# Modele addytywne oraz metody PRIM i MARS

Patrycja Hęćka

# Spis treści

# Uogólnione modele addytywne

- Uogólniony model addytywny ma formę

$$\mathrm{E}(Y|X_1, X_2, \ldots, X_p) = \alpha + f_1(X_1) + f_2(X_2) + \cdots + f_p(X_p).$$

- $X_1, \ldots, X_p$ - predyktory, Y – zmienna odpowiedzi
- $f_j$ – nieparametryczne funkcje

# Dwu-klasowa klasyfikacja

- μ(X) = Pr(Y = 1|X) – średnia odpowiedzi binarnej

$$\log\left(\frac{\mu(X)}{1-\mu(X)}\right) = \alpha + \beta_1 X_1 + \cdots + \beta_p X_p.$$

- Addytywny logistyczny model regresji zamienia każdy składnik liniowy w ogólniejszą formę funkcjonalną:

$$\log\left(\frac{\mu(X)}{1-\mu(X)}\right) = \alpha + f_1(X_1) + \cdots + f_p(X_p)$$

- g– funkcja linkująca

$$g[\mu(X)] = \alpha + f_1(X_1) + \cdots + f_p(X_p).$$

Przykłady funkcji linkujących:

- g(μ) = μ,
- g(μ) = logit(μ), g(μ) = probit(μ),
- g(μ) = log(μ),

- g(μ) = $X^T\beta + \alpha_k + f(Z)$ - model semiparametryczny

- Addytywna dekompozycja szeregu czasowego

$$Y_t = S_t + T_t + \epsilon_t,$$

$S_t -$ składowe sezonowe,

$T_t$ – trend,

$\epsilon -$ błąd pomiaru.

# Dopasowywanie modelu addytywnego

- Model addytywny ma formę:

$$Y = \alpha + \sum_{j=1}^{p} f_j(X_j) + \varepsilon,$$

gdzie $\epsilon$ ma średnią zero.

Korzystamy z kryterium sumy kwadratów z pewną karą:

$$\mathrm{PRSS}(\alpha, f_1, f_2, \ldots, f_p) = \sum_{i=1}^{N} \left( y_i - \alpha - \sum_{j=1}^{p} f_j(x_{ij}) \right)^2 + \sum_{j=1}^{p} \lambda_j \int f_j''(t_j)^2 dt_j$$

**Algorithm 9.1** *The Backfitting Algorithm for Additive Models.*

1. Initialize: $\hat{\alpha} = \frac{1}{N}\sum_1^N y_i$, $\hat{f}_j \equiv 0, \forall i, j$.

2. Cycle: $j = 1, 2, \ldots, p, \ldots, 1, 2, \ldots, p, \ldots,$

$$\hat{f}_j \leftarrow \mathcal{S}_j\left[\{y_i - \hat{\alpha} - \sum_{k \neq j} \hat{f}_k(x_{ik})\}_1^N\right],$$

$$\hat{f}_j \leftarrow \hat{f}_j - \frac{1}{N}\sum_{i=1}^N \hat{f}_j(x_{ij}).$$

until the functions $\hat{f}_j$ change less than a prespecified threshold.

Ten sam algorytm daje inne dopasowania, jeśli zastąpimy operator $S_j$

poprzez:

- Inne jednowymiarowe „wygładzacze” regresji, takie jak lokalne regresje wielomianowe i metody kernel,
- Operatory regresji liniowej dające dopasowania wielomianowe, odcinkowo stałe, parametryczne, sklejane dopasowania, szeregowe albo Fouriera,
- Bardziej skomplikowane operatory takie jak wygładzające powierzchnię dla  interakcji drugiego lub wyższego rzędu, bądź dla okresowych wygładzeń efektów sezonowych.

- Dla modelu regresji logistycznej i innych uogólnionych modeli liniowych właściwym kryterium jest kryterium logarytmu największej wiarogodności z pewną karą.

# Addytywna regresja logistyczna

- Modelujemy Pr(Y=1|X),
- $X^T = (X_1, \ldots, X_p)$,
- Uogólniony, logistyczny model addytywny jest postaci

$$\log \frac{\Pr(Y = 1|X)}{\Pr(Y = 0|X)} = \alpha + f_1(X_1) + \cdots + f_p(X_p).$$

- Funkcje $f_1, \ldots, f_p$ estymowane przez procedurę backfitting połączoną z procedurą Newtona-Raphsona.

**Algorithm 9.2** *Local Scoring Algorithm for the Additive Logistic Regression Model.*

1. Compute starting values: $\hat{\alpha} = \log[\bar{y}/(1-\bar{y})]$, where $\bar{y} = \text{ave}(y_i)$, the sample proportion of ones, and set $\hat{f}_j \equiv 0 \; \forall j$.

2. Define $\hat{\eta}_i = \hat{\alpha} + \sum_j \hat{f}_j(x_{ij})$ and $\hat{p}_i = 1/[1+\exp(-\hat{\eta}_i)]$.

   Iterate:

   (a) Construct the working target variable

   $$z_i = \hat{\eta}_i + \frac{(y_i - \hat{p}_i)}{\hat{p}_i(1-\hat{p}_i)}.$$

   (b) Construct weights $w_i = \hat{p}_i(1-\hat{p}_i)$

   (c) Fit an additive model to the targets $z_i$ with weights $w_i$, using a weighted backfitting algorithm. This gives new estimates $\hat{\alpha}, \hat{f}_j, \; \forall j$

3. Continue step 2. until the change in the functions falls below a prespecified threshold.

# Przykład: przewidywanie spamu

- 4601 wiadomości e-mail,
- Binarna zmienna odpowiedzi: email lub spam,
- 57 predyktorów:

- 48 predyktorów ilościowych – procent słów w e-mailu, które zawierają podane słowo np. „business”, „address”, „internet”, „free”, „george”;

- 6 predyktorów ilościowych – procent pojawiających się znaków w e-mailu postaci: ch;, ch(, ch[, ch!, ch$, lub ch#.

- średnia długość nieprzerwanych ciągów wielkich liter,
- długość nieprzerwanego najdłuższego ciągu wielkich liter,
- suma długości nieprzerwanych ciągów wielkich liter.

- spam = 1, email = 0,
- losowo wybrana próba testowa rozmiaru 1536,
- próba treningowa rozmiaru 3065,
- Użyto uogólnionego modelu addytywnego z sześcienną funkcją sklejaną z 4 stopniami swobody dla każdego predyktora.

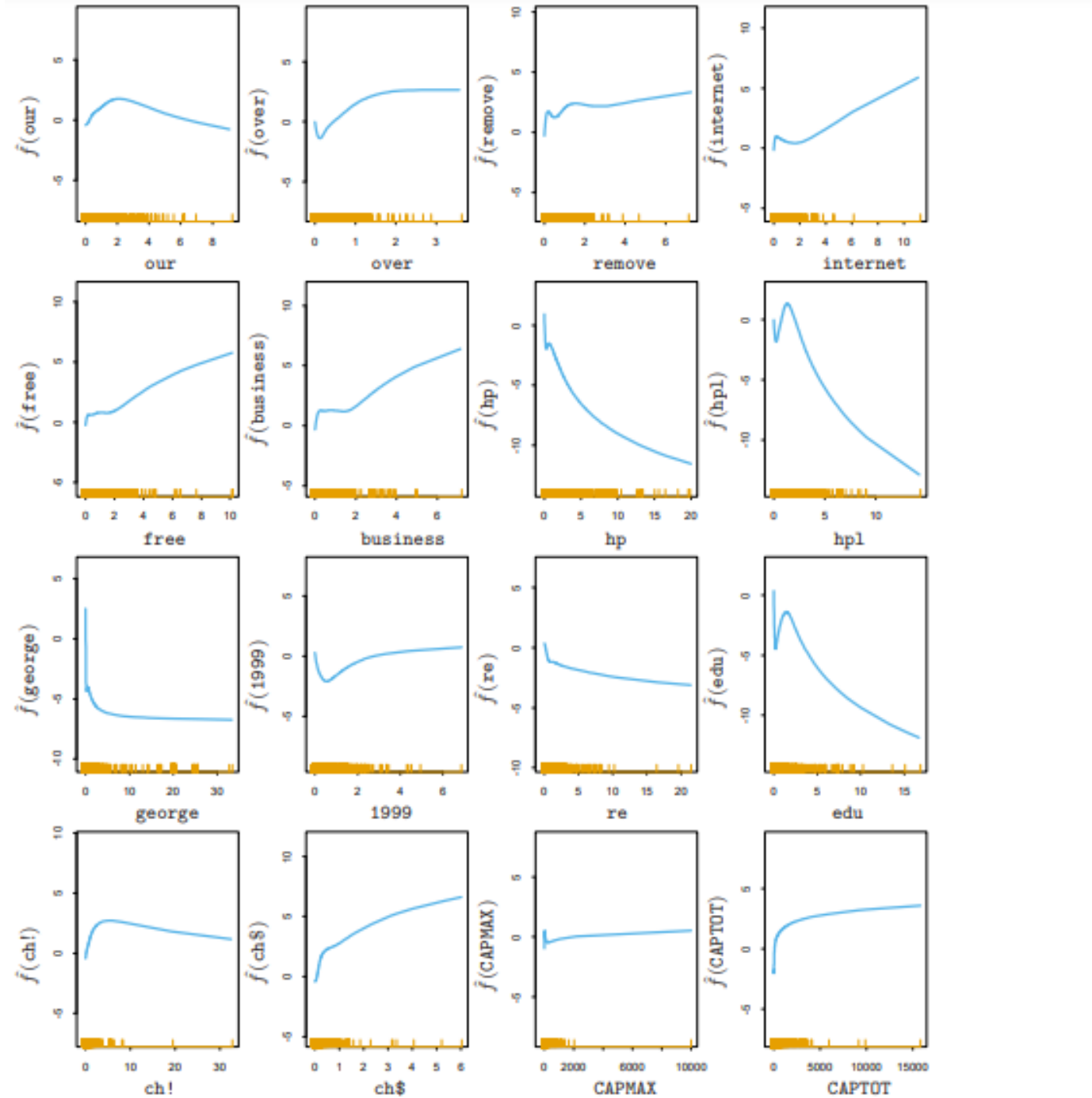
Tzn. że dla każdego predyktora $X_j$ parametr wygładzenia $\lambda_j$ został wybrany tak, aby ślad$\left[S_j\left(\lambda_j\right)\right] - 1 = 4$, gdzie $S_j(\lambda)$ jest macierzą wygładzania skonstruowaną z obserwowanych wartości $x_{ij}$.

- Nałożono na zmienne logarytm log(x+0.1).

| True Class | Predicted Class | |
|---|---|---|
| | email (0) | spam (1) |
| email (0) | 58.3% | 2.5% |
| spam (1) | 3.0% | 36.3% |

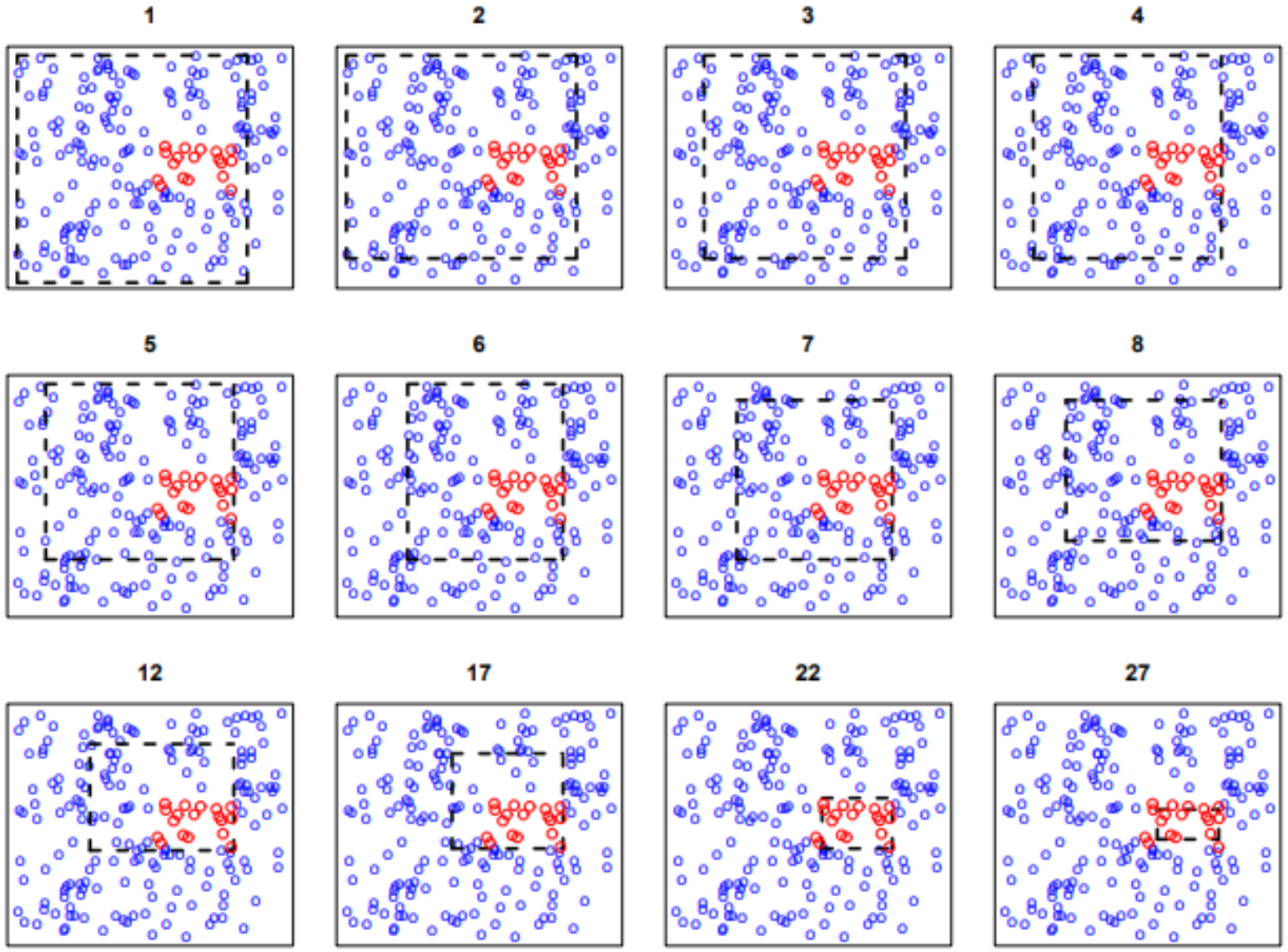
Błąd testowy wynosi 5.5%.
Dla porównania liniowy model regresji logistycznej miał błąd testowy 7.6%.

| Name | Num. | df | Coefficient | Std. Error | $Z$ Score | Nonlinear $P$-value |
|---|---|---|---|---|---|---|
| *Positive effects* | | | | | | |
| our | 5 | 3.9 | 0.566 | 0.114 | 4.970 | 0.052 |
| over | 6 | 3.9 | 0.244 | 0.195 | 1.249 | 0.004 |
| remove | 7 | 4.0 | 0.949 | 0.183 | 5.201 | 0.093 |
| internet | 8 | 4.0 | 0.524 | 0.176 | 2.974 | 0.028 |
| free | 16 | 3.9 | 0.507 | 0.127 | 4.010 | 0.065 |
| business | 17 | 3.8 | 0.779 | 0.186 | 4.179 | 0.194 |
| hpl | 26 | 3.8 | 0.045 | 0.250 | 0.181 | 0.002 |
| ch! | 52 | 4.0 | 0.674 | 0.128 | 5.283 | 0.164 |
| ch$ | 53 | 3.9 | 1.419 | 0.280 | 5.062 | 0.354 |
| CAPMAX | 56 | 3.8 | 0.247 | 0.228 | 1.080 | 0.000 |
| CAPTOT | 57 | 4.0 | 0.755 | 0.165 | 4.566 | 0.063 |
| *Negative effects* | | | | | | |
| hp | 25 | 3.9 | −1.404 | 0.224 | −6.262 | 0.140 |
| george | 27 | 3.7 | −5.003 | 0.744 | −6.722 | 0.045 |
| 1999 | 37 | 3.8 | −0.672 | 0.191 | −3.512 | 0.011 |
| re | 45 | 3.9 | −0.620 | 0.133 | −4.649 | 0.597 |
| edu | 46 | 4.0 | −1.183 | 0.209 | −5.647 | 0.000 |


$\hat{f}$(our)
our
$\hat{f}$(over)
over
$\hat{f}$(remove)
remove
$\hat{f}$(internet)
internet
$\hat{f}$(free)
free
$\hat{f}$(business)
business
$\hat{f}$(hp)
hp
$\hat{f}$(hpl)
hpl
$\hat{f}$(george)
george
$\hat{f}$(1999)
1999
$\hat{f}$(re)
re
$\hat{f}$(edu)
edu
$\hat{f}$(ch!)
ch!
$\hat{f}$(ch$)
ch$
$\hat{f}$(CAPMAX)
CAPMAX
$\hat{f}$(CAPTOT)
CAPTOT

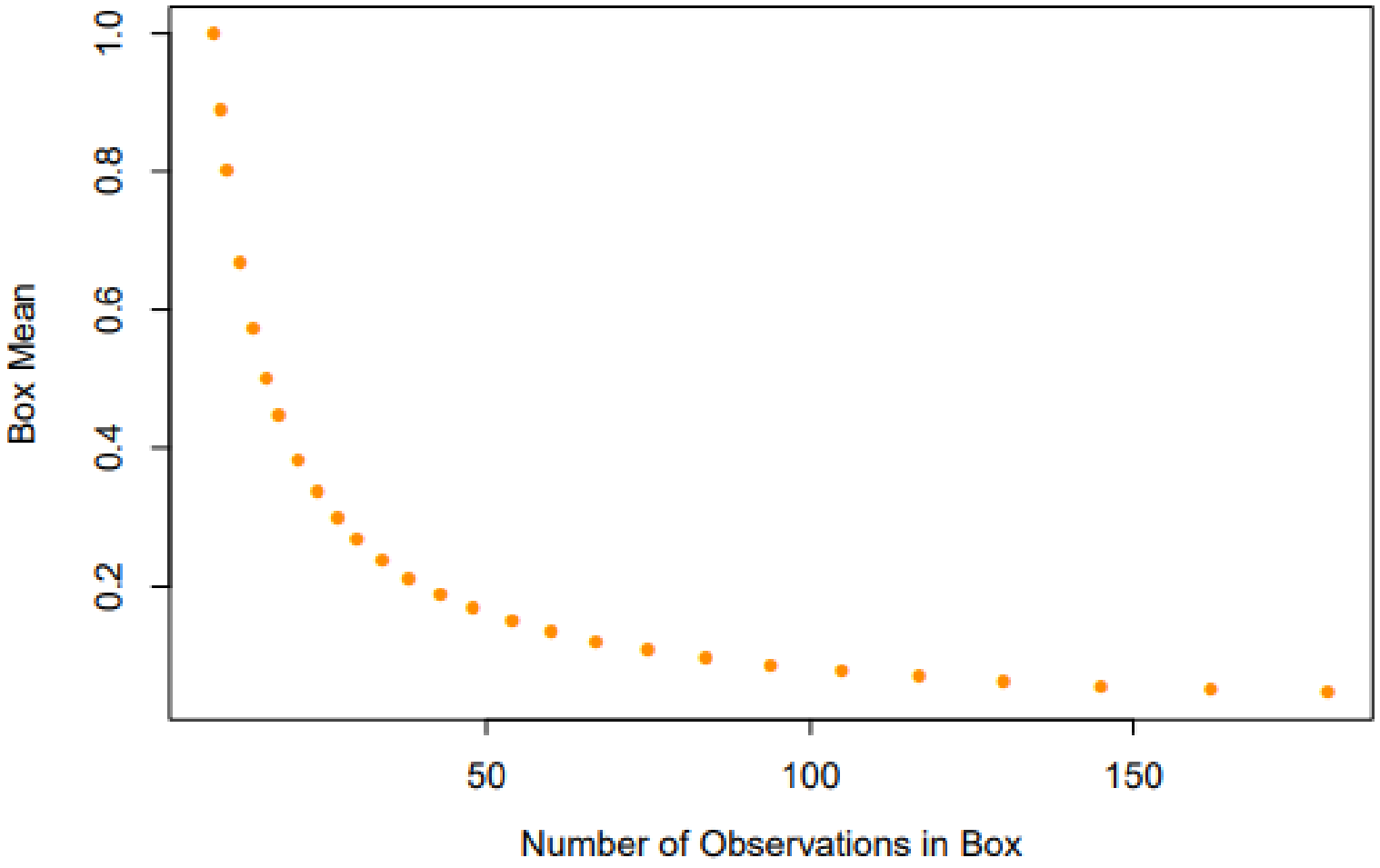
# PRIM: Bump Hunting

- Metoda cierpliwej indukcji reguł (patient rule induction method) znajduje pudełka w przestrzeni cech, szukając takich, dla których średnia odpowiedzi jest wysoka.
- Szuka maksimów w funkcji celu. Jeśli pożądane są minima, pracuje się z negatywnymi wartościami zmiennej odpowiedzi.




200 jednostajnie rozłożonych punktów

Zmienna odpowiedzi przyjmuje 1 (czerwień), jeśli $0.5 < X_1 < 0.8$ i $0.4 < X_2 < 0.6$.


Box Mean
0.2
0.4
0.6
0.8
1.0
50
100
150
Number of Observations in Box

- Po zajściu sekwencji zstępującej, PRIM odwraca proces, rozszerzając się wzdłuż dowolnej krawędzi, jeśli takie rozszerzenie zwiększy średnią w pudełku.
- Odgórna procedura jest zachłanna na każdym kroku, więc często dochodzi do odwracania procesu.
- Wynik jest sekwencją pudełek z różnymi liczbami obserwacji. Kroswalidacja, połączona z oceną analityka, jest stosowana do wyboru optymalnego rozmiaru pudełka.

- $B_1$ - indeksy obserwacji znalezionych w kroku 1,
- Procedura PRIM usuwa obserwacje umieszczone w $B_1$ ze zbioru treningowego. Następnie dwuetapowy proces – usuwanie i późniejsze dołączanie zmiennych jest powtarzane na pozostałym zbiorze danych.
- Ten proces jest powtarzany kilka razy, produkując sekwencję pudełek $B_1, B_2, \dots, B_k$. Każde pudełko jest definiowane poprzez zestaw reguł obejmujących podzbiór predyktorów takich jak:

$$(a_1 \leq X_1 \leq b_1) \quad i \quad (b_1 \leq X_3 \leq b_2)$$

Algorytm PRIM

1. Startujemy z pełnym zbiorem danych treningowych i maksymalnym pudełkiem, zawierającym wszystkie dane.
2. Zmniejszamy pudełko, usuwając procent obserwacji (zwykle 0.05 lub 0.1) mający najwyższe (lub najniższe) wartości predyktora $X_j$.
3. Powtarzamy krok 2 dopóki w pudełku nie zostanie minimalna liczba obserwacji (np. 10).
4. Powiększamy pudełko, jeśli to wpłynie na wzrost średniej.
5. Otrzymujemy sekwencję pudełek z różną liczbą obserwacji. Kroswalidacja pomoże wybrać optymalne pudełko nazwane $B_1$.
6. Usuwamy dane z pudełka $B_1$ ze zbioru danych i powtarzamy kroki 2-5, aby uzyskać drugie pudełko i kontynuujemy dopóki nie otrzymamy wystarczającej liczby pudełek.

# Przykład: dane ze spamem

- Spam = 1, email = 0

Pierwsze dwa pudełka znalezione przez PRIM:

| Rule 1 | Global Mean | Box Mean | Box Support |
|---|---|---|---|
| Training | 0.3931 | 0.9607 | 0.1413 |
| Test | 0.3958 | 1.0000 | 0.1536 |

$$\text{Rule 1} \begin{cases} \texttt{ch!} & > & 0.029 \\ \texttt{CAPAVE} & > & 2.331 \\ \texttt{your} & > & 0.705 \\ \texttt{1999} & < & 0.040 \\ \texttt{CAPTOT} & > & 79.50 \\ \texttt{edu} & < & 0.070 \\ \texttt{re} & < & 0.535 \\ \texttt{ch;} & < & 0.030 \end{cases}$$

Box support – proporcja obserwacji, które są w pudełku

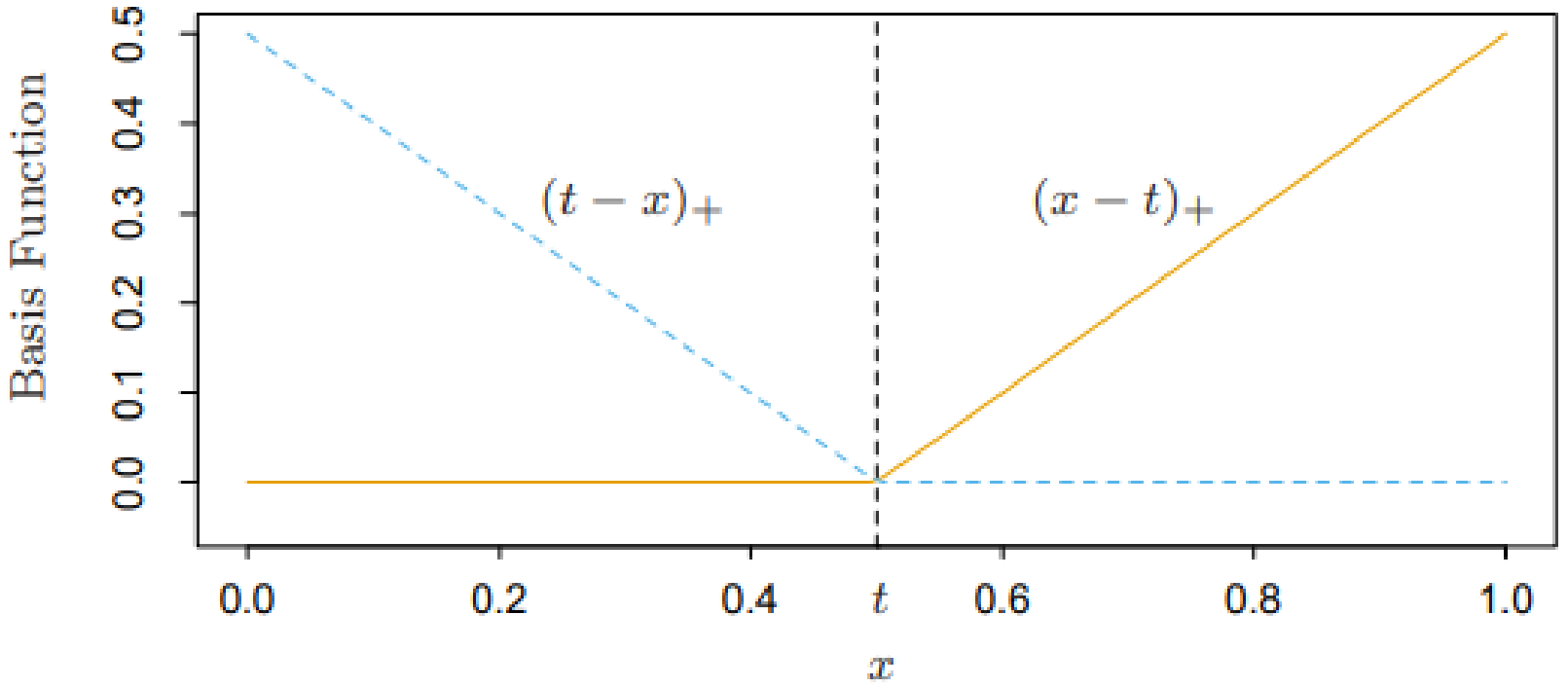
| Rule 2 | Remain Mean | Box Mean | Box Support |
|---|---|---|---|
| Training | 0.2998 | 0.9560 | 0.1043 |
| Test | 0.2862 | 0.9264 | 0.1061 |

$$\text{Rule 2} \begin{cases} \texttt{remove} & > & 0.010 \\ \texttt{george} & < & 0.110 \end{cases}$$

# MARS: Multivariate Adaptive Regression Splines

- Metoda MARS  rozszerza kawałkami liniowe funkcje bazowe postaci:

$$(x-t)_+ = \begin{cases} x-t, & \text{if } x > t, \\ 0, & \text{otherwise,} \end{cases} \quad \text{and} \quad (t-x)_+ = \begin{cases} t-x, & \text{if } x < t, \\ 0, & \text{otherwise.} \end{cases}$$




t=0.5

- Każda funkcja jest kawałkowo liniowa z węzłem w wartości t. Nazywamy te dwie funkcje „reflected pair”.
- Celem jest utworzyć „reflected pair” dla każdego wejścia $X_j$ z węzłami przy każdej zaobserwowanej wartości $x_{ij}$ tego wejścia.
- Wtedy kolekcja funkcji bazowych jest postaci

$$\mathcal{C} = \{(X_j - t)_+,\ (t - X_j)_+\}_{\substack{t \in \{x_{1j}, x_{2j}, \ldots, x_{Nj}\} \\ j = 1, 2, \ldots, p.}}$$

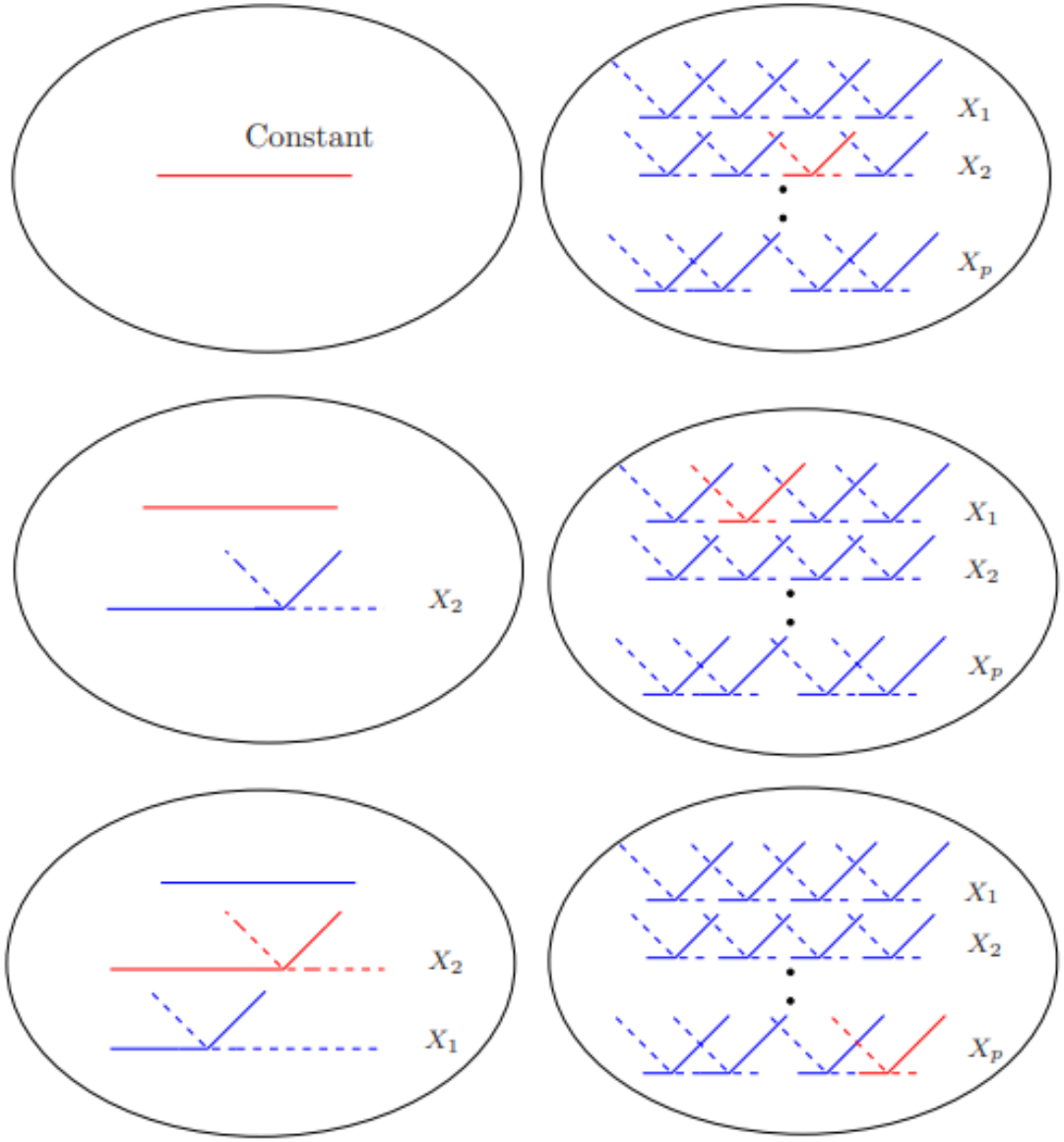
- Model ma formę

$$f(X) = \beta_0 + \sum_{m=1}^{M} \beta_m h_m(X),$$

gdzie $h_m$(X) jest funkcją ze zbioru C albo produktem dwóch lub więcej funkcji.

Współczynniki $\beta_m$ są estymowane przez minimalizowanie rezydualnej sumy kwadratów.

- Zaczynamy z $h_0(X) = 1$, a pozostałe funkcje ze zbioru C są funkcjami kandydującymi.



- Na każdym etapie rozważamy jako nową parę funkcji bazowych wszystkie produkty funkcji $h_m$ w zbiorze M z jedną parą „reflected pairs” ze zbioru C.
- Dodajemy do modelu M składnik

$$\hat{\beta}_{M+1} h_\ell(X) \cdot (X_j - t)_+ + \hat{\beta}_{M+2} h_\ell(X) \cdot (t - X_j)_+, \; h_\ell \in \mathcal{M},$$

który produkuje największy spadek błędu treningowego.

$\widehat{\beta_{M+1}}$ i $\widehat{\beta_{M+2}}$ są estymowane metodą najmniejszych kwadratów wraz ze wszystkimi innymi M+1 współczynnikami w modelu. Zwycięska para jest dodawana do modelu i proces jest kontynuowany, dopóki zbiór modelu M nie będzie zawierał maksymalnej liczby składników.

PRZYKŁAD

Załóżmy, że na początku chcemy dodać do modelu funkcję postaci

$\widehat{\beta_1}\ \ (X_j - x_{72})_+ + \widehat{\beta_2}\ \ (x_{72} - X_2)_+$

Wtedy ta para funkcji bazowych jest dodawana do zbioru M i następnym krokiem jest rozważenie par produktów postaci:

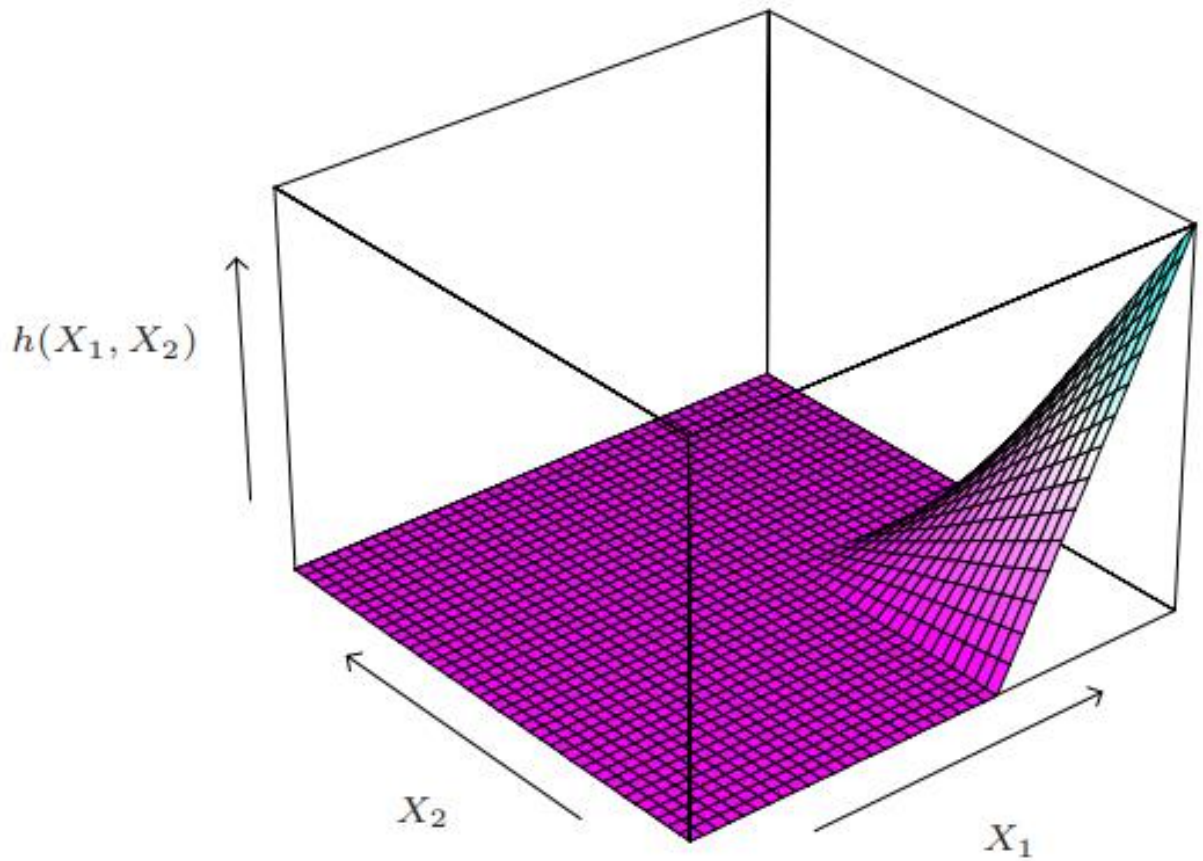
$h_m(X) * (X_j - t)_+$ oraz $h_m(X) * (t - X_j)_+$

gdzie jako $h_m$ mamy do wyboru:

$h_0(X) = 1, h_1(X) = (X_2 - x_{72})_+$

Albo $h_2(X) = (x_{72} - X_2)_+$.

Trzeci z tych wyborów produkuje funkcję postaci $(X_1 - x_{51})_+ * (x_{72} - X_2)_+$

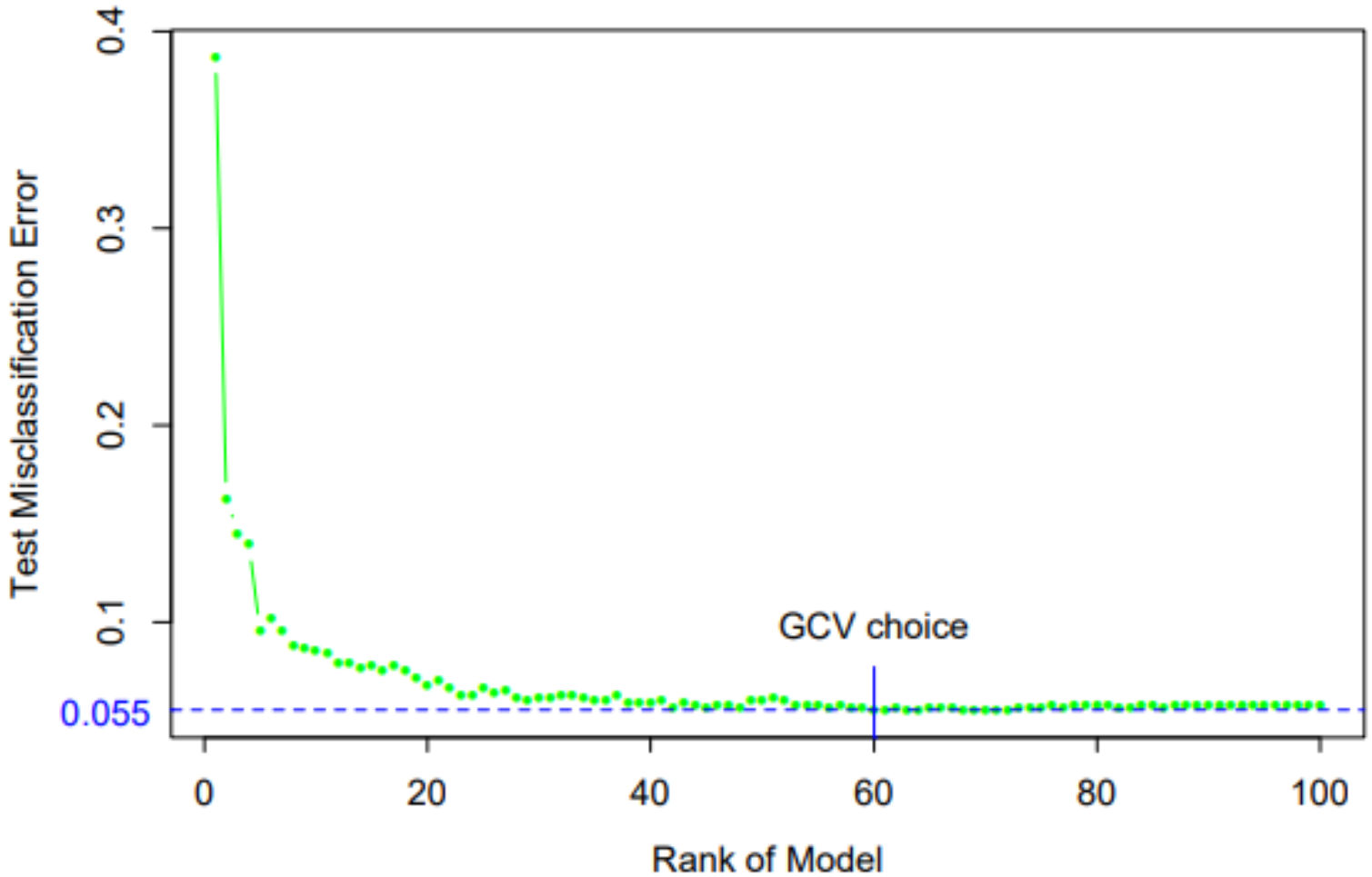

$h(X_1, X_2)$
$X_2$
$X_1$

- Uogólniona kroswalidacja

$$\mathrm{GCV}(\lambda) = \frac{\sum_{i=1}^{N}(y_i - \hat{f}_\lambda(x_i))^2}{(1 - M(\lambda)/N)^2}$$

M($\lambda$) – liczba parametrów w modelu: liczba składników w modelu + liczba parametrów wykorzystywana przy wyborze optymalnych pozycji węzłów;

M($\lambda$)=r+3K, r – liczba liniowo niezależnych funkcji bazowych w modelu, K – liczba węzłów.


Test Misclassification Error
0.4
0.3
0.2
0.1
0.055
GCV choice
0
20
40
60
80
100
Rank of Model

# Przykład

- N=100 obserwacji,
- predyktory $X_1, \ldots, X_p$, błędy $\epsilon$ - niezależne o rozkładzie standardowym normalnym.

Scenariusz 1.

$$Y = (X_1 - 1)_+ + (X_1 - 1)_+ \cdot (X_2 - .8)_+ + 0.12 \cdot \varepsilon.$$

Scenariusz 2

Taki sam jak 1, ale z p=20 predyktorami.

## Scenariusz 3

Struktura sieci neuronowej

$$
\begin{aligned}
\ell_1 &= X_1 + X_2 + X_3 + X_4 + X_5, \\
\ell_2 &= X_6 - X_7 + X_8 - X_9 + X_{10}, \\
\sigma(t) &= 1/(1 + e^{-t}), \\
Y &= \sigma(\ell_1) + \sigma(\ell_2) + 0.12 \cdot \varepsilon.
\end{aligned}
$$

$\mu(x)$ – prawdziwa średnia Y

$$\begin{aligned} \mathrm{MSE}_0 &= \mathrm{ave}_{x\in\mathrm{Test}}(\bar{y}-\mu(x))^2, \\ \mathrm{MSE} &= \mathrm{ave}_{x\in\mathrm{Test}}(\hat{f}(x)-\mu(x))^2. \end{aligned}$$

$$R^2 = \frac{\mathrm{MSE}_0 - \mathrm{MSE}}{\mathrm{MSE}_0}$$

| Scenario | Mean (S.E.) |
|---|---|
| 1: Tensor product $p = 2$ | 0.97 (0.01) |
| 2: Tensor product $p = 20$ | 0.96 (0.01) |
| 3: Neural network | 0.79 (0.01) |

# Literatura

Trevor Hastie, Robert Tibshirani, Jerome Friedman - The Elements of Statistical Learning Data Mining, Inference, and Prediction, Springer Series in Statistics, Second Edition, February 2009

DZIĘKUJĘ ZA UWAGĘ!