Kroswalidacja, bootstrap i inne

Melka Kamil

Uniwersytet Wrocławski

11.04.2022

Plan prezentacji

Oznaczenia

- Ogólne działanie metod uczących jest powiązane ze zdolnością predykcji na niezależnych danych testowych. Jest to istotne z praktycznego punktu widzenia.
- Y - zmienna objaśniana.
- X - wektor zmiennych objaśniających.
- $\hat{f}(X)$ - model utworzony na podstawie zbioru treningowego $\mathcal{T}$.
- Funkcja straty mierząca błędy pomiędzy Y i $\hat{f}(X)$.

$$L(Y, \hat{f}(X)) = \begin{cases} (Y - \hat{f}(X))^2 & \text{błąd kwadratowy,} \\ |Y - \hat{f}(X)| & \text{błąd bezwzględny.} \end{cases}$$

Błędy

- $\text{Err}_\tau = E[L(Y, \hat{f}(X))|\mathcal{T}]$ - błąd testowy.
- $\text{Err} = E[L(Y, \hat{f}(X))] = E[\text{Err}_\tau]$ - oczekiwany błąd testowy/predykcji.
- $\overline{\text{err}} = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{f}(x_i))$ - błąd treningowy.

Minimum Description Length

- To podejście jest podobne do BIC, natomiast jest ono lepsze w przypadku optymalnego kodowania.
- Myślimy o danej z jako o wiadomości, którą chcemy zakodować i wysłać do "odbiorcy".
- O modelu myślimy w taki sposób, aby zakodowane dane dały najkrótszy możliwy sposób na przekazanie wiadomości.

Przykład

- Kod może używać skończonego alfabetu długości A, np. kod binarny $\{0, 1\}$ ma długość $A = 2$.
- Przykładowo chcemy zakodować dane z_1, z_2, z_3, z_4. W tym celu możemy użyć kodowania 0, 10, 110, 111 (żadne z kodowań nie jest przedrostkiem innego, przez co będziemy mieli jednoznaczność w odczytywaniu wiadomości).

Wiadomość	z_1	z_2	z_3	z_4
Kod	0	10	110	111

Przykład

- Jak dobrać kodowanie do danych z_1, z_2, z_3 i z_4?
- Dla najczęściej występującej wiadomości chcielibyśmy mieć jak najkrótszy kod.
- Załóżmy, że wiadomości są wysyłane z pewnym prawdopodobieństwem $P(z_i)$, $i = 1, 2, 3, 4$.
- Wtedy zgodnie z twierdzeniem Shannona powinniśmy użyć kodu długości $l_i = -\log_2 P(z_i)$ i średnia długość wiadomości spełnia $E(\text{długość}) \geq -\sum P(z_i) \log_2(P(z_i))$.
- Przesłanie wiadomości o zmiennej losowej, której prawdopodobieństwo wystąpienia wynosi $P(z)$, potrzebuje około $-\log_2 P(z)$ bitów informacji.

Zastosowanie

- Model M z parametrami θ i danymi $\mathbf{Z} = (\mathbf{X}, \mathbf{y})$.
- Rozważmy $P(\mathbf{y}|\theta, M, \mathbf{X})$, które określa prawdopodobieństwo wektora wyjścia przy danym modelu M.
- Wtedy długość wiadomości to:
  długość $= -\log P(\mathbf{y}|\theta, M, \mathbf{X}) - \log P(\theta|M)$.
- Chcemy minimalizować tę wartość, co jest równoważne maksymalizowaniu prawdopodobieństwu a posteriori.

Wordle

Rysunek: Przykładowe rozwiązanie gry Wordle, `https://www.youtube.com/watch?v=v68zYyaEmEA`

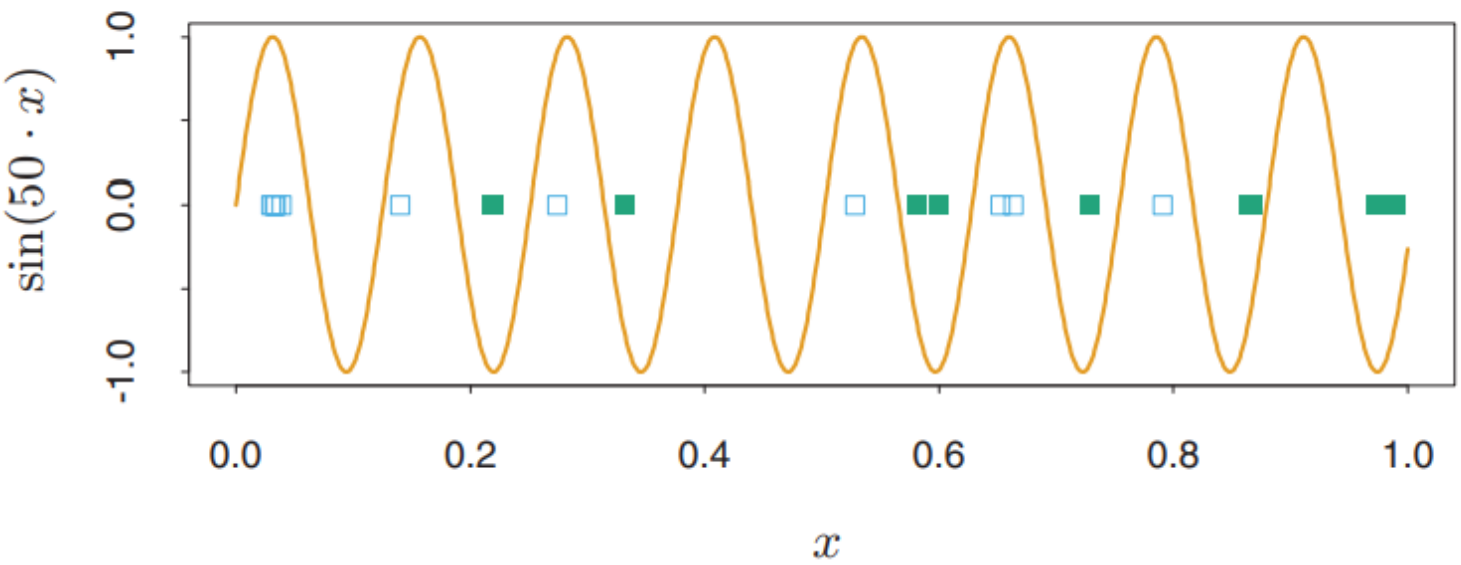
Wymiar Wapnika-Czerwonienkisa

- Trudnością w estymowaniu błędów jest to, aby dobrze określić liczbę parametrów używanych w modelu.
- Wymiar Wapnika-Czerwonienkisa jest ogólną miarą złożoności, która może pomóc w tym problemie.
- Rozważmy klasę funkcji $\{f(x, \alpha)\}$ indeksowaną przez wektor parametrów α, $x \in \mathbb{R}^p$.

Początkowe podejście

- Załóżmy, że f jest funkcją charakterystyczną. Jeśli $\alpha = (\alpha_0, \alpha_1)$ i $f = \mathbb{1}(\alpha_0 + \alpha_1^T x > 0)$.
- Wydaje się zasadne określić poziom złożoności modelu jako $p + 1$, bo tyle parametrów potrzebujemy do określenia modelu.
- Rozważmy to podejście dla $p = 1$. Wtedy pojawia się pytanie, czy funkcja $f(x, \alpha) = \mathbb{1}(sin(\alpha \cdot x))$ ma większą złożoność od $\mathbb{1}(\alpha_0 + \alpha_1 x)$?

Przykład



Rysunek: Z rysunku można zobaczyć jak dużo punktów może rozdzielić funkcja $\mathbb{1}(sin(\alpha x) > 0)$, źródło: *Elements of statistical learning, fig. 7.5*

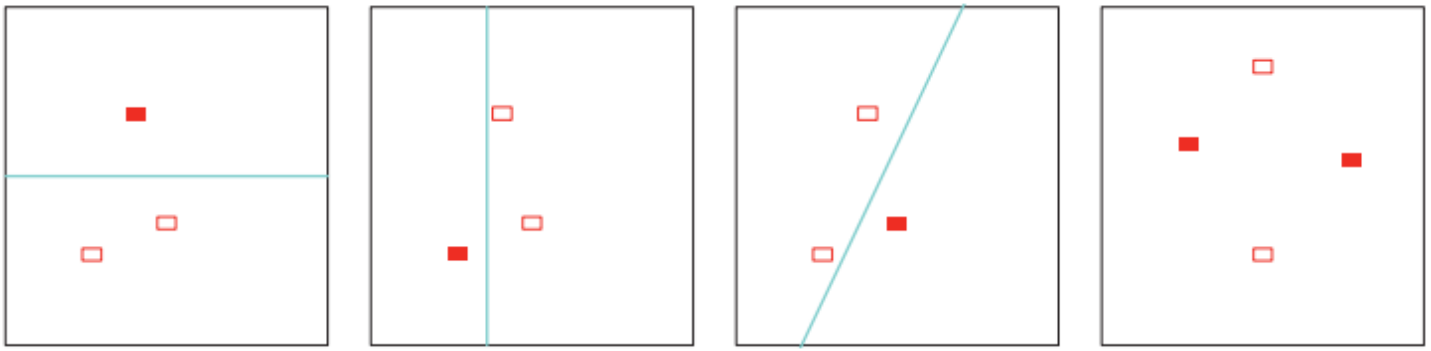
Definicja

Wymiar Wapnika-Czerwonienkisa jest sposobem mierzenia złożoności klasy funkcji, ustalając jak bardzo te funkcje są faliste.

Definicja

Wymiar Wapnika-Czerwonienkisa klasy $\{f(x, \alpha)\}$ *jest równy największej liczbie punktów, które mogą być rodzielone przez pewną funkcję z klasy* $\{f(x, \alpha)\}$.

Przykład



Rysunek: Z rysunku można zobaczyć, że dla trzech punktów zawsze można znaleźć prostą rozdzielającą klasy, natomiast dla czterech punktów nie jest to możliwe, źródło: *Elements of statistical learning, fig. 7.6*

Rozszerzenie klas funkcji do funkcji rzeczywistych

- Niech $\{g(x, \alpha)\}$ będzie klasą funkcji rzeczywistych.
- Wtedy wymiar Wapnika-Czerwonienkisa takiej klasy jest równy wymiarowi klasy funkcji $\{\mathbb{1}(g(x, \alpha) - \beta > 0)\}$, gdzie β przyjmuje wartości należące do przeciwdziedziny g.
- Wymiar Wapnika-Czerwonienkisa można wykorzystać do szacowania Err_τ.

Szacowania Err_τ

Jeśli modelujemy N punktów treningowych, używając klasy funkcji $\{f(x,\alpha)\}$ o wymiarze Wapnika-Czerwonienkisa równym h, to z prawdopodobieństwem większym od $1-\eta$ mamy:

$$\mathrm{Err}_\tau \leq \overline{\mathrm{err}} + \frac{\varepsilon}{2}\left(1+\sqrt{1+\frac{4\cdot\overline{\mathrm{err}}}{\varepsilon}}\right) \quad \text{(klasyfikator binarny)}$$

$$\mathrm{Err}_\tau \leq \frac{\overline{\mathrm{err}}}{(1-c\sqrt{\varepsilon})_+} \quad \text{(regresja)},$$

gdzie $\varepsilon = a_1 \frac{h[\log(a_2 N/h)+1]-\log(\eta/4)}{N}$ i $0 < a_1 \leq 4$, $0 < a_2 \leq 2$.

Alternatywne oszacowanie Err_τ dla regresji

$$\mathrm{Err}_\tau \leq \overline{\mathrm{err}} \left(1 - \sqrt{\rho - \rho \log \rho + \frac{\log N}{2N}} \right)_+^{-1},$$

gdzie $\rho = \frac{h}{N}$.

SRM - structural risk minimization

- W tym podejściu używamy sekwencji zagnieżdżonych modeli z rosnącym wymiarem Wapnika-Czerwonienkisa $h_1 < h_2 < \cdots$, a następnie wybieramy model z najmniejszą wartością górnego oszacowania Err_τ.
- To podejście może być używane z sukcesem w problemie *support vector classifier*.

K-Fold Cross-Validation

- Dzielimy zbiór danych na K części o zbliżonej liczności (najczęściej $K = 5$ lub $K = 10$).
- Dla k-tego zbioru obliczamy błąd predykcji na podstawie modelu utworzonego z pozostałych $K - 1$ podzbiorów.
- Robimy to dla $k = 1, 2, \ldots, K$ i następnie łączymy w konkretny sposób te K estymatorów.

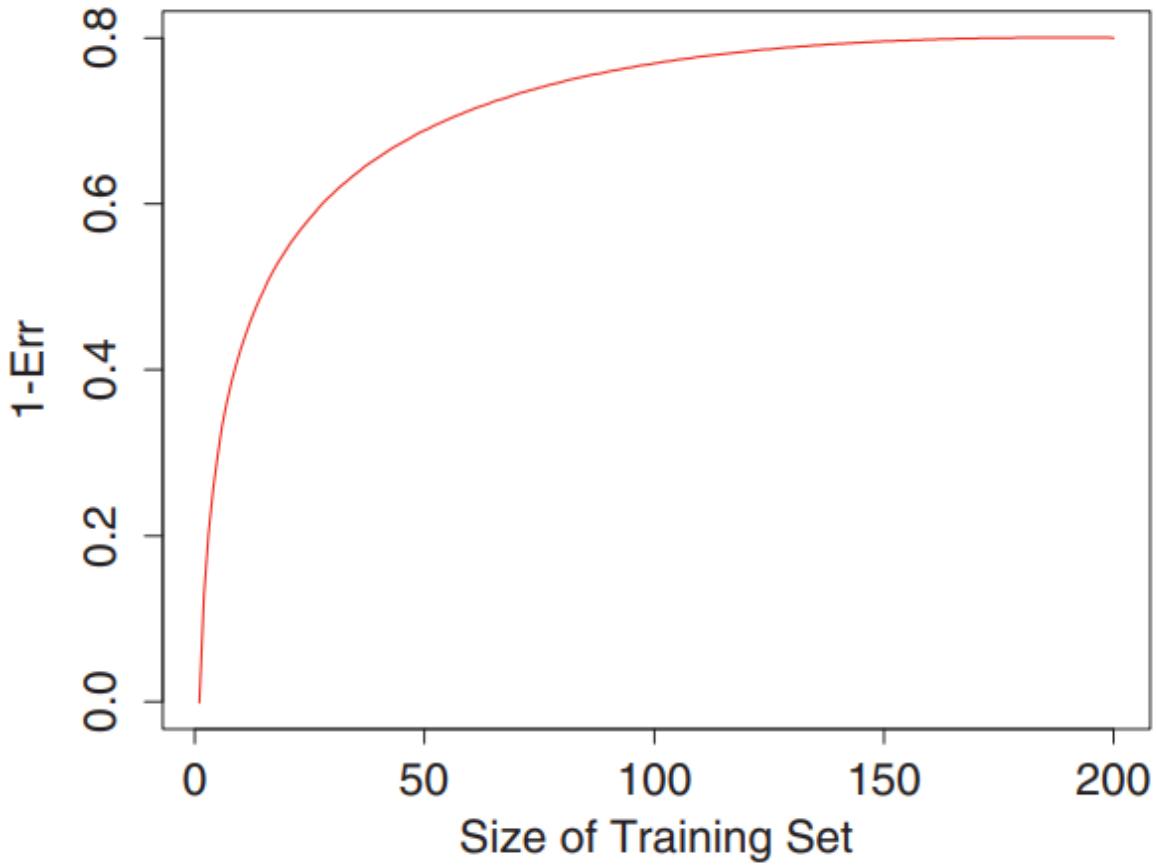
K-Fold Cross-Validation

- Niech $\kappa : \{1, \ldots, N\} \to \{1, \ldots, K\}$ będzie funkcją przypisującą każdej obserwacji przynależność do danego podzbioru (zazwyczaj robi się to w sposób losowy).
- Niech $\hat{f}^{-k}(x)$ oznacza funkcję modelującą utworzoną na podstawie danych, które nie mają k-tego podzbioru.
- Wtedy estymacja błędu predykcji za pomocą kroswalidacji przedstawia się jako: $CV(\hat{f}) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, \hat{f}^{-\kappa(i)}(x_i))$.
- W przypadku $K = N$ mamy do czynienia z *leave-one-out cross-validation*. Wtedy $\kappa(i) = i$.

Obciążenie i wariancja

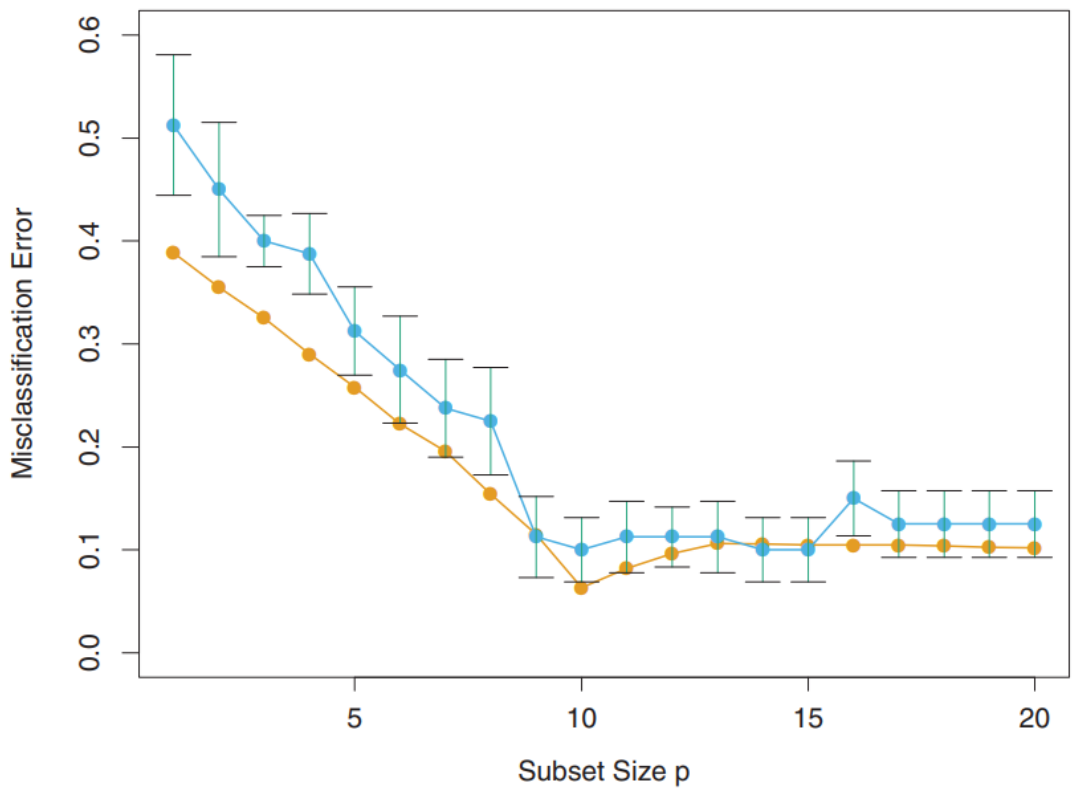
- Dla $K = N$ zbiory treningowe są bardzo zbliżone do siebie, przez co estymator błędu predykcji jest niemal nieobciążony, ale może mieć dużą wariancję.
- Dla $K = 5$ estymator ma mniejszą wariancję, ale przy niedostatecznie dużej próbie może wystąpić obciążenie.

Krzywa ucząca



Rysunek: Hipotetyczna krzywa ucząca dla pewnego klasyfikatora, źródło: *Elements of statistical learning, fig. 7.8*

Obciążenie błędu predykcji



Rysunek: Błąd predykcji (pomarańczowy) oraz krzywa *10-fold cross-validation* estymowana na podstawie pojedynczego zbioru treningowego, źródło: *Elements of statistical learning, fig. 7.9*

Generalized cross-validation

- To podejście daje nam wygodny sposób na aproksymację metody *leave-one-out cross-validation* dla regresji liniowej z kwadratową funkcją straty.
- Powyższą regresję można zapisać w taki sposób: $\hat{\mathbf{y}} = \mathbf{S}\mathbf{y}$.
- Wtedy *leave-one-out cross-validation* daje:
  $\frac{1}{N}\sum_{i=1}^{N}\left[y_i - \hat{f}^{-i}(x_i)\right]^2 = \frac{1}{N}\sum_{i=1}^{n}\left[\frac{y_i - \hat{f}(x_i)}{1 - S_{ii}}\right]^2$, gdzie S_{ii} to i-ty element na diagonali $\mathbf{S}$.
- Wtedy można zrobić przybliżenie za pomocą GCV:
  $$\mathrm{GCV}(\hat{f}) = \frac{1}{N}\sum_{i=1}^{N}\left[\frac{y_i - \hat{f}(x_i)}{1 - \mathrm{trace}(\mathbf{S})/N}\right]^2.$$

Sposób użycia kroswalidacji

Rozważmy problem klasyfikacji z dużą liczbą predyktorów. Typowa strategia może być następująca:

1. Zmniejsz liczbę predyktorów: znajdź podzbiór "dobrych"predyktorów, które są silnie skorelowane z wartościami klas.
2. Używając tego zbioru predyktorów zbuduj wielowymiarowy klasyfikator.
3. Używając kroswalidacji, wyestymuj nieznaną wartość parametrów oraz wartość błędu predykcji ostatecznego modelu.

Błędne działanie kroswalidacji

- Rozważmy scenariusz, gdzie mamy próbkę rozmiaru $N = 50$, która jest podzielona na dwie równoliczne klasy. Załóżmy, że mamy $p = 5000$, gdzie predyktory są iid $\sim N(0, 1)$. Są one niezależne z wektorem oznaczeń klas.
- Wtedy prawdziwy błąd dla klasyfikacji wynosi 50%.
- Wykonujemy algorytm z poprzedniego slajdu, używając w drugim kroku 1-NN classifier.
- Po 50 symulacjach otrzymaliśmy wartość błędu na poziomie 3%.

Poprawny sposób użycia kroswalidacji

1. Podzielmy w sposób losowy próbę na K podzbiorów.
2. Dla każdego podzbioru $k = 1, 2, \ldots, K$
   - Znajdujemy podzbiór "dobrych"predyktorów, które są silnie skorelowane z wartościami klas, używając wszystkich prób spoza k-tego podzbioru.
   - Używając tego zbioru predyktorów budujemy wielowymiarowy klasyfikator, także używając wszystkich prób spoza k-tego podzbioru.
   - Używamy powyższego klasyfikatora do predykcji wartości klas dla próby z k-tego podzbioru.

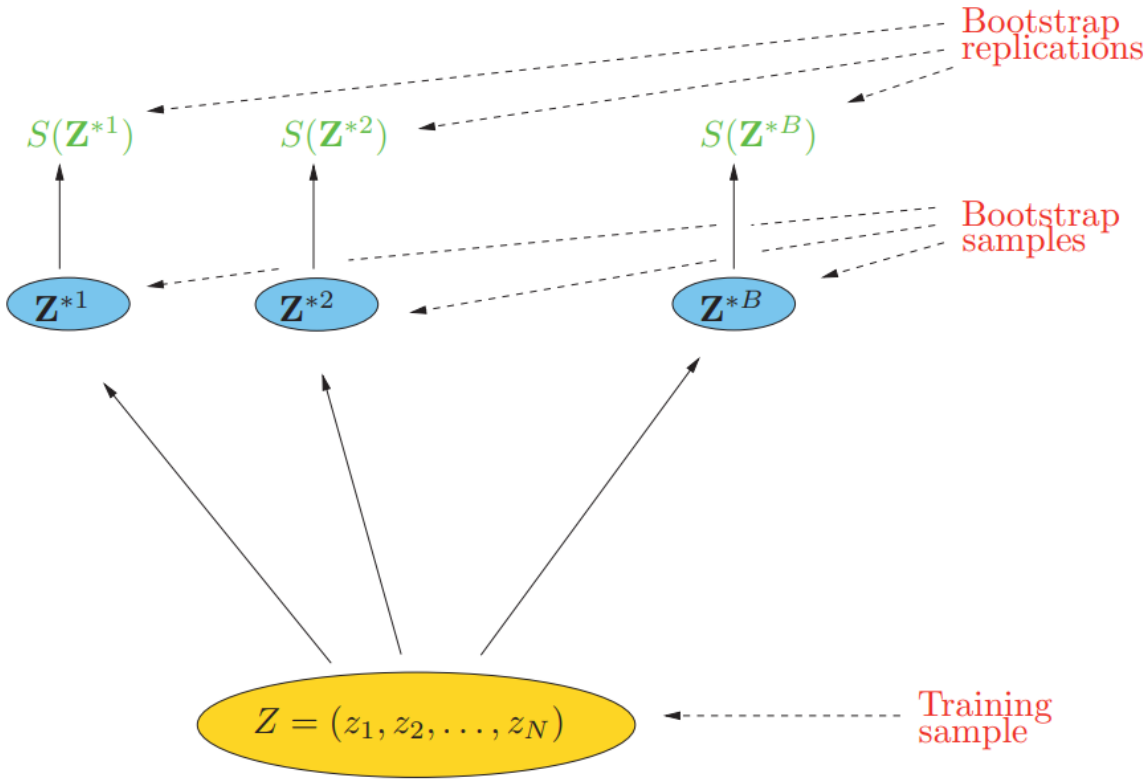
Porównanie złego i dobrego podejścia

Rysunek: Histogramy pokazują korelacje między wektorem wartości klas a 100 predyktorami wybranymi przez dany algorytm (na podstawie 10 losowo wybranych próbek), źródło: *Elements of statistical learning, fig. 7.10*

Bootstrap

- Bootstrap jest ogólnym narzędziem do oceniania statystycznej dokładności.
- Opiszmy zbiór treningowy jako $\mathbf{Z} = (z_1, z_2, \ldots, z_N)$, gdzie $z_i = (x_i, y_i)$.
- Podstawową ideą jest wylosowanie nowego zestawu danych z powyższego zbioru treningowego. Robimy to losując ze zwracaniem.
- Robimy taką procedurę B razy, następnie wyznaczamy na tej podstawie model i ostatecznie obserwujemy zachowania modeli na podstawie tych B replikacji.

Schemat procesu bootstrapowego



Rysunek: Schemat procesu bootstrapowego, źródło: *Elements of statistical learning, fig. 7.12*

Estymacja

- Za pomocą bootstrapu możemy np. estymować wariancję statystyki S obliczonej na podstawie zbioru treningowego $\mathbf{Z}$.
- $\widehat{\mathrm{Var}}[S(\mathbf{Z})] = \frac{1}{B-1}\sum_{b=1}^{B}(S(\mathbf{Z}^{*b}) - \overline{S}^{*})^2$, gdzie $\overline{S}^{*} = \sum_b S(\mathbf{Z}^{*b})/B$.
- W podobny sposób można próbować estymować błąd predykcji.
- $\widehat{\mathrm{Err}}_{\mathrm{boot}} = \frac{1}{B}\frac{1}{N}\sum_{b=1}^{B}\sum_{i=1}^{N} L(y_i, \hat{f}^{*b}(x_i))$.

1-NN classifier

- Prawdziwy błąd predykcji wynosi 0.5.
- $\widehat{\text{Err}}_{\text{boot}} = 0$, gdy i-ta obserwacja pojawia się w próbie b. W przeciwnym przypadku oczekiwany błąd będzie wynosić 0.5.
- $P(\text{obserwacja } i \in \text{bootstrapowej próbki } b) = 1 - (1 - \frac{1}{N})^N \approx 1 - e^{-1} = 0.632$.
- Stąd oczekiwana wartość $\widehat{\text{Err}}_{\text{boot}}$ wynosi około $0.5 \cdot 0.368 = 0.184$.

Leave-one-out bootstrap

- $\widehat{\text{Err}}^{(1)} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{|C^{-i}|}\sum_{b\in C^{-i}} L(y_i, \hat{f}^{*b}(x_i))$, gdzie C^{-i} jest zbiorem indeksów bootstrapowych próbek b, które nie zawierają i-tej obserwacji.
- $\widehat{\text{Err}}^{(0.632)} = 0.368 \cdot \overline{\text{err}} + 0.632 \cdot \widehat{\text{Err}}^{(1)}$.
- Dla 1-NN classifier mamy $\overline{\text{err}} = 0$, $\widehat{\text{Err}}^{(1)} = 0.5$. Wtedy $\widehat{\text{Err}}^{0.632} = 0.632 \cdot 0.5 = 0.316$.

Dziękuję za uwagę.