie.

- Jeżeli $I < K + 2$ (gdzie K jest parametrem), to w zewnętrznej pętli Algorytmu 1 sprawdzamy wszystkich M rodziców.
- Jeżeli $I \geqslant K + 2$ sprawdzamy tylko K "najlepszych" rodziców.

Ranking "najlepszych" rodziców jest ustalany w następujący sposób. W iteracji numer I stosujemy następującą procedurę:

- Dla K najlepszych rodziców (w iteracji $I = K + 2$ są to wszyscy rodzice), aktualizujemy RSS oraz miarę poprawy dopasowania $IOF = -RSS$.
- Dla pozostałych rodziców, przepisujemy IOF z poprzednich iteracji.
- Wyliczamy rangi $R(l)$ dla wszystkich rodziców $l = 1, \ldots, M$ na podstawie IOF. Ranga M zostanie przypisana temu rodzicowi dla którego mamy największą wartość IOF.
- Dwie, dodane zmienne (potomkowie) dostają rangi: $R(M+1) = M+1$, $R(M+2) = M+2$.
- Dla wszystkich zmiennych $l = 1, 2, \ldots, M + 2$ obliczamy:

  $$P(l) := R(l) + \beta(I - I_{last}(l)),$$

  gdzie $\beta > 0$ jest parametrem, natomiast $I_{last}(l)$ to numer iteracji w której zmienna l została ostatni raz zaktualizowana. Parametr β jest nazywany współczynnikiem starzenia.
- W kolejnej iteracji $(I+1)$, K najlepszych rodziców zostaje wybranych na podstawie miary $P(l)$.

3.7 Wybór węzłów

W powyższych rozdziałach zostało opisane w jaki sposób wykonać wewnętrzną pętlę w Algorytmie 1, bez konieczności dopasowania modelu liniowego oddzielnie, dla każdego t. Dodatkowe przyspieszenie można osiągnąć wybierając kandydatów na węzły t. Wybór węzłów jest też istotny kiedy dane są "zaszumione". Wówczas, wybór wszystkich unikalnych wartości zmiennej x_v jako węzły, może prowadzić do przeuczenia. Wybieramy t jedynie ze zbioru $\{x_{v,i} : B_{l,i} > 0\}$ (wybierając t z poza tego zbioru, funkcja potomna $B_l(t - x_v)_+$ lub $B_l(x_v - t)_+$ będzie równa 0, dla wszystkich obserwacji). W dalszej części tego rozdziału x_v będzie oznaczać zmienną x_v, obciętą do zbioru $B_{l,i} > 0$. Przez N_l oznaczmy liczbę obserwacji dla których $B_{l,i} > 0$. Zauważmy, że nie ma sensu wybierać jako węzeł, największej wartości zmiennej x_v, ponieważ wówczas zmienna $B_l(x_v - t)_+$ będzie stale równa 0.

Niech $\tilde{x}_v$ oznacza unikalne wartości zmiennej x_v. Niech $\tilde{n}$ oznacza długość $\tilde{x}_v$ (czyli liczbę unikalnych wartości x_v). Poniżej przedstawiamy 3 możliwe podejścia:

1. Za węzły bierzemy wszystkie wartości $\tilde{x}_v$ (oprócz największej wartości $\tilde{x}_v$). Jak zostało wspomniane, takie podejście może prowadzić do przeuczenia.

2. Metoda Friedmana (zobacz: równania (43) i (45) w pracy [1]). Wprawadźmy dwie zmienne:

$$l = -\log_2 \left[-\frac{1}{pN_l} \log(1-\alpha) \right] /2.5,$$

oraz

$$le = 3 - \log_2(\alpha/p),$$

gdzie zwykle przyjmujemy $\alpha = 0.05$. Pierwszy parametr l oznacza jaka powinna być minimalna liczba obserwacji między węzłami. Drugi parametr le odnosi się do wymaganej liczby obserwacji między minimalną (maksymalną) wartością x_v a pierwszym (ostatnim) węzłem.

Uzasadnienie użycia pierwszego wzoru jest następujące. Chcemy uniknąć sytuacji w której przypadkowy układ błędów o takich samych znakach powoduje dodanie zawiasu w niewłaściwym miejscu. Oczekiwana liczba układów błędów o takim samym znaku i długość l^*, przy założeniu symetrii rozkładu błędów losowych, wynosi $\alpha = pN_l 2^{-l^*}$. Stosując przybliżenie $\alpha \approx -\log(1-\alpha)$, otrzymamy, że l^* jest tożsame z licznikiem we wzorze na l.

Stosujemy następującą procedurę:

- Jeżeli $\tilde{n} > 2le$, to odrzucamy le największych obserwacji i le najmniejszych obserwacji spośród $\tilde{x}_v$. Pozostałe wartości są kandydatami na węzły. Jeżeli $\tilde{n} \leqslant 2le$, bierzemy środkową wartość $\tilde{x}_v$ jako jedyny węzeł (dla parzystej liczności $\tilde{x}_v$, bierzemy mniejszą ze środkowych wartości). Niech $\bar{x}_v$ oznacza te wartości $\tilde{x}_v$, które zostały po pierwszym kroku i niech $\bar{n}$ oznacza liczność $\bar{x}_v$.

- Jeżeli $\bar{n} > 1$, to wykonujemy następującą procedurę: jeżeli $l + 2 > \bar{n}$, to za węzeł bierzemy najmniejszą wartość $\bar{x}_v$. Jeżeli $l + 2 \leqslant \bar{n}$, to za węzły bierzemy posortowane w porządku rosnącym wartości $\bar{x}_v$ o współrzędnych: $1, 1+(l+1), 1+2(l+1), \ldots$. Wybór węzłów opisany w tym punkcie jest zaimplementowany w funkcji `select_knots1`.

3. Metoda heurystyczna: Jeżeli $\tilde{n} <= 10$, to za węzły bierzemy wszystkie wartości $\tilde{x}_v$. Jeżeli $\tilde{n} > 10$, to liczbę węzłów ustalamy na $lk := \lfloor \sqrt{\tilde{n}} + 7 \rfloor$ i bierzymy lk równo rozmieszczonych wartości $\tilde{x}_v$.

Bez względu na to, którą z powyższych metod użyjemy, osobno traktujemy zmienne binarne. Dla zmiennych binarnych, rozważamy tylko jeden węzeł, równy mniejszej wartości tej zmiennej.

4 Uwagi

Poniżej przedstawiamy dodatkowe uwagi:

1. Macierz X powinna zawierać tylko zmienne ilościowe i nie może zawierać braków danych.

2. Przed wykonaniem kroku FORWARD, kolumny macierzy X powinny zostać centrowane i standaryzowane. Ten krok jest potrzebny aby uniknąc problemów numerycznych z rozkładem Cholesky'ego. Wprowadzenie kary $V + \epsilon I$ (zobacz sekcja 3.1) ma sens tylko wtedy gdy wszystkie zmienne są na tej samej skali.

3. Zauważmy, że pętle przebiegające różne wartości v oraz l mogą zostać zrównoleglone.

4. W opisywanej implementacji wprowadzamy również parametr określający maksymalny stopień interakcji d. W każdej iteracji przypisujemy dodawanym zmiennym (potomkom): stopień rodzica+1. W kolejnych iteracjach, nie używamy rodziców, którym w poprzednich iteracjach przypisano stopień równy d. Pierwszy rodzic (odpowiadający za wyraz wolny) ma zawsze stopień równy 0, więc może być wykorzystany w każdej iteracji. Zauważmy, że $d = 1$ odpowiada modelowi bez interakcji.
5. W opisywanej implementacji przyjmujemy następujące wartości domyślne parametrów:
   - Liczba najlepszych rodziców $K = 5$
   - Parametr starzenia $\beta = 1$
   - Maksymalna liczba iteracji $n_k = 15$.
   - Maksymalny rząd interakcji $d = 2$.
   - Kara w GCV $pen = 3$.
6. Przyspieszenie działania algorytmu można uzyskać na kilka sposobów:
   - Zmniejszenie n_k
   - Zmniejszenie d
   - Zwiększenie K
   - Wybór odpowiedniej strategii doboru węzłów (metody 2,3 są szybsze niż metoda 1).

5 Opis funkcji w R

Implementacja w R składa się z następujących funkcji:

1. `mars_forward(y,X,nk,fast.beta,fast.k,degree,selkn)`.

   Argumenty:

   - `y`- zmienna odpowiedzi
   - `X`- macierz zmiennych objaśniających
   - `nk`- maksymalna liczba iteracji
   - `fast.k`- liczba najlepszych rodziców K
   - `fast.beta`- parametr starzenia β
   - `degree`- maksymalny rząd interakcji d
   - `selkn`- metoda wyboru węzłów (domyślnie "NO"- metoda 1, "FR"- metoda 2, "HR"= metoda 3)

   Funkcja wykonuje krok FORWARD i zwraca macierz przekształconych zmiennych `B` i macierz `trans` zawierającą informację o przekształceniach zmiennych.

2. `mars_backward(y,B,type,penalty)`.

   Argumenty:

   - `y`- zmienna odpowiedzi
   - `B`- macierz przekształconych zmiennych z funkcji `mars_forward`.

- `type`- rodzaj funkcji kryterialnej (domyślnie "GCV"- metoda uogólnionej kroswalidacji, "BIC"- kryterium Bayesa, "AIC"- kryterium Akaike)
- `penalty`- kara *pen* w GCV.

Funkcja wykonuje krok BACKWARD i zwraca zbiór istotnych zmiennych `act`, spośród kolumn macierzy `B`.

3. `mars_transform(X,trans)`.

   Argumenty:

   - `X`- macierz nowych danych
   - `trans`- macierz `trans` zawierającą informację o przekształceniach zmiennych z funkcji `mars_forward`.

   Funkcja dokonuje transformacji zmiennych znalezionych przez MARS, dla nowej macierzy danych `X`.

Literatura

[1] J. H. Friedman. Multivariate adaptive regression splines. *The Annals of Statistics*, 19(1):1–141, 1991.

[2] J. H. Friedman. Fast mars, 1993. Technical Report.