

Techniki eksploracji danych

Krzysztof Gajowniczek

Rok akademicki: 2020/2021

- 1 Teoretyczne podstawy przetwarzania równoległego
- 2 Wykorzystywana notacja matematyczna
- 3 Miary jakości modeli
- 4 Sprawdzian krzyżowy
- 5 Optymalizacja parametrów
- 6 Literatura

Section 1

Teoretyczne podstawy przetwarzania równoległego

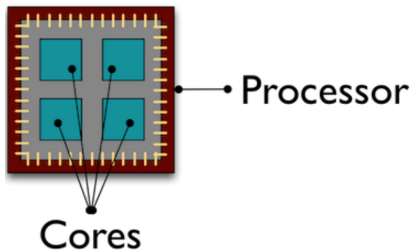
- Przetwarzanie dużych ilości danych za pomocą złożonych modeli może być czasochłonne.
- Nowe rodzaje wykrywania oznaczają, że skala gromadzenia danych jest obecnie ogromna, a modelowane wyniki mogą być również duże.
- Jeśli coś zajmuje mniej czasu, jeśli odbywa się poprzez **przetwarzanie równoległe**, dlaczego nie zrobić tego i zaoszczędzić czas?
- Nowoczesne laptopy i komputery PC mają obecnie **procesory wielordzeniowe** z wystarczającą ilością dostępnej pamięci, którą można wykorzystać do szybkiego generowania wyników.

- Zrównoleglenie kodów ma swoje liczne zalety.
- Zamiast czekać kilka minut lub godzin na zakończenie zadania, można podmienić kod, uzyskać dane wyjściowe w ciągu kilku sekund lub minut i jednocześnie sprawić, by było to wydajne.
- Efektywność kodu jest obecnie jedną z najbardziej poszukiwanych umiejętności w branży i niewiele osób jest w stanie z niej korzystać.
- Kiedy już nauczysz się zrównoleglania kodu, będziesz tylko żałować, że nie nauczyłeś się go wcześniej.

- Wiele programów R działa szybko i dobrze na jednym procesorze. Ale czasami obliczenia mogą być:
 - **cpu-bound**: zajmuje zbyt dużo czasu procesora.
 - **memory-bound**: zajmuje zbyt dużo pamięci.
 - **I/O-bound**: odczyt/zapis z dysku zajmuje zbyt dużo czasu.
 - **network-bound**: przesyłanie zajmuje zbyt dużo czasu.

- Sercem każdego komputera jest nowoczesny procesor CPU (ang. Central Processing Unit).
- Podczas gdy tradycyjne komputery miały jeden procesor, nowoczesne komputery posiadają wiele procesorów, z których każdy może zawierać wiele rdzeni.
- Te procesory i rdzenie są dostępne do wykonywania obliczeń.

- Komputer z jednym procesorem może nadal mieć 4 rdzenie (quad-core), co pozwala na jednoczesne wykonywanie 4 obliczeń.



Subsection 1

Podejścia do równoległości w R

- R może korzystać z kilku struktur/narzędzi/pakietów, aby umożliwić zrównoleglenie:
- `parallel` jest prawdopodobnie najbardziej rozwiniętym i najbardziej dojrzałym i opiera się na pracy z pakietem `multicore` i `snow` (z których pierwszy został usunięty z CRAN, ponieważ został całkowicie wchłonięty przez `parallel`), z których drugi może być używane do zrównoleglania obliczeń na jednym lub wielu komputerach.
- `foreach` umożliwia zrównoleglenie poprzez rozszerzenia `foreach` pętlifor, pakiet `doMC` i słowo kluczowe `%dopar%`. Może korzystać z frameworka `multicore`, z tymi samymi ograniczeniami systemu operacyjnego wymienionymi powyżej, lub frameworka `snow`.

Subsection 2

Pakiet parallel

```
library(parallel)
no_cores <- detectCores()
clust <- makeCluster(no_cores)
parLapply(clust,1:3, function(x) c(x^2,x^3))
```

```
## [[1]]
## [1] 1 1
##
## [[2]]
## [1] 4 8
##
## [[3]]
## [1] 9 27
```

```
stopCluster(clust)
```

```
library(parallel)
no_cores <- detectCores()
clust <- makeCluster(no_cores)
base <- 4
clusterExport(clust, "base")
parSapply(clust, 1:5, function(exponent) base^exponent)

## [1]      4     16     64    256   1024

stopCluster(clust)
```

Subsection 3

Pakiety `doParallel` i `foreach`

```
library(doParallel)
```

```
## Loading required package: foreach
```

```
## Loading required package: iterators
```

```
cl <- makeCluster(2)  
registerDoParallel(cl)  
foreach(i=1:3) %dopar% sqrt(i)
```

```
## [[1]]
```

```
## [1] 1
```

```
##
```

```
## [[2]]
```

```
## [1] 1.414214
```

```
##
```

```
## [[3]]
```

- Pakiet doParallel umożliwia określenie różnych opcji podczas uruchamiania polecenia foreach, które są obsługiwane przez podstawową funkcję mclapply: “preschedule”, “set.seed”, “silent” i „cores”.

```
mcoptions <- list(preschedule=FALSE, set.seed=FALSE)
foreach(i=1:3, .options.multicore=mcoptions) %dopar% sqrt(i)

## [[1]]
## [1] 1
##
## [[2]]
## [1] 1.414214
##
## [[3]]
## [1] 1.732051
```


Section 2

Wykorzystywana notacja matematyczna

Wyjaśnienie symboli

- n - liczba obserwacji.
- i - iterator obserwacji, $i \in \{1, \dots, n\}$.
- y - zmienna celu.
 - dla problemu regresji: $y \in \mathbb{R}$.
 - dla problemu klasyfikacji: $y \in \{1, \dots, l, \dots, k\}$.
 - y_i - i -ta wartość zmiennej celu.
- \hat{y} - wartość teoretyczna zmiennej celu.
 - dla problemu regresji: $\hat{y} \in \mathbb{R}$.
 - dla problemu klasyfikacji: $\forall l, \hat{y} \in [0, 1], \sum_{l=1}^k \hat{y}^{(l)} = 1$.
- x - atrybut, $x \in X$.
- j - iterator atrybutu, $j \in \{1, \dots, p\}$.
 - $\mathbf{x}^T \in X^{(p)}$.

Section 3

Miary jakości modeli

Subsection 1

Miary dla problemów regresyjnych

Mean absolute error

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|$$

Mean squared error

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

Mean absolute percentage error

$$MAPE = \frac{1}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right|$$

Subsection 2

Miary dla problemów klasyfikacyjnych

- W zależności o wykorzystanej miary, pracować będziemy albo na wektorach prawdopodobieństwa:

$$\hat{\mathbf{y}}^T = \{\hat{y}^{(1)}, \hat{y}^{(2)}\} = \{0.35, 0.65\}$$

- albo na finalnych etykietach klas, wyznaczanych jako:

$$\hat{y} = \arg \max_l \hat{y}^{(l)} = \{2\}$$

- w niektórych jednak przypadkach dla klasyfikacji binarnej finalna etykieta wyznaczana jest jako:

$$\hat{y} = \arg_{l > t} \hat{y}^{(l)} = \{2\}$$

- etykieta, której prawdopodobieństwo jest większe od pewnego progu t , najczęściej ustawionego na wartość 0.5.

Klasyfikacja binarna

- Klasa pozytywna (P) jest to ta klasa, która nas najbardziej interesuje.
- Najczęściej jest niedoreprezentowana, tj. 5%-95%.

		Przewidywane	
		P (2)	N (1)
Prawdziwe	P (2)	TP	FN
	N (1)	FP	TN

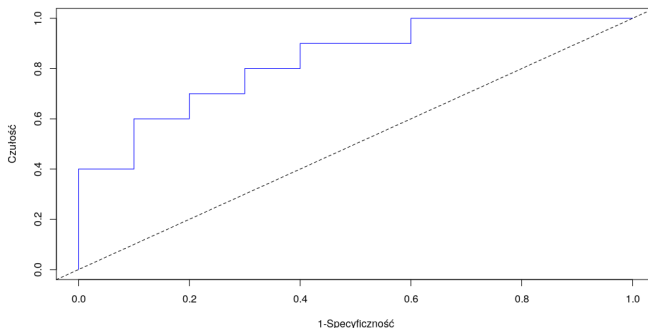
$$\text{Czułość} = \frac{TP}{TP + FN}$$

$$\text{Specyficzność} = \frac{TN}{TN + FP}$$

$$\text{Jakość} = \frac{TN + TP}{TN + TP + FP + FN}$$

Klasyfikacja binarna

- Krzywa ROC (ang. Receiver Operating Characteristic):
graficzna reprezentacja efektywności modelu poprzez
wykreślenie miar *Czułości* i *Specyficzności* powstałych z
modelu przy zastosowaniu wielu różnych punktów odcięcia t .



Pole pod krzywą ROC

- AUC (ang. Area Under Curve) - wielkość pola pod krzywą ROC mieści się w przedziale $[0; 1]$.

Interpretacja jak pole

$$AUC = \int_0^1 ROC(t) dt$$

Interpretacja jako prawdopodobieństwo

$$AUC = P(\hat{y}^{(2)} > \hat{y}^{(1)})$$

Metody estymacji AUC

- Istnieje kilka różnych sposobów oszacowania tej miary.
- Obserwacje muszą być posortowane nierosnąco względem $\hat{y}^{(2)}$.

Interpretacja jak pole

$$AUC = -\frac{1}{2} \sum_{i=2}^n (Czułłość_{i-1} Specyficzność_i - Czułłość_i Specyficzność_{i-1})$$

Interpretacja jako prawdopodobieństwo

$$AUC = \frac{R_2 - \frac{n_2(n_2-1)}{2}}{n_2 n_1}$$

gdzie: n_2 - liczba obserwacji z klasy 2, n_1 - liczba obserwacji z klasy 1, R_2 - suma rang obserwacji z klasy 2.

Klasyfikacja wieloklasowa

		Przewidywane					
		1	2	...	j	...	k
Prawdziwe	1	T	F	...	F	...	F
	2	F	T	...	F	...	F

	j	F	F	...	T	...	F

	k	F	F	...	F	...	T

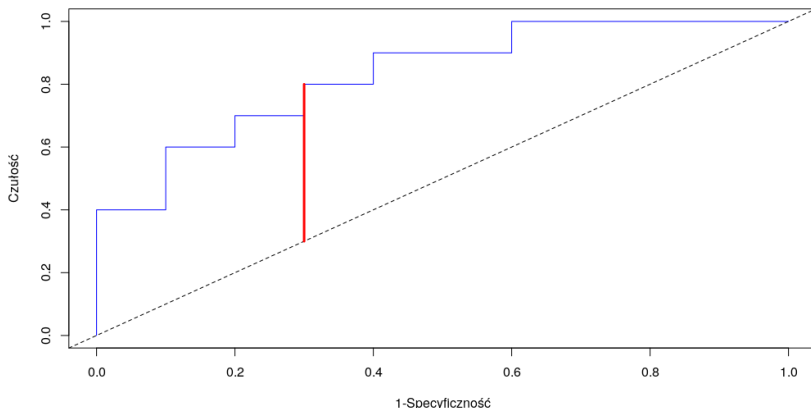
$$Jakość = \frac{1}{n} \sum_{i=1}^n I(y_i = \arg \max_l \hat{y}_i^{(l)})$$

$$multiAUC = \frac{1}{k(k-1)} \sum_{l,k} AUC_{l,k}$$

Youden index (J)

- Wyznaczenie optymalnej wartości t wyznacza się na podstawie:

$$J = \arg_t \text{Czułość} + \text{Specyficzność} - 1$$



Section 4

Sprawdzian krzyżowy

- Sprawdzian krzyżowy (walidacja krzyżowa, kroswalidacja, sprawdzanie krzyżowe) – metoda statystyczna, polegająca na podziale próby statystycznej na podzbiory, a następnie przeprowadzaniu wszelkich analiz na niektórych z nich (zbiór uczący), podczas gdy pozostałe służą do potwierdzenia wiarygodności jej wyników (zbiór testowy, zbiór walidacyjny).

Prosta walidacja

- Najbardziej typowy rodzaj walidacji, w którym próbę dzieli się losowo na rozłączne zbiory:
 - Uczący i testowy najczęściej w proporcjach 80%-20% lub 70%-30%.
 - Uczący, walidacyjny i testowy najczęściej w proporcjach 60%-30%-10%.

k-krotna walidacja

- Oryginalna próba jest dzielona na k podzbiorów, gdzie każdy z nich bierze się jako zbiór testowy, a pozostałe razem jako zbiór uczący. Analiza jest więc wykonywana k razy. K rezultatów jest następnie uśrednianych (lub łączonych w inny sposób) w celu uzyskania jednego wyniku.
 - Proporcja zbiorów wynosi $(1-1/k)\%-(1/k)\%$, np. dla $k = 4$, 75%-25%; $k = 10$, 90%-10%.

Kroswalidacja stratyfikowana

- Nie jest to w zasadzie osobna odmiana kroswalidacji, a odnosi się do wszystkich jej rodzajów wymienionych powyżej.
- Kroswalidacja stratyfikowana (ang. stratified cross-validation) polega na takim podziale obiektów pomiędzy zbiór uczący i zbiór testowy, aby zachowane były oryginalne proporcje pomiędzy klasami decyzyjnymi.
- Zastosowanie kroswalidacji stratyfikowanej jest szczególnie ważne w przypadku, gdy w oryginalnym zbiorze danych występują znaczne dysproporcje w liczebności przykładów należących do poszczególnych klas decyzyjnych.

Section 5

Optymalizacja parametrów

- Modele uczenia maszynowego są sparametryzowane, aby ich zachowanie można było dostosować do danego problemu.
- Modele mogą mieć wiele parametrów, a znalezienie najlepszej kombinacji parametrów może być traktowane jako problem wyszukiwania.
- Proces ten nazywany jest optymalizacją hiperparametrów, gdzie parametry algorytmu są określane jako hiperparametry, podczas gdy współczynniki znalezione przez sam algorytm uczenia maszynowego są określane jako parametry.
- Optymalizacja wskazuje na poszukiwawczy charakter problemu.
- Określając ten proces jako problem wyszukiwania, można użyć różnych strategii wyszukiwania, aby znaleźć dobry i niezawodny parametr lub zestaw parametrów dla algorytmu dla danego problemu.

Dostrajanie parametrów poprzez przeszukiwanie siatki (ang. grid search)

- Przeszukiwanie siatki to podejście do strojenia parametrów, które metodycznie buduje i ocenia model dla każdej kombinacji parametrów algorytmu określonych w siatce.

Losowe dostrajanie parametrów (ang. random search)

- Wyszukiwanie losowe to podejście do dostrajania parametrów, które będzie próbkować parametry algorytmu z losowego rozkładu jednostajny) dla ustalonej liczby iteracji.
- Model jest konstruowany i oceniany dla każdej kombinacji wybranych parametrów.

Section 6

Literatura

- *Multicore Data Science with 'R' and Python*
- *Beyond Single-Core R* by Jonoathan Dursi (also see [GitHub repo](#) for slide source)
- *The venerable Parallel 'R'* by McCallum and Weston (a bit dated on the tooling, but conceptually solid)
- <http://www.bytemining.com/2010/08/taking-r-to-the-limit-part-ii-large-datasets-in-r/>
- Adler, D., Nenadic, O., Zucchini, W., & Glaser, C. (2007). *The "ff" package: Handling Large Data Sets in 'R' with Memory Mapped Pages of Binary Flat Files*. Institute for Statistics and Econometrics.
- Emerson, J. W., & Kane, M. J. (2009). *The 'R' Package bigmemory: Supporting Efficient Computation and Concurrent Programming with Large Data Sets*. Journal of Statistical Software, 10.