WYKŁAD VII: Metody estymacji funkcji regresji

MiNI PW, semestr letni 2013/2014

Niech $(\mathbf{X}, Y) \in R^{p+1}$ będzie wektorem losowym takim, że

$$Y = f(\mathbf{X}) + \varepsilon,$$

gdzie ε- błąd losowy taki, że $E(\varepsilon|\mathbf{X} = \mathbf{x}) = 0$ dla dowolnego $\mathbf{x}$, w szczególności $E(\varepsilon) = 0$. $f(\mathbf{x})$- funkcja regresji Y względem $\mathbf{X}$. Estymacja na podstawie próby losowej $(\mathbf{X}_1, Y_1), \ldots, (\mathbf{X}_n, Y_n)$. $\mathbf{X}_1, \ldots, \mathbf{X}_n$- wielkości losowe, lub ciąg deterministyczny. W wykładzie o drzewach klasyfikacyjnych rozważana była krótko estymacja $f(\mathbf{x})$ metodą drzew regresyjnych.

Tu omówimy inne, podstawowe metody, estymacji funkcji regresji. Rozważymy:

- metody parametryczne estymacji funkcji regresji;
- metody nieparametryczne dla niskiego wymiaru $\mathbf{X}$ $(p = 1, 2)$;
- metody nieparametryczne dla dużego wymiaru p przy przyjęciu pewnych ogólnych założeń o postaci f.

Metody parametryczne estymacji funkcji regresji

Niech

$$Y_i = f(\mathbf{x}_i, \alpha) + \varepsilon_i, \qquad i = 1, \ldots, n, \qquad (\star)$$

gdzie $\varepsilon_1, \ldots, \varepsilon_n$ -niezależne zmienne losowe o tym samym rozkładzie, $E\varepsilon = 0$ i $\mathbf{x}_1, \ldots \mathbf{x}_n$- wielkości deterministyczne. Postać $f(\cdot, \alpha)$- znana, nieznany parametr $\alpha = (\alpha_1, \ldots, \alpha_q)$. Jeśli f nie jest funkcją afiniczną $\mathbf{x}$, to równanie $(\star)$ równanie regresji nieliniowej.
Podstawowa metoda estymacji α: Metoda Nieliniowych Najmniejszych Kwadratów (MNNK). Próba $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$.

$$\hat{\alpha} = \operatorname{argmin} \sum_{i=1}^{n} (y_i - f(\mathbf{x}_i, \alpha))^2 := Q(\alpha).$$

Równanie punktu stacjonarnego

$$\frac{\partial Q}{\partial \alpha_k} = \sum_{i=1}^{n} -2(y_i - f(\mathbf{x}_i, \alpha)) \frac{\partial f(\mathbf{x}_i, \alpha)}{\partial \alpha_k} = 0, \quad k = 1, \ldots, q$$

nie daje się z reguły rozwiązać explicite.

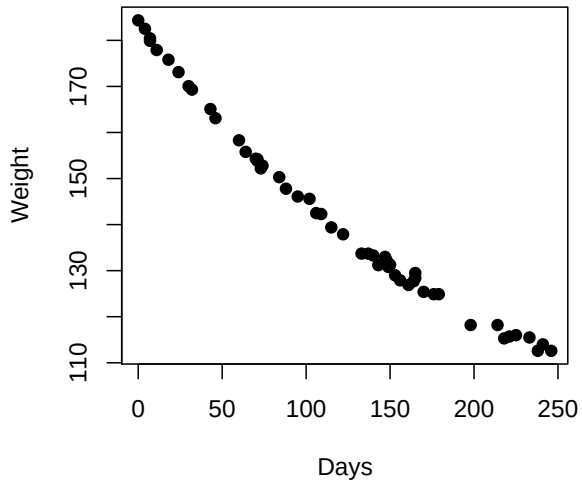
Iteracyjna metoda Gaussa-Newtona.
Próba $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$). $\alpha^{(0)} = (\alpha_1^{(0)}, \ldots, \alpha_q^{(0)})'$- początkowy estymator. Rozwiniecie Taylora daje ($\alpha = (\alpha_1, \ldots, \alpha_q)'$)

$$f(\mathbf{x}_i, \alpha) \approx f(\mathbf{x}_i, \alpha^{(0)}) + \sum_{k=1}^{q} \frac{\partial f(\mathbf{x}_i, \alpha)}{\partial \alpha_{k\,|\alpha=\alpha^{(0)}}} (\alpha_k - \alpha_k^{(0)}) \qquad i = 1, \ldots, n$$

Zastępujemy $f(\mathbf{x}_i, \alpha)$ przez y_i i otrzymujemy równanie regresji liniowej z parametrami $\eta_k = \alpha_k - \alpha_k^{(0)}$ i macierzą eksperymentu $(x_{ik})_{n\times q} := (\frac{\partial f(\mathbf{x}_i, \alpha)}{\partial \alpha_{k\,|\alpha=\alpha^{(0)}}})$. Po rozwiązaniu

$$\alpha_k^{(1)} = \alpha_k^{(0)} + \eta_k \qquad k = 1, \ldots, q$$

Przykład Dane dotyczą kontrolowanej utraty wagi przez pacjenta w wieku 48 lat, wzrostu 193 cm, waga początkowa 183.4 kg (zbiór `wtloss`, pakiet Mass).



Postulowana postać parametryczna

$$Weight = \beta_0 + \beta_1 2^{-Days/\theta} + \varepsilon,$$

β_0 -waga do osiągnięcia, β_1 -waga do stracenia, θ- czas potrzebny do stracenia $\beta_1/2$.

Ustalamy wartości początkowe parametrów na podstawie ich interpretacji.
Procedura realizująca MNNK: nls

```
wtloss.st=c(b0=90,b1=95,th=120)

wtloss.st<-nls(Weight~b0+b1*2^(-Days/th),data=wtloss,
start=wtloss.st,trace=T)

67.54349 :   90  95 120
40.18081 :   82.72629 101.30457 138.71374
39.24489 :   81.39868 102.65836 141.85859
39.2447 :   81.37375 102.68417 141.91052
```

Stabilizacja wartości po trzech iteracjach.

Estymowana waga docelowa - mniejsza niż początkowa wartość,
większa wartość parametru th

Nieparametryczne estymatory funkcji regresji (małe p)

Nie zakładamy nic o postaci funkcji regresji, poza założeniami o jej gładkości.

- estymator średniej ruchomej;
- estymatory lokalnie liniowe;
- funkcje sklejane (spline'y): na przykładzie naturalnego spline'u kubicznego.

Średnia ruchoma z parametrem $h = h_n$: Estymator $f(\mathbf{x}) = E(Y|\mathbf{X} = \mathbf{x})$
Średnia próbkowa wartości Y_i odpowiadających $\mathbf{x}_i$ takich, że $||\mathbf{x}_i - \mathbf{x}|| \leqslant h$.
h-parametr wygładzający tego estymatora: $h \uparrow$ - obciążenie $\uparrow$, $h \downarrow$ - wariancja $\uparrow$.

Estymator lokalnie liniowy

Rozwiązanie problemu ważonych najmniejszych kwadratów, dla którego wielkość wag jest, regulowana odległością wartości $\mathbf{x}_i$ od punktu $\mathbf{x}$. Dla ustalonego $\mathbf{x}$ minimalizowana jest funkcja

$$\sum_{i=1}^{n} K(\mathbf{x} - \mathbf{x}_i)/h_n)(Y_i - \beta_0(\mathbf{x}) - \beta_1(\mathbf{x})^T(\mathbf{x}_i - \mathbf{x}))^2$$

ze względu na β_0 i β_0. K: gęstość prawdopodobieństwa na R^p mająca maksimum w 0.

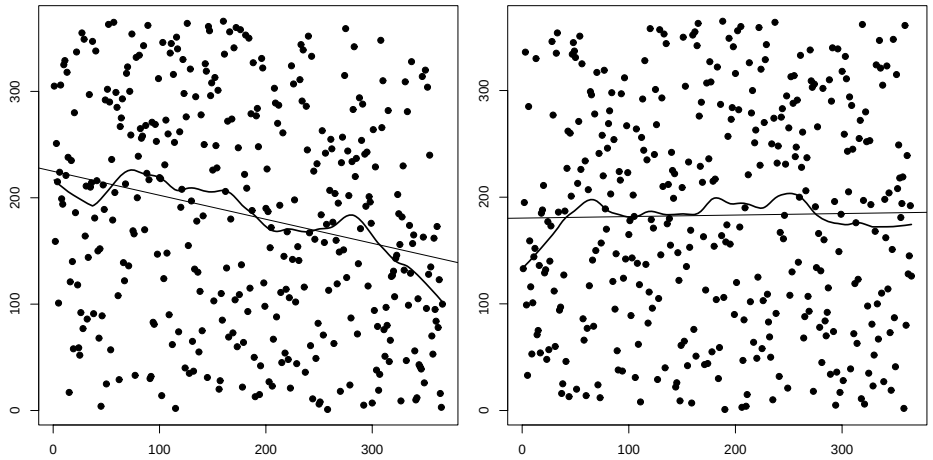
$$\hat{f}_{loc}(\mathbf{x}) = \hat{\beta}_0(\mathbf{x})$$

Funkcja realizująca w R estymator lokalnie wielomianowy jest funkcja loess. Dla wyboru parametru wygładzającego metoda kroswalidacji (walidacji krzyżowej)

$$h_{n,CV} = \operatorname{argmin}_h \sum_{i=1}^{n} (y_i - \hat{f}_{loc}^{-i}(x))^2, \tag{1}$$

-gdzie $\hat{f}_{loc}^{-i}$ jest estymatorem na podstawie próby z usuniętą obserwacją $(\mathbf{x}_i, y_i)$.

Przykład Zbiory draft70yr.dat i draft71yr.dat: dane dotyczące losowań kolejności powszechnego poboru do wojska w czasie wojny w Wietnamie w latach 1970 i 1971. Losowania dotyczyły kapsułek z 366 możliwymi datami urodzin i polegały na kolejnym ich losowaniu bez zwracania. Metoda poboru: urodzeni w latach 1944–1950 o pierwszej wylosowanej dacie urodzin byli wcielani w pierwszej kolejności, o drugiej wylosowanej dacie urodzin jako następni itd. Diagramy rozproszenia dla kolejnych lat (oś x – numer daty urodzenia od 1 do 366, oś y – numer losowania, w którym data została wylosowana). Do wykresów dopasowano **estymator lokalnie liniowy i estymator MNK**.



Rysunek: Diagramy rozproszenia dla lat 1970, 1971

Funkcje sklejane

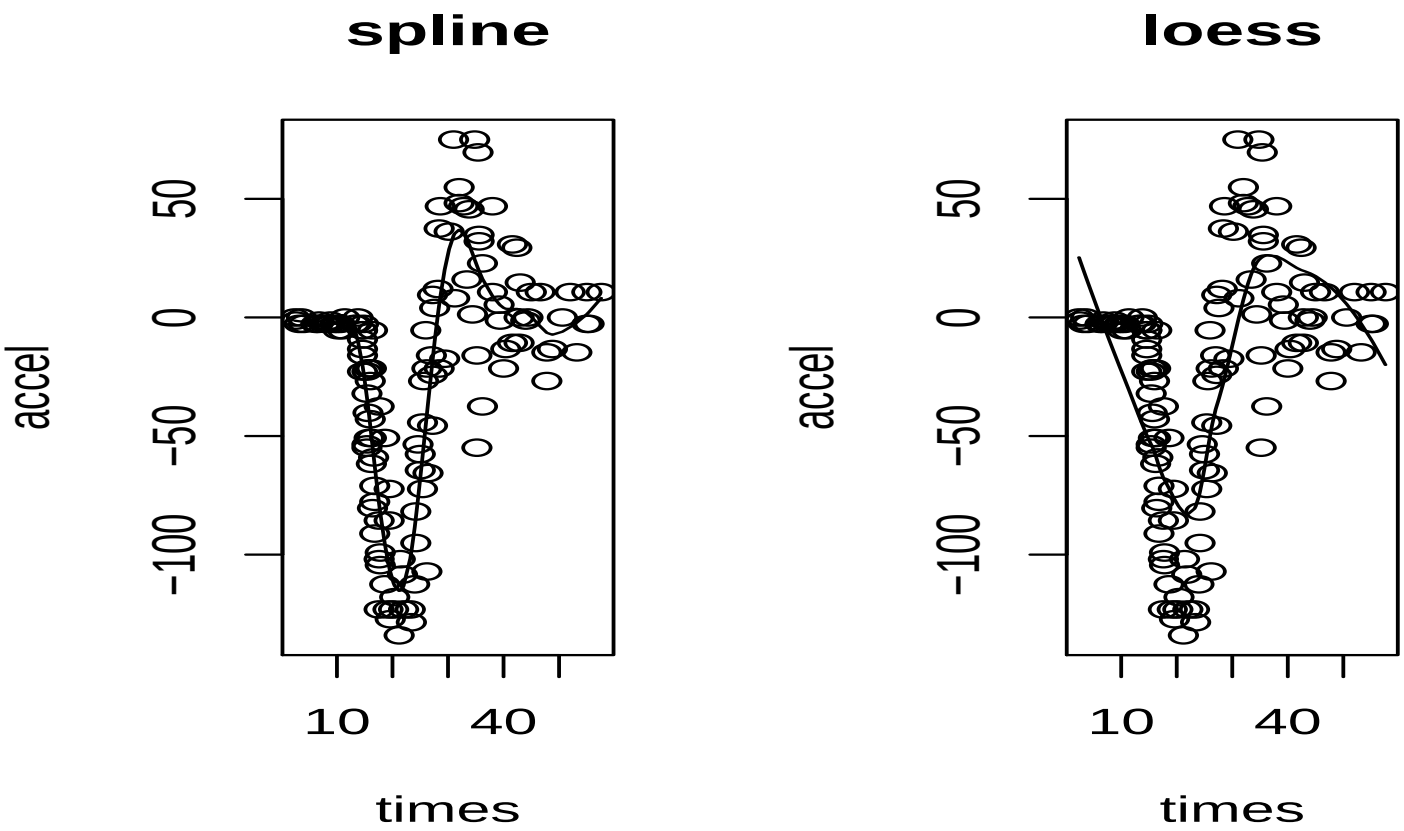
W ustalonej klasie funkcji szukamy rozwiązania problemu minimalizacji funkcji kryterialnej

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - g(x_i))^2 + \lambda \int [g^{(2)}(x)]^2 \, dx,$$

gdzie λ jest pewną wybraną stałą dodatnią. Dla gładkiej i malo zmiennej funkcji g drugi człon, odgrywający rolę kary za zmienność, jest mały. Rozwiązaniem jest kubiczna funkcja sklejana, która między kolejnymi obserwacjami jest wielomianem trzeciego stopnia, tak dobranym, że cała krzywa ma drugą pochodną ciągłą.

Przykład Zbiór `mcycle` w MASS dotyczy symulowanych wypadkow motocyklowych. ?mcycle daje opis zbioru. Estymujemy zależność accel od times.

```
plot(accel~times,mcycle,main="spline")
lines(smooth.spline(mcycle$times,mcycle$accel))
plot(accel~times,mcycle,main="loess")
#uwaga -syntax tu jest troche inny !
f=loess(accel~times,mcycle)
lines(f$x,f$fitted)
plot(accel~times,mcycle,main="ksmooth")
lambda=(range(mcycle$times)[2]-range(mcycle$times)[1])/10
lines(ksmooth(mcycle$times,mcycle$accel,"normal",bandwidth=lambda))
# ksmooth realizuje tzw. estymator jądrowy regresji
```



Domyślny parametr wygładzający dla estymatora lokalnie wielomianowego za duży ! Częsta sytuacja: dobór metodą prób i błędów lub metoda kroswalidacji.

Nieparametryczne estymatory funkcji regresji (duże p)

Problem wymiarowości (curse of dimensionality)

Przypuścmy, że estymujemy funkcję regresji na kwadracie $[0,1] \times [0,1]$ i uznajemy, że żeby dobrze przybliżyć w środkach kwadratów o bokach dlugości 0.1, potrzebne jest 10 obserwacji wewnątrz każdego kwadratu. Potrzebujemy więc $10 \times 10^2 = 1000$ obserwacji dla dobrej estymacji w tych punktach, w przestrzeni trójwymiarowej potrzebujemy analogicznie $10 \times 10^3 = 10000$, w przestrzeni p-wymiarowej 10×10^p obserwacji. Liczba obserwacji potrzebna do satysfakcjonującej estymacji rośnie potęgowo wraz z wymiarem.

Metody przedstawione poprzednio nieskuteczne dla dużego p. Metoda: Przyjęcie założeń o strukturze funkcji regresji i sprowadzenie problemu do estymacji funkcji zależnych od jednej zmiennej.

Drzewa regresyjne

Drzewa regresyjne: konstrukcja taka sama jak dla drzew klasyfikacyjnych CART, jedyna zmiana: w każdym węźle m szukamy podziału na m_L i m_R, aby

$$SSE(m_L) + SSE(m_R) \quad (\star)$$

było minimalne.

$SSE(m_L)$ – suma kwadratów rezyduów, gdy regresja dla m_L estymowana jest przez średnią próbkową wartości zmiennej objaśnianej dla tego węzła itd.
Minimalizacja $(\star)$ równoważna maksymalizacji różnicy zmiany SSE przy przejściu od rodzica do dzieci.
Przycinanie drzewa. Funkcja kryterialna:

$$R_\alpha(T) = SSE(T) + \alpha|T|$$

$|T|$ - liczba liści w drzewie T, $\alpha > 0$. $SSE(T)$: suma SSE dla wszystkich liści.
Estymator regresji: dla kostki związanej z liściem: średnia y w liściu (dzięki przycinaniu i prostej strukturze estymatora unikamy przeuczenia)

MARS Multivriate Addaptive Regression Splines

Wady drzew regresyjnych:

- dopasowana funkcja - schodkowa;
- nie są skuteczne dla modeli addytywnych postaci $y = f_1(x_1) + \ldots + f_p(x_p) + \varepsilon$;
- niestabilność.

Postać estymatora metodą drzewa regresyjnego:

$$\hat{g}(x) = \sum_{j=1}^{p} c_j B_j(x),$$

gdzie

$$B_j(x) = I(x \in N_j) = \prod_{l} H(\pm(x_{v(l)} - t_l)),$$

$H(s) = I\{s \geqslant 0\}$.
Idea: zastąpić funkcje $H(x)$ przez funkcje ciągłe kawałkami liniowe $R^p \to R$ postaci $(\pm(x - t)_+)^q$ + adaptacyjny dobór składników.

Strategia MARS

Rodzina funkcji bazowych (z $R^p \to R$)

$$\mathcal{C} = \{(x_j - t)_+, (t - x_j)_+\}_{j=1,\ldots,p,\, t \in \{x_{1j},\ldots,x_{nj}\}}$$

węzły we wszystkich możliwych wartościach każdego predyktora.
Model postaci

$$g(x) = E(Y|X = x) = \beta_0 + \sum_{m=1}^{M} \beta_m h_m(x),$$

gdzie $h_m \in \mathcal{C}$ lub jest iloczynem pewnej liczby funkcji z $\mathcal{C}$.
$(\beta_0, \beta_1, \ldots, \beta_m)$ estymowane MNK. **problem:** jak wybierać funkcje h_m ?
Adaptacyjny dobór funkcji h_m (analogiczny do wyboru podziału w węźle).
Na pewnym etapie M mamy rodzinę $\mathcal{M}$ funkcji. Na następnym etapie szukamy funkcji postaci $(h_l \in \mathcal{M})$

$$\hat{\beta}_{M+1} h_l(x)(x_j - t)_+ + \hat{\beta}_{M+2} h_l(x)(t - x_j)_+$$

dającej największy spadek SSE i dodajemy do $\mathcal{M}$ nowe funkcje bazowe $h_l(x)(x_j - t)_+$ i $h_l(x)(t - x_j)_+$. Kontynuujemy aż do iloczynów zadanego rzędu, później regresja krokowa.

Model addytywny

Przyjmujemy, że wektor losowy $(Y, \mathbf{X}) \in R^{p+1}$, gdzie $\mathbf{X} = (X_1, \ldots, X_p)'$ spełnia

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

gdzie f_j - nieznane, gładkie funkcje, ε- błąd losowy, taki, że $E\varepsilon = 0$. Model $(\star)$ nadokreślony, dlatego przyjmujemy dla każdego j $Ef_j(X_j) = 0 \Rightarrow \alpha = EY$.

Algorytm dopasowania wstecznego (back-fitting). Oparty na obserwacji, że w modelu $(\star)$

$$E(Y - \alpha - \sum_{j \neq k} f_j(X_j) | X_k = x) = f_k(x)$$

Algorytm dopasowania wstecznego

- krok 0: $\hat{\alpha} = \bar{y}$, $f_j^{[0]} \equiv 0$, $j = 1, \ldots, p$.
- Powtarzaj cyklicznie $j = 1, \ldots, p, 1, \ldots, p, \ldots$ aż do uzyskania maksymalnej różnicy przybliżenia w dwóch kolejnych iteracjach mniejszej od ε:
  $f_j^{[k]}$ estymator nieparametryczny (np. spline) funkcji regresji f_j uzyskany na podstawie danych postaci $(\mathbf{x}, y - \sum_{l \neq j} f_l^{[k-1]}(\mathbf{x}))$.

Przykład Rozpatrzmy zbior ozone zawierający dane o zależności między koncentracją ozonu i warunkami meteo w okolicy Los Angeles 03 -stezenie ozonu ibh-inversion base height, ibt-inversion top temperature. Rozpocznijmy od zwykłego modelu liniowego O3 $\sim$ temp+ibh+ibt

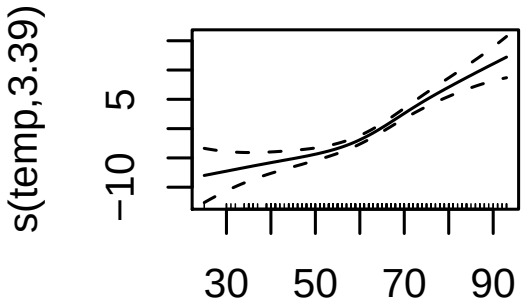
```
olm=lm(O3~temp+ibh+ibt,ozone)
summary(olm)
```

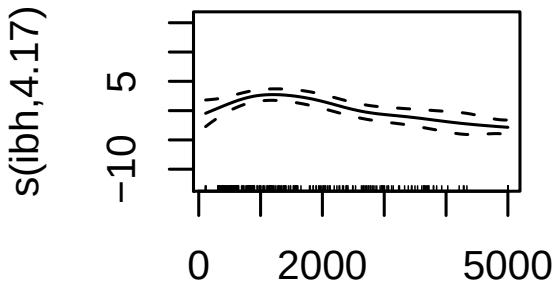
```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -7.7279822  1.6216623  -4.765 2.84e-06 ***
temp         0.3804408  0.0401582   9.474  < 2e-16 ***
ibh         -0.0011862  0.0002567  -4.621 5.52e-06 ***
ibt         -0.0058215  0.0101793  -0.572    0.568
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4.748 on 326 degrees of freedom
Multiple R-Squared: 0.652,      Adjusted R-squared: 0.6488
```

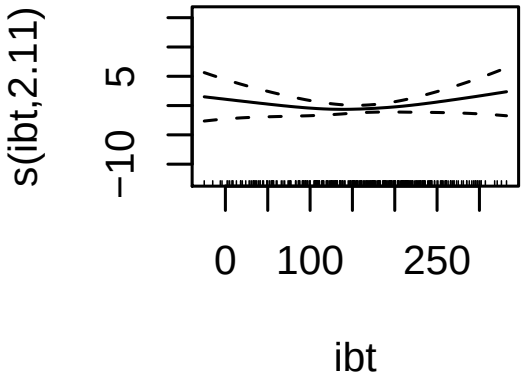
Model addytywny: procedura gam w bibliotece mgcv

```
par(mfrow=c(2,2))
g=gam(O3~s(temp)+s(ibh)+s(ibt),data=ozone)
plot(g)
summary(g)
```

```
Approximate significance of smooth terms:
               edf       chi.sq     p-value
s(temp)      3.386       88.047     < 2.22e-16
 s(ibh)      4.174       37.559     4.2825e-07
 s(ibt)      2.112        4.2263    0.13418

R-sq.(adj) =  0.708   Deviance explained = 71.7%
GCV score = 19.346   Scale est. = 18.72     n = 330
 Procent wyjaśnionego odchylenia 0.71, R^2  w modelu liniowym 0.65
```








```
# pas ufności dla ibt zawiera prosta o wsp. 0, wyrzucamy ja z modelu,
# dla dwu  pozostałych wykres sugeruje zależność kawałkami liniową
```

```
# Definiujemy odpowiednie funkcje bazowe:

rhs=function(x,c) ifelse(x>c,x-c,0)
lhs=function(x,c) ifelse(x<c,c-x,0)

Rysunek sugeruje punkty łamania: 1000 dla ibh i 60 dla temp.
olm1=lm(03~rhs(temp,60)+lhs(temp,60)+rhs(ibh,1000) +lhs(ibh,1000))
print(summary(olm1))
Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)         13.0005414  0.7662612  16.966  < 2e-16 ***
lhs(temp,60)        -0.1161735  0.0378660  -3.068 0.002336 **
rhs(temp,60)         0.5364407  0.0331849  16.165  < 2e-16 ***
lhs(ibh,1000)       -0.0050403  0.0014260  -3.534 0.000468 ***
rhs(ibh,1000)       -0.0014859  0.0001985  -7.486 6.72e-13 ***


Multiple R-Squared: 0.7098,     Adjusted R-squared: 0.7062

Dopasowanie lepsze niż modelu linowego i porównywalne z modelem
addytywnym. Istotna wiedza  na temat konieczności transformacji.
```

Metoda poszukiwania interesujących kierunków

Model funkcji regresji

$$f(\mathbf{x}) = \alpha + \sum_{j=1}^{J} f_j(\alpha_j' \mathbf{x}),$$

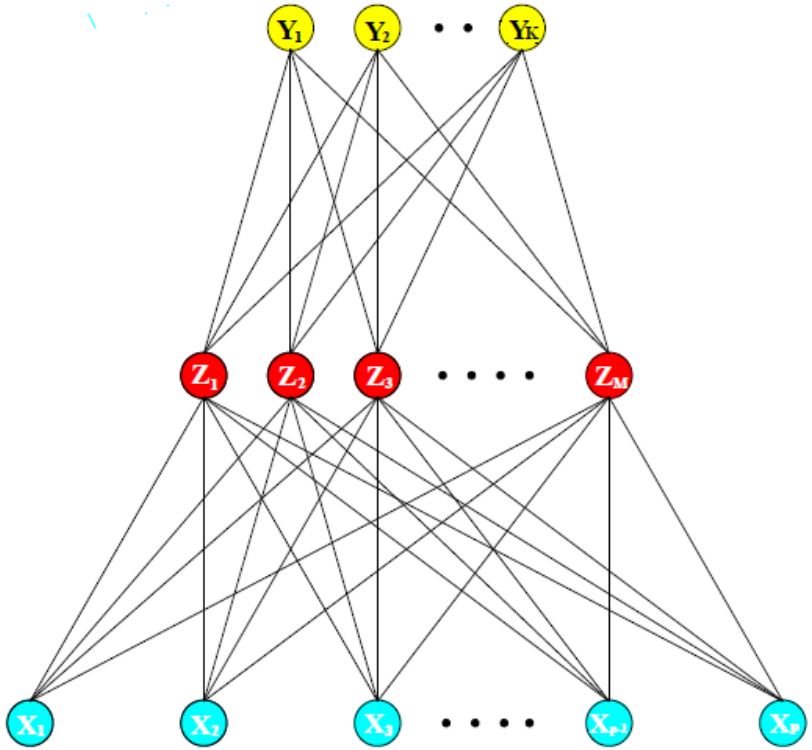
gdzie $\alpha = EY$, α_j są interesującymi kierunkami, które determinują zachowanie funkcji regresji f. J - nieznany parametr. Estymacja oparta na sekwencyjnym wyznaczaniu interesujących kierunków.
Na j-tym etapie wyznaczmy kierunek α_j, dający maksymalną wartość wyrażenia

$$\sum_{i=1}^{n} (r_i - \hat{f}(\alpha' \mathbf{x}_i))^2 / \sum_{i=1}^{n} r_i^2,$$

gdzie r_i: bieżące rezydua po znalezieniu $j-1$ kierunków, a $\hat{f}$ - dowolny ustalony estymator jednowymiarowej funkcji regresji.
Realizacja: funkcja `ppr` w pakiecie MASS.

Sieci neuronowe

'There has been a great deal of hype surrounding neural networks, making them seem magical and mysterious. As we make clear .. they are just nonlinear statistical models, much like projection pursuit regression model ..' (Hastie, Tibshirani, Friedman, ESL)



Obejmują model klasyfikacyjny jak i regresyjny. Dla K-klasowej klasyfikacji $Y_i = (0, \ldots, 1, \ldots, 0)$ (1 na i-tym miejscu). Dla regresji z jednowymiarową odpowiedzią $K = 1$ i $Y_1 \in R$.

$$\mathbf{X} = (X_1, \ldots, X_p)$$

$$Z_m = \sigma(\alpha_{0m} + \alpha'_m \mathbf{X}) \qquad m = 1, \ldots, M$$

$\sigma(\cdot)$- funkcja sigmoidalna lub tanges hiperboliczny

$$\sigma(s) = \frac{e^s}{1 + e^s} = \frac{1}{1 + e^{-s}} \quad \text{lub} \quad \sigma_c(s) = \sigma(cs).$$

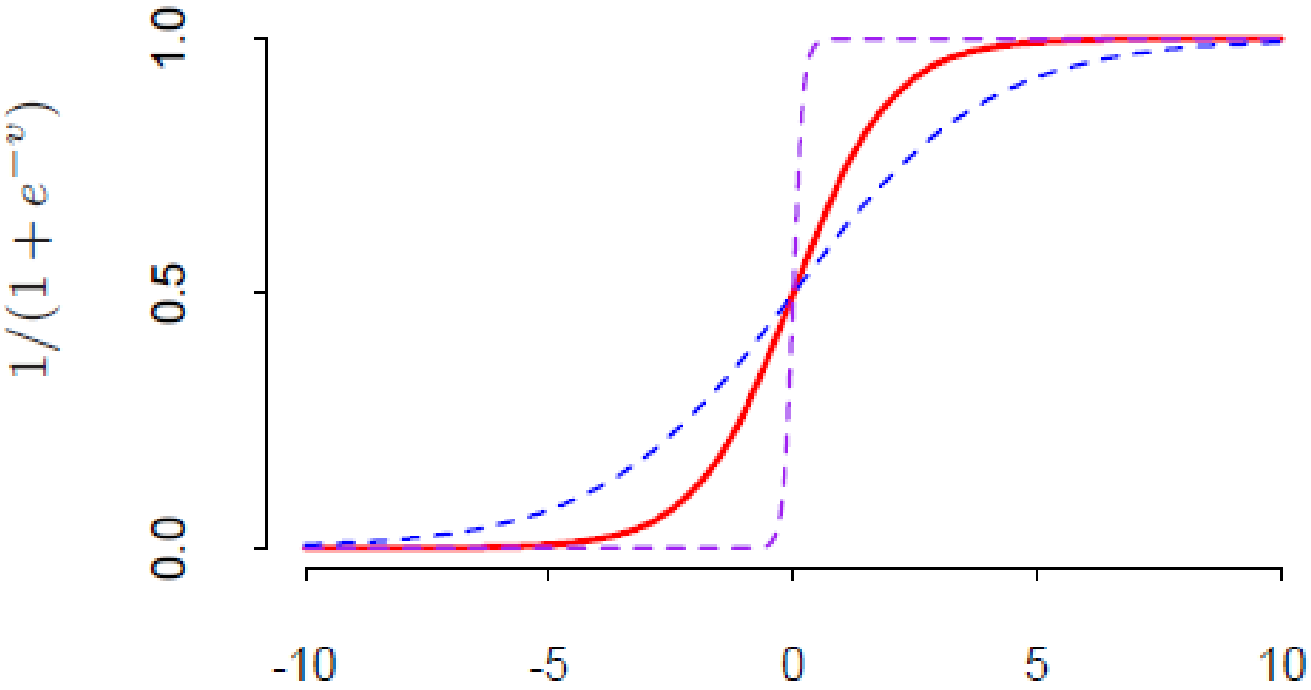
$$\mathbf{Z} = (Z_1, \ldots, Z_M) \quad \mathbf{T} = (T_1, \ldots, T_K): \quad T_k = \beta_{0k} + \beta'_K \mathbf{Z}$$

Aproksymacja Y_k postaci $f_k(\mathbf{X}) = g_k(\mathbf{T})$.
Zwykle dla klasyfikacji g_k: funkcja softmax

$$g_k(\mathbf{T}) = \frac{e^{T_k}}{\sum_{i=1}^{K} e^{T_i}}$$

dla regresji funkcja identycznościowa: $(g_1(\mathbf{T}), \ldots, g_k(\mathbf{T})) = \mathbf{T}$.

$Z_m = \sigma(\alpha_{0m} + \alpha'_m \mathbf{X}), i = 1, \ldots, M$: nowe zmienne objaśniające będące nieliniowymi przekształceniami kobinacji liniowych zmiennych $X_1, \ldots, X_p$. Zauważmy, że $\sigma(s) \approx s$ dla s-małych, zatem jeśli $\alpha_{0m} + \alpha'_m \mathbf{X}$ małe, to $Z_m \approx \alpha_{0m} + \alpha'_m \mathbf{X}$.



Z_m: **zmienne (węzły) ukryte** (ich wartości nie są bezpośrednio obserwowalne). Duża liczba parametrów opisujących model (potrzebna regularyzacja !). Wektor parametrów θ opisujący sieć neuronową obejmuje

$$\alpha_{0m}, \alpha_M, \quad m = 1, \ldots, M \quad \# = M(p+1)$$

$$\beta_{0k}, \beta_k, \quad k = 1, \ldots, K \quad \# = K(M+1)$$

Funkcje kryterialne:
Dla problemu regresji

$$R(\theta) = \sum_{k=1}^{K} \sum_{i=1}^{n} (Y_{ik} - f_k(\mathbf{X}_i))^2$$

$$\hat{Y}(\mathbf{x}) = (\hat{f}_1(\mathbf{x}), \ldots, \hat{f}_K(\mathbf{x})),$$

gdzie $(\hat{f}_1(\mathbf{x}), \ldots \hat{f}_K(\mathbf{x}))$ odpowiada $\hat{\theta} = \operatorname{argmin} R(\theta)$

Dla problemu klasyfikacji:

$$R(\theta) = -\sum_{i=1}^{K}\sum_{i=1}^{n} Y_{ik} \log f_k(\mathbf{X}_i)$$

$$\hat{Y}(\mathbf{x}) = \mathrm{argmax}_k\{\hat{f}_1(\mathbf{x}), \ldots \hat{f}_K(\mathbf{x}\}$$

gdzie $(\hat{f}_1(\mathbf{x}), \ldots \hat{f}_K(\mathbf{x}))$ odpowiada $\hat{\theta} = \mathrm{argmin} R(\theta)$.
Metoda minimalizacji $R(\theta)$ -metoda gradientu, zwana dla tego problemu metodą propagacji wstecznej (*back-propagation*). Ponieważ funkcja regresji w modelu jest złożeniem kilku funkcji, gradient można prosto wyliczyć metodą łańcuchową.

Propagacja wsteczna dla regresji

$(\alpha_{0m}, \beta_{0m} = 0)$

$$R(\theta) = \sum_{i=1}^{n} R_i(\theta) =: \sum_{i=1}^{K} \sum_{i=1}^{n} (Y_{ik} - f_k(\mathbf{X}_i))^2$$

$$\frac{\partial R_i}{\partial \beta_{km}} = -2(Y_{ik} - f_k(\mathbf{X}_i)) g_k'(\beta_k' \mathbf{Z}_i) Z_{mi}$$

$$\frac{\partial R_i}{\partial \alpha_{ml}} = -\sum_{k=1}^{K} 2(Y_{ik} - f_k(\mathbf{X}_i)) g_k'(\beta_k' \mathbf{Z}_i) \beta_{km} X_{il}$$

Zdefiniujmy

$$\frac{\partial R_i}{\partial \beta_{km}} = \delta_{ki} Z_{mi}$$

i

$$\frac{\partial R_i}{\partial \alpha_{ml}} = s_{mi} X_{il}$$

Wielkości δ_{ki} i s_{mi} mają interpretację 'błędów' dla bieżacego modelu na poziomie wyjścia i zmiennych ukrytych odpowiednio. Ich związek

$$s_{mi} = \sigma'(\alpha'_m \mathbf{X}_i) \sum_{k=1}^{K} \beta_{km} \delta_{ki} \qquad (*)$$

(**równanie propagacji wstecznej**)
Propoagacja wsteczna: algorytm dwustopniowy.
Krok wprzód (*forward pass*):
Ustalone wagi, na ich podstawie liczymy $\hat{f}_k(\mathbf{X}_i)$
Krok w tył (*backward pass*): Liczymy δ_{ki} i propagujemy je w tył licząc s_{mi} na podstawie (*). Mając δ_{ki} i s_{mi} liczymy gradienty i nową iterację współczynników

$$\beta_{km}^{(r+1)} = \beta_{km}^{(r)} - \gamma_r \sum_{i=1}^{n} \frac{\partial R_i}{\partial \beta_{km}}_{|\beta_{km}=\beta_{km}^{(r)}}$$

$$\alpha_{ml}^{(r+1)} = \alpha_{ml}^{(r)} - \gamma_r \sum_{i=1}^{n} \frac{\partial R_i}{\partial \alpha_{ml}}_{|\alpha_{ml}=\alpha_{ml}^{(r)}}$$

γ_r- szybkość uczenia (*learning rate*).

Zaleta propagacji wstecznej: jej charakter lokalny. Węzły ukryte dostają/przekazują informację od/do węzłów, z którymi mają połaczenia.
Wersja omówiona: wersja wsadowa (*batch learning*). Oprócz tego istnieje wersja sekwencyjna (gradient jest modyfikowany po każdej obserwacji).
γ_r: dla wersji wsadowej stała. Dla wersji sekwencyjnej musi spełniać warunki dla algorytmów aproksymacji stochastycznej:

$$\gamma_r \to 0 \qquad \sum_r \gamma_r = \infty \qquad \sum_r \gamma_r^2 < \infty$$

(częsty wybór $\gamma_r = r^{-1}$).

Kilka uwag

- Wagi początkowe - często bliskie 0 i wtedy sieć neuronowa - w przybliżeniu model liniowy. W miarę jak wagi się zwiększają nieliniowość modelu się zwiększa.
- **Regularyzacja** Zamiast optymalizować $R(\theta)$ optymalizujemy

$$R(\theta) + \lambda \left(\sum_{k,m} \beta_{km}^2 + \sum_{m,l} \alpha_{ml}^2 \right),$$

  $\lambda \geqslant 0$. Prosta adaptacja metody propagacji wstecznej opisanej poprzednio.
- Zmienne wyjściowe: często standaryzowane do średniej 0 i wariancji 1.
- Liczba węzłow ukrytych 5 do 100 (lepiej więcej i później je wyeliminować, niż zadać na początku za mało)

Dopasowanie sieci neuronowej z dwoma węzłami ukrytymi i liniowym wyjsciem dla danych ozone

```
nnet2=nnet(upo3~vdht +ibht+ibtp,ozone,size=2,linout=T)
# to jest nnet2=nnet(O3~temp +ibh+ibt,ozone,size=2,linout=T)
# weights:  11
initial  value 69600.383457
final  value 21115.406061
converged
# dla modelu zerowego
sum((ozone[,1]-mean(ozone[,1]))^2)
[1] 21115.41
#skalujemy zmienne
 sx=scale(ozone)
  nnet2=nnet(upo3~vdht +ibht+ibtp,sx,size=2,linout=T)
   weights:  11
initial  value 562.397017
iter  10 value 134.691891
iter  20 value 126.240450
iter  30 value 125.517936
iter  40 value 123.059981
iter  50 value 122.981601
iter  60 value 122.981315
final  value 122.981245
```

```
converged
# z regularyzacja
 nnet3=nnet(upo3~vdht +ibht+ibtp,sx,size=2,decay=0.001,linout=T)
# weights:  11
initial  value 359.766539
.........................
iter  80 value 125.844137
iter  90 value 125.839401
iter 100 value 125.826962
final  value 125.826962
stopped after 100 iterations

# inna funkcja kryterialna w tym wypadku.
```