Projekt 2021 - část 1

předmět Zpracování a vizualizace dat v prostředí Python

Cílem projektu je získat, zpracovat a analyzovat data dostupná na internetu.

Orientační struktura projektu

- Část 1 (20 b)
  - stažení a zpracování dat
  - jednoduchá vizualizace
- Část 2 (20 b)
  - různé pohledy na data (např. dny, kdy se stává nejvíc nehod a podobně)
  - pokročilá vizualizace výsledků
  - zpracování závěrů (porozumění datům)
- Celkový projekt (60 b)
  - znázornění dat na mapě, operace nad těmito daty
  - korelace a predikce
  - automatické vytváření částí zpráv
  - spojení do analytické zprávy

V této části stáhnete volně dostupná data o nehodách v ČR a jejich následcích a budete je zpracovávat. V dalších částech využijete kód z této první části.

Získání a předzpracování dat (20 bodů)

Z URL adresy https://ehw.fit.vutbr.cz/izv/ stáhněte Statistiky nehodovosti Policie ČR pro jednotlivé roky. Seznamte se se strukturou datových položek v dokumentu na stránkách (pozor na duplicitní data v jednotlivých měsících!). Vytvořte jeden souhrnný dataset agregující data přes všechny roky a vybrané kraje. Při zpracování surových dat dejte pozor na časové údaje a neplatné hodnoty (např. minuty rovny 60 znamenají neznámou minutu; hodiny rovny 25 znamenají úplně neznámý čas nehody).

Nad očištěnými vstupními daty vytvořte skript, který vygeneruje statistiku počtu nehod podle určení přednosti jízdě v jednotlivých krajích (v grafický výstup).

Vycházejte z šablon souborů v dokumentovém skladu ve WISu.

Ve všech dalších úkolech budete pracovat s skriptem pro stahování a zpracování dat z tohoto úkolu *(pokud byla jeho implementace chybná, bude možné ji nahradit; chyba v prvním úkolu samozřejmě neznamená neúspěch v dalších částech).*

Technické řešení

Vytvořte soubor `download.py`, který bude obsahovat třídu `DataDownloader`. Tato třída bude implementovat následující metody (můžete samozřejmě přidat další vlastní, případně můžete přidat další parametry do funkcí, pokud to uznáte za vhodné):

- `__init__(self, url="https://ehw.fit.vutbr.cz/izv/", folder="data", cache_filename="data_{}.pkl.gz")`
  inicializátor - obsahuje volitelné parametry:
  - `url` - ukazuje, z jaké adresy se data načítají. Defaultně bude nastavený na výše uvedenou URL.
  - `folder` - říká, kam se mají dočasná data ukládat. Tato složka nemusí na začátku existovat!
  - `cache_filename` - jméno souboru ve specifikované složce, které říká, kam se soubor s již zpracovanými daty z funkce `get_dict` bude ukládat a odkud se budou data brát pro další zpracování a nebylo nutné neustále stahovat data přímo z webu. Složené závorky (formátovací řetězec) bude nahrazený tříznakovým kódem (viz tabulka níže) příslušného kraje. Pro jednoduchost podporujte pouze formát "pickle" s kompresí gzip.
- `download_data(self)`
  metoda stáhne do datové složky `folder` všechny soubory s daty z adresy `url` definované v inicializátoru. Je nutné načíst adresy ZIP souborů z HTML souboru z uvedené URL. Není dovoleno přímo odhadovat názvy ZIP souborů, mít adresy uložené přímo v kódu atd,
- `parse_region_data(self, region)`
  pokud nejsou data pro daný kraj již stažena ve složce `folder`, stáhnou se voláním `download_data`. Poté se pro daný region specifikovaný tříznakovým kódem (viz tabulka níže) provede **vždy** (tj. data nejsou cachována) **extrakce dat** do slovníku (`dict`), kde klíčem bude řetězec odpovídající kódu sloupce (např. p1, p13b, p51, ..) a hodnotou pole NumpyArray, tzn. dle následujícího schematu

  `dict(str: np.ndarray)`

  Počet položek (klíčů) tohoto slovníku bude odpovídat počtu hlaviček plus navíc jeden **nový klíč "**region", který bude obsahovat NumPy pole, ve kterém se bude opakovat tříznaký kód kraje.

  Musí platit to, že všechny hodnoty (tj. NumPy pole) ve slovníku mají stejný rozměr (včetně regionu). Pro každý sloupec zvolte **vhodný datový typ** (t.j. pokud je to možné, snažte se vyhnout textovým řetězcům a použít numerické typy, vyřešte desetinnou čárku, nepoužívejte PyObject atp.). Při otevírání souboru nezapomeňte specifikovat kódování: `open(f, 'r', encoding="cp1250")`
- `get_dict(self, regions = None)`
  Vrací zpracovaná data pro vybrané kraje (regiony). Argument `regions` umožňuje specifikovat kraje, pro které chceme výsledek vrátit. Jedná se o seznam (list) obsahující třípísmenné kódy. Pokud je použito None nebo je seznam prázdný, zpracují se všechny kraje včetně města Prahy. Výstupem je složení výstupů funkce `parse_region_data` pro všechny kraje do jednoho asociativního pole (klíčem bude řetězec hlavičky např. p1, p15, region,... a hodnotami bude vždy `np.ndarray` - všechny o stejném rozměru).

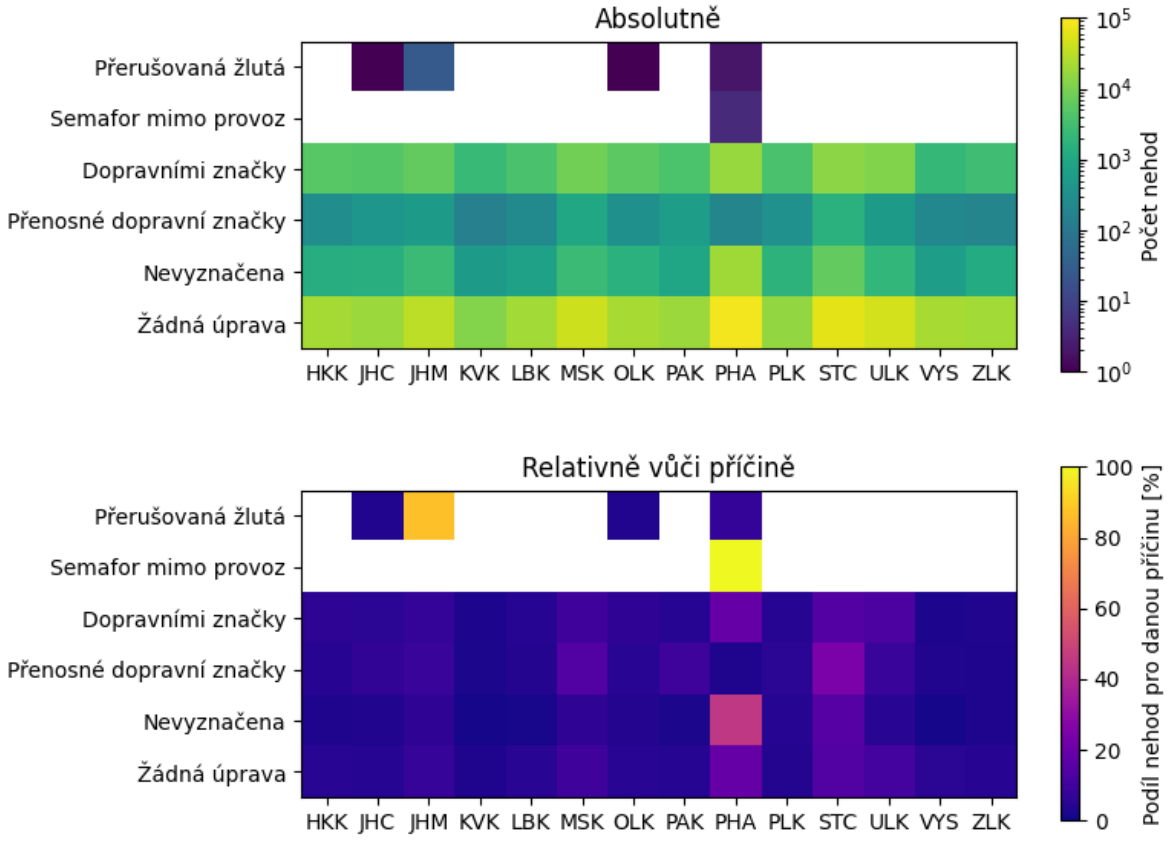
  Metoda pracuje nad extrahovanými daty, které získá voláním metody `parse_region_data` Abychom co nejvíce zefektivnili manipulaci s daty a vyhnuli se opakovanému zpracování data, budou **extrahovaná data uchovávána v paměti**

(v zvoleném atributu instance třídy) a současně ukládána do pomocného cache souboru pomocí následujícího schématu:

- pokud už je výsledek načtený v paměti (tj. dostupný ve vámi zvoleném atributu), vrátí tuto dočasnou kopii
- pokud není uložený v paměti, ale je již zpracovaný v cache souboru, tak načte výsledek z cache, uloží jej do atributu a vrátí.
- jinak se zavolá funkce `parse_region_data`, výsledek volání se uloží do cache, poté do paměti a výsledek vrátí

Pokud bude skript spuštěný přímo (tj. nebude importovaný jako modul)[1], stáhněte data pro 3 vámi vybrané kraje (s využitím funkce `get_dict`) a vypište do konzole základní informace o stažených datech (jaké jsou sloupce, počet záznamů a jaké kraje jsou v datasetu).

Dále vytvořte soubor `get_stat.py`, který bude obsahovat funkci `plot_stat(data_source, fig_location = None, show_figure = False)` která vezme zpracovaná data z vašeho `DataDownloader` (při volání použijete `data_source = DataDownloader().get_dict()`), načte data a vizualizuje počty nehod jako je na následujícím příkladu (vaše řešení může vypadat jinak!)



- Graf vizualizuje dva pohledy na počet nehod v jednotlivých krajích podle *Místní úpravy přednosti v jízdě* (sloupec *p24*)
  - matice absolutního počtu nehod v logaritmickém měřítku

[1] Při spuštění `python3 download.py` (lze testovat pomocí podmínky `__name__=="__main__"`)

    - relativní počet nehod v lineárním měřítku (v procentech) dle úpravy přednosti, kdy součet hodnot v každém řádku (přes příčinu) musí dát 100%
- Využívejte maximálně funkce NumPy- iterování přes všechna data je neefektivní a povede ke ztrátě bodového hodnocení. Většinu výpočtů je možné realizovat přímým voláním funkcí z knihovny NumPy.
- Kombinace *místní úpravy přednosti v jízdě* a *kraje*, kde nedošlo k žádné nehodě, zobrazte bílou barvou.
- Graf by měl splňovat všechny náležitosti, které u grafu očekáváme, měl by být přehledný a jeho velikost by měla být taková, aby se dal čitelně použít v šířce A4 (t.j. cca 18 cm). Toto omezení není úplně striktní, ale negenerujte grafy, které by byly přes celý monitor.
- Pokud bude nastavený "`fig_location`", tak se do dané cesty uloží obrázek. Cesta může být např. `statistika.png` nebo `fig/stat.png`. Pokud cesta obsahuje i složku, zkontrolujte, zda odpovídající složka existuje a případně ji vytvořte (u *fig/res/stat.png* vytvořte adresáře *fig* a *fig/res*).
- Pokud je nastavený parametr "`show_figure`", tak se graf zobrazí v okně pomocí volání `plt.show()`.
- Při kreslení používejte přímo knihovnu **`matplotlib`**, ne `seaborn` či funkci `pandas.DataFrame.plot()`!

Pokud bude skript spuštěný jako hlavní (tj. nebude importovaný jako modul), pomocí modulu *argparse* zpracujte argumenty příkazové řádku. Definujte volitelné parametry *--fig_location* a *--show_figure* a umožněte spuštění kódu z příkazové řádky.
Pozor - tento skript bude využívat vaši implementaci v souboru `download.py`. Aby bylo možné hodnotit tuto podčást, je tedy nutné mít funkční i stahování dat! Pro zpracování dat není na této úrovni dovoleno použít pokročilých knihoven jako je Pandas. Kromě vestavěných knihoven (os, sys, re, gzip, pickle, csv, zipfile…) byste si měli vystačit s:, numpy, matplotlib, BeautifulSoup, requests . Další knihovny je možné použít po schválení opravujícím (např ve fóru WIS).

Dokumentace všech částí (souborů, tříd a funkcí) bude přímo v odevzdaných souborech. Snažte se dodržovat konvenci PEP 257 [https://www.python.org/dev/peps/pep-0257] a PEP 8 [https://www.python.org/dev/peps/pep-0008/]. Pozor, Python má přímo definovaný kódovací styl (narozdíl od C), a proto kontrola bude součástí hodnocení.

Kódy krajů

Ve svém kódu využívejte následující kódy (bez diakritiky, s velkými písmeny) pro identifikaci krajů.

Jméno	Kód dle ČSÚ
Hlavní město Praha	PHA
Středočeský	STC
Jihočeský	JHC

Plzeňský	PLK
Karlovarský	KVK
Ústecký	ULK
Liberecký	LBK
Královéhradecký	HKK
Pardubický	PAK
Olomoucký	OLK
Moravskoslezský	MSK
Jihomoravský	JHM
Zlínský	ZLK
Kraj Vysočina	VYS

Odevzdávání a hodnocení

Do 12. 11. 2021 odevzdejte dva samostatné soubory. Pro práci používejte
Pro práci s projektem je možné využít všechny knihovny představené při přednáškách. Skripty musí všechna pomocná data ukládat do složky "./data/" (viz parametr folder), kterou si případně vytvoří.

Hodnotit se bude zejména:

- správnost výsledků
- vizuální dojem z grafů
- kvalita kódu
    - efektivitu implementace a reprezentace (i rychlost v porovnání s ostatními řešeními), využívání efektivních funkcí (např. NumPy)
    - přehlednost kódu
    - správné ukládání mezivýsledků
    - dodržení standardů a zvyklostí pro práci s jazykem Python - dokumentační řetězce, splnění PEP8
    - dokumentace funkcí a souborů
    - znovupoužitelnost kódů - správná izolace potřebných částí do funkcí

Celkem za první část můžete získat až 20 bodů, přičemž je k zápočtu nutné získat z této části minimálně 2 body.

Dotazy a připomínky

Na fóru WIS případně na mailu mrazek@fit.vutbr.cz.