Druhá část projektu - základní analýza dat

Tato část projektu navazuje na první část. Cílem je implementovat základní analýzu dat (Statistika nehodovosti Policie ČR), která máme stažena a předzpracována. Řešení se skládá ze 4 úkolů, přičemž každý úkol bude jedna samostatná funkce implementovaná v jazyce Python.

Vstupní data

Aby byly stejné podmínky pro všechny a nemuseli jste opravovat první část projektu, budeme na základě vašich požadavků v dotazníku k předmětu **pracovat se souborem accidents.pkl.gz**, který stáhnete z adresy: http://ehw.fit.vutbr.cz/izv/accidents.pkl.gz (stahování souboru není součástí tohoto úkolu, není třeba proto implementovat)

Obsah souboru byl získán následovně:

```
dw = DataDownloader()
df = pd.DataFrame(dict(zip(*dw.parse_data())))
df.to_pickle("accidents.pkl.gz")
```

Serializovaný DataFrame obsahuje data organizovaná ve sloupích, které jsou pojmenované tak, jak jsou uvedeny v popisu datového souboru (*p1*, *p13a*, …). Data jsou uložena jako `float`, `int` nebo `str` podle sloupce. Neznámé hodnoty jsou u typů float reprezentované jako `np.nan`, u int hodnotou `-1` a v případě řetězců jako prázdný řetězec.

Požadovaný výstup

Cílem je vytvořit kód, který vizualizuje tři různé závislosti v datech. Veškeré kódy budou součástí jednoho souboru `analysis.py`, jehož **kostru naleznete v souborovém skladu** ve WIS. Předpokládá se, že budete primárně pracovat s knihovnami Pandas a Seaborn + je dovolené využít všechny knihovny zmiňované během přednášek.

Odevzdávání a hodnocení

Soubor `analysis.py` odevzdejte do 9. 12. 2020. Hodnotit se bude zejména:

- správnost výsledků
- vizuální zpracování grafů
- kvalita kódu
    - efektivita implementace (nebude hodnocena rychlost, ale bude kontrolováno, zda nějakým způsobem řádově nezvyšujete složitost)
    - přehlednost kódu
    - dodržení standardů a zvyklostí pro práci s jazykem Python (PEP8)
    - dokumentace kódu

Celkem lze získat až 20 bodů, přičemž k zápočtu je nutné získat z této části minimálně 2 body.

Zpracování a vizualizace dat v prostředí Python 2020

Úkol 1: Příprava dat (až 5 bodů)

Signatura funkce, která bude odpovídat tomuto úkolu:

```
def get_dataframe(filename : str = "accidents.pkl.gz",
                  verbose : bool = False
                  ) -> pd.DataFrame:
```

Funkcionalita

- Funkce načte lokálně uložený soubor se statistikou nehod (odpovídající http://ehw.fit.vutbr.cz/izv/accidents.pkl.gz), jehož umístění je specifikováno argumentem filename.
- Funkce vytvoří v DataFrame sloupec *date*, který bude ve formátu pro reprezentaci data (berte v potaz pouze datum, t.j. sloupec *p2a*)
- S výjimkou sloupce region reprezentujte vhodné sloupce pomocí kategorického datového typu. Měli byste se dostat pod velikost v paměti 0.5 GB. Sloupec *region* je vhodné ponechat v původní podobě pro lepší práci s figure-level funkcemi Seaborn.
- Při povoleném výpisu (`verbose == True`) spočítejte kompletní (hlubokou) velikost všech sloupců v datovém rámci před a po vaší úpravě a vypište na standardní výstup pomocí funkce print následující 2 řádky
  `orig_size=X MB`
  `new_size=X MB`
  Čísla vypisujte na 1 desetinné místo a počítejte, že 1 MB = 1 048 576 B.

Upozornění: Další skripty budou využívat vaši implementaci parsování datového rámce. Bez této implementace není možné hodnotit další úkoly**.**

Zpracování a vizualizace dat v prostředí Python 2020

Úkol 2: následky nehod v jednotlivých regionech (až 5 bodů)

Signatura funkce, která bude odpovídat tomuto úkolu:

```
def plot_conseq(df: pd.DataFrame, fig_location: str = None,
                show_figure: bool = False):
```

Funkcionalita

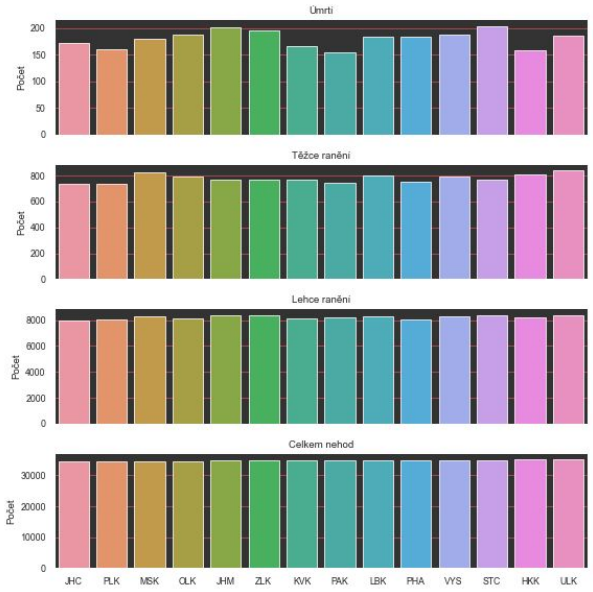
Vytvořte graf následků nehod v jednotlivých regionech, který uložte do souboru specifikovaného argumentem `fig_location` a případně zobrazte pokud `show_figure` je `True`. Argument `df` odpovídá DataFrame jenž je výstupem funkce `get_dataframe`. Graf se bude skládat z 4 podgrafů organizovaných do matice tvořené jedním sloupcem a 4 řádky.

Požadavky:

1) Protože se jedná o kategorická data, budou všechny grafy sloupcového typu s kraji na ose X.
2) Na ose Y bude postupně pro jednotlivé podgrafy uveden:
   a) počet lidí, kteří zemřeli při nehodě (*p13a*),
   b) počet lidí, kteří byli těžce zranění (*p13b*)
   c) počet lidí, kteří byli lehce zranění (*p13c*)
   d) celkový počet nehod v daném kraji
3) Kraje na ose X seřaďte podle celkového počtu nehod (argument `order`), přičemž kraj s největší hodnotou bude vlevo.
4) Graf upravte tak, aby popisky na osách, popisek druhu následku atd. dával smysl. Dle zásad dobré vizualizace (viz přednáška 5) zvolte vhodnou barvu a vhodný styl. Podgrafu nastavte vlastní pozadí.

Tip: nejdříve budete muset agregovat hodnoty ve sloupcích a až poté je vykreslovat. Hodnoty *p13a-c* je nutné agregovat sumou, pro počet nehod pomůže agregace `count` některého sloupce. Pomůže vám také převod na tzv. stacked formát Pandas (`pd.melt`).

Příklad výstupu: (použita náhodná data a záměrně zcela nevhodná grafická forma)



Zpracování a vizualizace dat v prostředí Python 2020

Úkol 3: Příčina nehody a škoda (až 5 bodů)

Signatura funkce, která bude odpovídat tomuto úkolu:

```
def plot_damage(df: pd.DataFrame, fig_location: str = None,
                show_figure: bool = False):
```

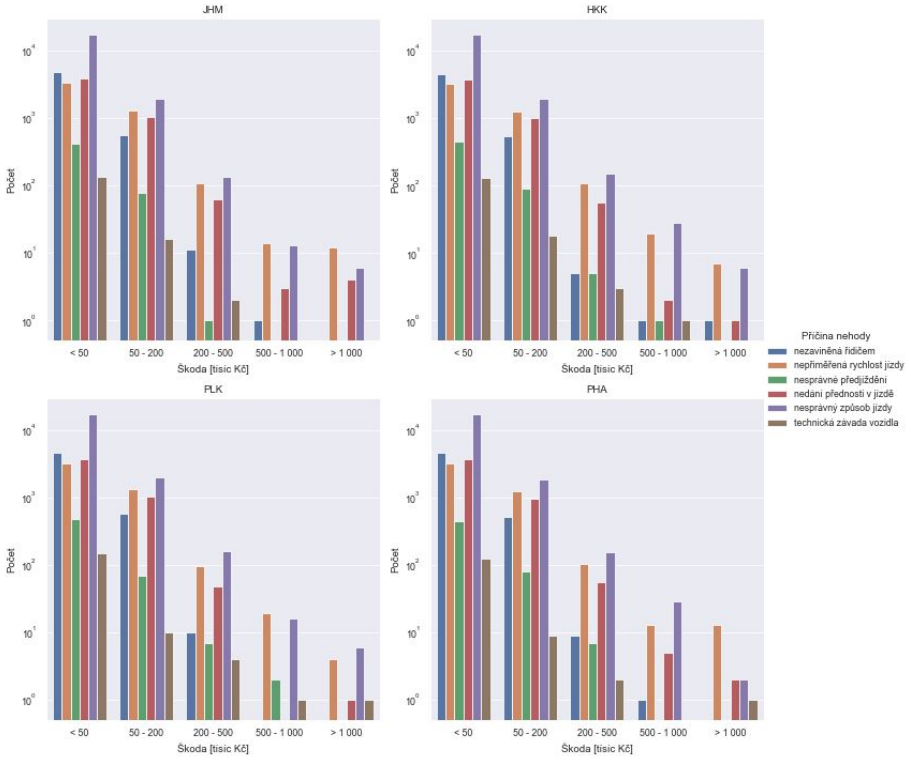
Funkcionalita

Pro čtyři vámi vybrané kraje znázorněte počet nehod v závislosti na škodě na vozidlech (*p53*) uvedené v tisících Kč, která bude rozdělena do několika tříd. Počty vizualizujte zvlášť podle hlavní příčiny nehody (*p12*).

Požadavky:

1) Příčinu nehody rozdělte do tříd (vč. pojmenování) jak je uvedeno v popisu datových položek na řádcích 62 - 67 popisu položek (funkce `pd.cut`).
2) Škodu v tis. Kč rozdělte do pěti tříd: < 50, 50 - 200, 200 - 500, 50 - 1000, >1000.
3) Kvůli značným rozdílům v datech použijte logaritmické měřítko osy Y.
4) Graf upravte tak, aby popisky na osách, popisek druhu následku atd. dával smysl.

Příklad výstupu: (použita náhodná data)



Zpracování a vizualizace dat v prostředí Python 2020

Úkol 4: Stav vozovky v jednotlivých měsících (až 5 bodů)

Signatura funkce, která bude odpovídat tomuto úkolu:

```
def plot_surface(df: pd.DataFrame, fig_location: str = None,
                 show_figure: bool = False):
```

Funkcionalita

Pro čtyři vámi vybrané kraje pro různé měsíce vykreslete čárový graf, který bude zobrazovat pro jednotlivé měsíce (osa X- sloupec *date*) počet nehod při různém stavu povrchu vozovky (*p16*).

Doporučený postup

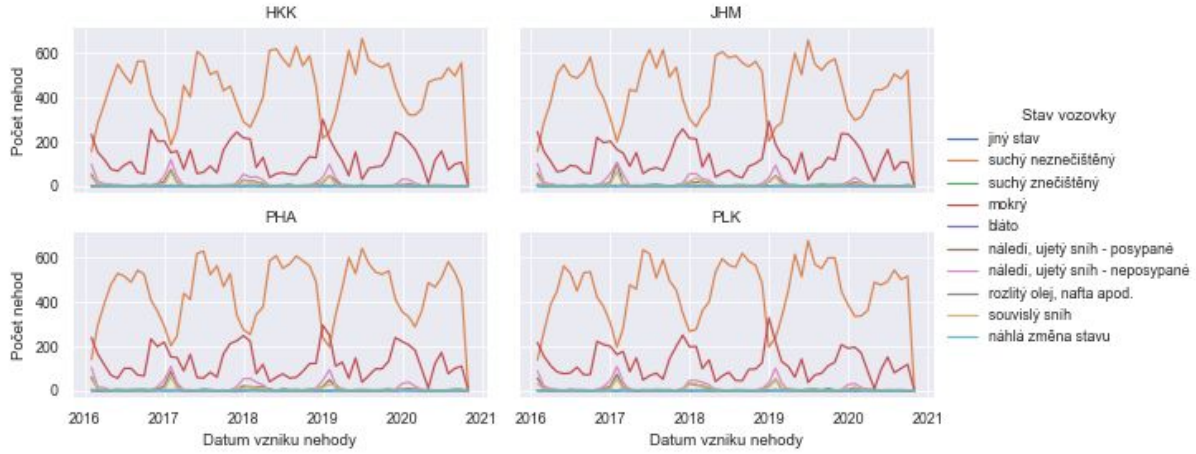
1. Vytvořte si pomocný DataFrame s potřebnými řádky a potřebnými sloupci

2. (Varianta 1) Změňte velikost tabulky pomocí funkce `pivot_table` tak, že budete brát sloupec *p16* jako zdroj pojmenování a sloupec, který je ve všech záznamech (např. *p1*) jako hodnotu (agregujte počtem)

   (Varianta 2) Vytvořte kontingenční tabulku (`pd.crosstab`) s indexy *region* a *date* a sloupci *p16*

3. Přejmenujte sloupce 0-9 na vhodné pojmenování podle řádků 151 - 160 popisu položek (funkce `pd.DataFrame.rename`)
4. Proveďte agregaci dat podle regionů, podvzorkujte na měsíce a převeďte správně na stacked formát.
5. Vykreslete patřičné grafy upravené tak, aby popisky na osách, popisek druhu následku atd. dával smysl.

Příklad výstupu: (použita náhodná data)



Poznámky k implementaci

Soubor, který vytvoříte, bude při hodnocení importovaný a budou volány jednotlivé funkce. Mimo tyto funkce, část importů a dokumentační řetězce nepište žádný funkční kód. Blok na konci souboru ohraničený podmínkou

```
if __name__ == "__main__":
    pass
```

naopak můžete upravit libovolně pro testovací účely. Dále můžete přidat další funkce (pokud budete potřebovat), pro názvy těchto funkcí použijte prefix "_".

Stručnou dokumentaci všech částí (souboru a funkcí) uveďte přímo v odevzdaných souborech. Respektuje konvenci pro formátování kódu PEP 257 [PEP 257 -- Docstring Conventions] a PEP 8 [PEP 8 -- Style Guide for Python Code].

Grafy by měly 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.

Grafy v zadání jsou pouze ukázkové. Data byla randomizována a vaše výsledky budou vypadat jinak. Není nutné ani chtěné, aby grafy vizuálně vypadaly stejně - vlastní invence směrem k větší přehlednosti a hezčímu vzhledu se cení! Co není přímo specifikováno v zadání můžete vyřešit podle sebe.

Dotazy a připomínky

Dotazy a připomínky směrujte na fórum ve WIS případně na mail mrazek@fit.vutbr.cz.