Příklad ke zkoušce #91 – Medián

Úvod do programování

Martin Řanda

Úvod

Tento dokument slouží jako technická zpráva k programům `stat_median.py` a `data_manip.py`.

První z výše zmíněných programů se zabývá hledáním mediánu nesetříděné posloupnosti reálných čísel. Soubor `data_manip.py` potom obsahuje podpůrné funkce k načtení posloupnosti čísel a k exportu výsledku. Z toho důvodu mu není věnováno tolik pozornosti.

Oba programy byly otestovány a shledány funkčními na verzích 3.8 a 3.9 programovacího jazyka Python.

Zadání

Navrhněte program v jazyku Python, který z textového souboru načte nesetříděnou posloupnost reálných čísel tvořenou n prvky, spočte jejich medián a výsledek uloží zpět do textového souboru.

Definice problému a medián

Nejprve si formálně zadefinujme úlohu hledání mediánu v kontextu pravděpodobnostních rozdělení (dle Hudecová 2012). Považujme data $x_1, x_2, ..., x_n$ za realizaci náhodného výběru $X_1, X_2, ..., X_n$ z nějakého neznámého rozdělení, kde n je přirozené číslo.

Odhadem charakteristiky θ budeme rozumět funkci $\hat{\theta}_n$ pozorování $X_1, X_2, ..., X_n$. V našem případě nás bude zajímat odhad mediánu. Nejdříve je však potřeba nějakým způsobem náhodný výběr setřídit či seřadit, abychom byli schopni medián naleznout.

Náhodný výběr $X_1, X_2, ..., X_n$ považujeme za uspořádaný, pokud je jeho seznam hodnot seřazený vzestupně podle velikosti (tedy od nejmenšího pozorování po největší). Skutečnost, že je náhodný výběr uspořádaný označme následovně

$$X'_1, X'_2, ..., X'_n, \quad kde \quad X'_1 \leq X'_2 \leq ... \leq X'_n$$

Pro uspořádaný výběr pak medián odpovídá jeho prostřední hodnotě. Je však nutné rozlišit dva případy, pro které se bude způsob nalezení mediánu lišit, a to pro n liché a sudé.

$$m = \begin{cases} X_{(n+1)/2} & \textit{pro liché } n, \\ (X_{n/2} + X_{(n/2)+1})/2 & \textit{pro sudé } n. \end{cases}$$

Nyní lze úlohu analogicky převést do problematiky posloupností. Označme tedy neuspořádanou (nesetříděnou) posloupnost reálných čísel jako $\{Y_i\}_1^n$. Potom v situaci, kdy má setříděná posloupnost $\{Y_i'\}_1^n$ lichý počet prvků, je medián této posloupnosti roven hodnotě takového prvku, který se nachází *uprostřed* této posloupnosti – má tak stejný počet prvků na levé i pravé straně. V případě sudého počtu prvků posloupnosti se za medián označuje aritmetický průměr hodnot prvků na pozicích $n/2$ a $(n/2) + 1$.

Existující algoritmy

Nejjednodušší postup nalezení mediánu nesetříděné posloupnosti spočívá v tom, že je nejprve setříděna posloupnost a poté je na odpovídající pozici nalezena příslušná hodnota mediánu.

K výkonnějšímu způsobu patří tzv. *median-of-medians* algoritmus (Moore et al. 2016), který využívá strategii *divide and conquer*. Nejprve dojde k rozdělení nesetříděné posloupnosti na několik podposloupností a v každé z nich se určí přibližná hodnota mediánu. Výsledek pro každou podposloupnost je potom uložen do nové posloupnosti a je nalezen její medián. Tento nový odhad se potom používá jako *pivot* (podobně jako v algoritmu *quicksort*) a porovnává se s ostatními prvky posloupnosti. Pomocí rekurze je tato aproximace neustále vylepšována dokud není nalezena výsledná hodnota.

Popis zvoleného algoritmu

Budeme se zabývat prvním ze dvou zmíněných přístupů. Rozdělme si proto algoritmus hledání mediánu nesetříděné posloupnosti do 4 kroků. Předpokládejme, že posloupnost prvků máme již patřičným způsobem načtenou:

1. *Krok*
   Setřídit posloupnost prvků $\{Y_i\}_1^n$
2. *Krok*
   Naleznout počet prvků posloupnosti
3. *Krok*
   Vypočítat index prostředního prvku posloupnosti
4. *Krok*
   Naleznout hodnotu prvku s příslušným indexem – naleznout medián

Rozbor algoritmu a problematické situace

Chceme-li vytvořit postup, pomocí kterého v jazyku Python vypočítáme medián setříděné posloupnosti, musíme si nejdříve uvědomit několik skutečností.

V první řadě je nutné zmínit, že Python označuje první prvek posloupnosti indexem 0. Z toho vyplývá, že je potřeba přizpůsobit naši definici mediánu tomuto požadavku. Označme tedy výsledný index prvku posloupnosti písmenem j, pro který platí $j = (n + 1)/2$.

Pro posloupnost s lichým počtem prvků bude naším cílem získat vždy takový index k, pro který platí, že $k = j - 1$, což vyplývá právě ze způsobu, jakým Python indexuje. V sudém případě budeme potřebovat dva prvky s pozičními argumenty k a $k + 1$.

Index k můžeme naleznout například s využitím celočíselného dělení, pro které je v jazyku Python používán operátor "//" (floor division).

Uvažujeme-li n jakožto počet prvků dané posloupnosti nalezený v *Kroku 2*, potom platí, že prvek k je roven:

$$(n - 1) \;//\; 2$$

Pro ilustraci využití tohoto operátoru je níže uveden příklad.

```
Y = [5, 6, 7]
n = len(Y)
k = (n - 1) // 2

print(Y[k])
#> 6
```

Z příkladu výše je pak zřejmé, že pro setříděnou posloupnost s lichým počtem prvků je hodnota prvku na pozici k mediánem této posloupnosti.

Další problematická situace přímo vyplývá z definice mediánu, a to že existují dva různé případy, ve kterých se finální výpočet mediánu zásadně liší. V *Kroku 3* tedy rozlišujeme řešení pro lichý a sudý počet prvků posloupnosti.

Tento problém lze vyřešit několika různými způsoby. Například je možné využít operátor "%" (modulus), jehož výstupem je zbytek po dělení. Myšlenka je taková, že posloupnost se sudým počtem prvků bude mít vždy zbytek po dělení číslem 2 rovný nule, kdežto v případě lichého počtu prvků se zbytek po dělení bude vždy rovnat jedné.

Opět uvažujme n jakožto počet prvků dané posloupnosti nalezený v *Kroku 2*, potom platí:

$$n \,\%\, 2 = \begin{cases} 1 & \textit{pro lichá } n, \\ 0 & \textit{pro sudá } n. \end{cases}$$

Pokud bychom tento nález chtěli využít například v podmínce `if`, není nutné specifikovat, jestli se `n % 2` rovná jedné (pravda) nebo nule (nepravda), což vyplývá z vlastností datového typu boolean:

```
n = 3
if n % 2:
    print(n, "is odd")

#> 3 is odd
```

Pro sudé n je pak možné medián spočítat jako aritmetický průměr prvků na pozicích k a $k + 1$.

Vstupní a výstupní data

Z textového souboru načteme nesetříděnou posloupnost reálných čísel, která je tvořena n prvky. Je důležité, aby hodnoty byly od sebe nějakým způsobem odděleny, a to například s využitím zalomení řádku (line break) – každý prvek se nachází na samostatné lince textového souboru. Potom je možné jednoduše pomocí metody `readlines()` načíst jednotlivé řádky do seznamu. V repozitáři je k dispozici soubor `sequence.txt`, který obsahuje ilustrační vstupní data.

Výstupem programu je textový soubor, který obsahuje číselnou hodnotu mediánu dané posloupnosti.

Závěrečné shrnutí

V tomto dokumentu bylo popsáno, jakým způsobem lze naleznout medián nesetříděné posloupnosti reálných čísel.

Při implementaci algoritmu v programovacím jazyce Python je potřeba si uvědomit, že první prvek posloupnosti je označován indexem 0 a počet prvků je přirozené číslo n (tedy nikoliv $n - 1$). Dále je třeba úlohu vyřešit rozdílně pro posloupnost se sudým a lichým počtem prvků.

Soubor `stat_median.py` obsahuje konkrétní řešení tohoto příkladu a s pomocí programu `data_manip.py` jsou načítána a exportována relevantní data.

Seznam literatury

Hogg, R.V., McKean, J. and Craig, A.T., 2005. *Introduction to mathematical statistics.* Pearson Education.

Hudecová, Š., 2012. *Matematická statistika* [Online]. Dostupné z: https://www2.karlin.mff.cuni.cz/~hudecova/education/archive11/download/chem_predn/predn_slides_06.pdf [k 16.01.2021].

Larsen, R.J. & Marx, M.L., 2005. *An introduction to mathematical statistics.* Prentice Hall.

Moore, K. et al., 2016. *Median-finding Algorithm* [Online]. Brilliant.org. Dostupné z https://brilliant.org/wiki/median-finding-algorithm/ [k 11.01.2021].

Saha, A., 2015. *Doing Math with Python: Use Programming to Explore Algebra, Statistics, Calculus, and More!.* No Starch Press.