Gymnázium Dr. Emila Holuba

Použití knihovny GeoPandas v Pythonu

Seminární práce

Autor: **Stanislav Kozák**
Studijní obor: Zeměpisný seminář
Vedoucí práce: Mgr. Marek Janů

Gymnázium Dr. Emila Holuba

Zadání seminární práce

Autor: **Stanislav Kozák**

Studijní obor: Zeměpisný seminář

Vedoucí práce: Mgr. Marek Janů

Název práce: Použití knihovny GeoPandas v Pythonu

Název práce v AJ: Application of GeoPandas Library in Python

Cíl práce: Představit možnosti knihovny GeoPandas

Vedoucí práce: Mgr. Marek Janů

Datum zadání práce: 1. 9. 2020

Datum odevzdání práce: 15. 4. 2021

Poběžovice u Holic Duben 2021

Čestné prohlášení

Prohlašuji, že jsem seminární práci vypracoval samotně a že jsem uvedl všechny použité prameny a literaturu.

V Poběžovicích, dne 15. 4. 2021. Stanislav Kozák

Poděkování

Rád bych poděkoval Mgr. Markovi Janů za odborné vedení a Mgr. Janu Češíkovi za cenné rady při zpracování mé seminární práce.

Anotace

KOZÁK, Stanislav. *Použití knihovny GeoPandas v Pythonu*. Holice: Gymnázium Dr. Emila Holuba, 2021. 36 s.

Tato seminární práce představuje klíčové koncepty a funkce geografické knihovny GeoPandas v Pythonu. Obsahuje také ukázky použití GeoPandas. Okomentovaný zdrojový kód k těmto ukázkám je přístupný online ve formě Jupyter Notebooku.

Klíčová slova:

Python, GeoPandas, geografický informační systém

Annotation

KOZÁK, Stanislav. *Application of GeoPandas Library in Python*. Holice: Gymnazium Dr. Emila Holuba, 2021. 36 p.

This seminary thesis introduces key concepts and functions of the GeoPandas library in Python. It also contains examples of this library's usage. Source code for these examples with comments is available online as Jupyter Notebook files.

Keywords:

Python, GeoPandas, geographic information system

Obsah

Úvod

V dnešní době je orientace v ohromné záplavě informací klíčová dovednost, kterou by si měl osvojit každý jedinec. Snaha přiblížit se k objektivnímu popisu skutečnosti vede v současnosti k vyšší popularitě datové vědy. Někdo si v této souvislosti vybaví vzorce a tabulky v Excelu, to ale zdaleka není jediný nástroj, který se pro datovou analýzu může používat.

Cílem této práce je představit funkce a možnosti použití GeoPandas, což je knihovna programovacího jazyka Python. Je určená pro manipulaci s geometrickými daty a jejich vizualizaci na mapě. Toto téma jsem si zvolil proto, že mi umožnilo propojit můj zájem o programování v jazyce Python se zájmem o různé metody vizualizace dat, například právě pomocí map.

Kromě popisu funkcí knihovny GeoPandas bych se chtěl v této práci stručně zabývat knihovnou Pandas, z které GeoPandas vychází. Dále bych chtěl popsat různé možnosti tvorby vizualizací pomocí různých dalších knihoven. Na toto naváže praktická část, ve které vytvořím několik malých map, abych ukázal použití GeoPandas v praxi. Součástí této práce bude i postup instalace použitých technologií (Příloha 1) a zdrojový kód k praktické části a k ukázkám z ostatních kapitol (Příloha 2).

1 Python

Python je open-source programovací jazyk vytvořený na začátku 90. let 20. století Guidem van Rossumem. [1] Jedná se o interpretovaný jazyk, a před spuštěním se tak nekompiluje (na rozdíl od mnoha jiných jazyků). Zároveň je dynamicky typovaný, při tvorbě proměnných se proto předem nemusí určovat jejich typ. Podporuje použití objektově orientovaného, procedurálního i funkcionálního programovacího paradigmatu. Má poměrně jednoduchou syntaxi, na rozdíl od ostatních jazyků například nepoužívá k uvození bloků kódu závorky, ale odsazení. Díky těmto vlastnostem je vhodný i pro začátečníky. [2]

V posledních letech zaznamenává tento jazyk poměrně velký nárůst popularity, využívá ho například Google, Instagram, Spotify nebo Netflix. [3][4] Mimo jiné se používá pro datovou analýzu a další vědecké účely (společně s jazyky R, MATLAB a další). [5] Samotný jazyk je zdarma ke stažení na webových stránkách https://www.python.org/downloads/. V minulosti byly paralelně vyvíjeny verze Python 2 a Python 3. V dubnu 2020 ale s vydáním verze 2.7.18 přestal být Python 2 podporován. V době psaní této práce je poslední vydaná verze 3.9.0. [6]

1.1 Vybrané knihovny jazyka Python

Knihovna je v programování označení pro ucelený soubor zdrojového kódu, který je určen pro opětovné použití. [7] Python disponuje rozsáhlou standardní knihovnou (ta obsahuje knihovny, které se instalují společně s Pythonem). Navíc existuje online repozitář PyPI (Python Package Index), který umožňuje instalaci dalších knihoven. Tyto knihovny lze stahovat pomocí správce *pip*, který se instaluje společně s Pythonem. [8] Níže krátce představím tři hlavní knihovny, které využívá knihovna GeoPandas, a budou tedy zmíněny i v dalších částech.

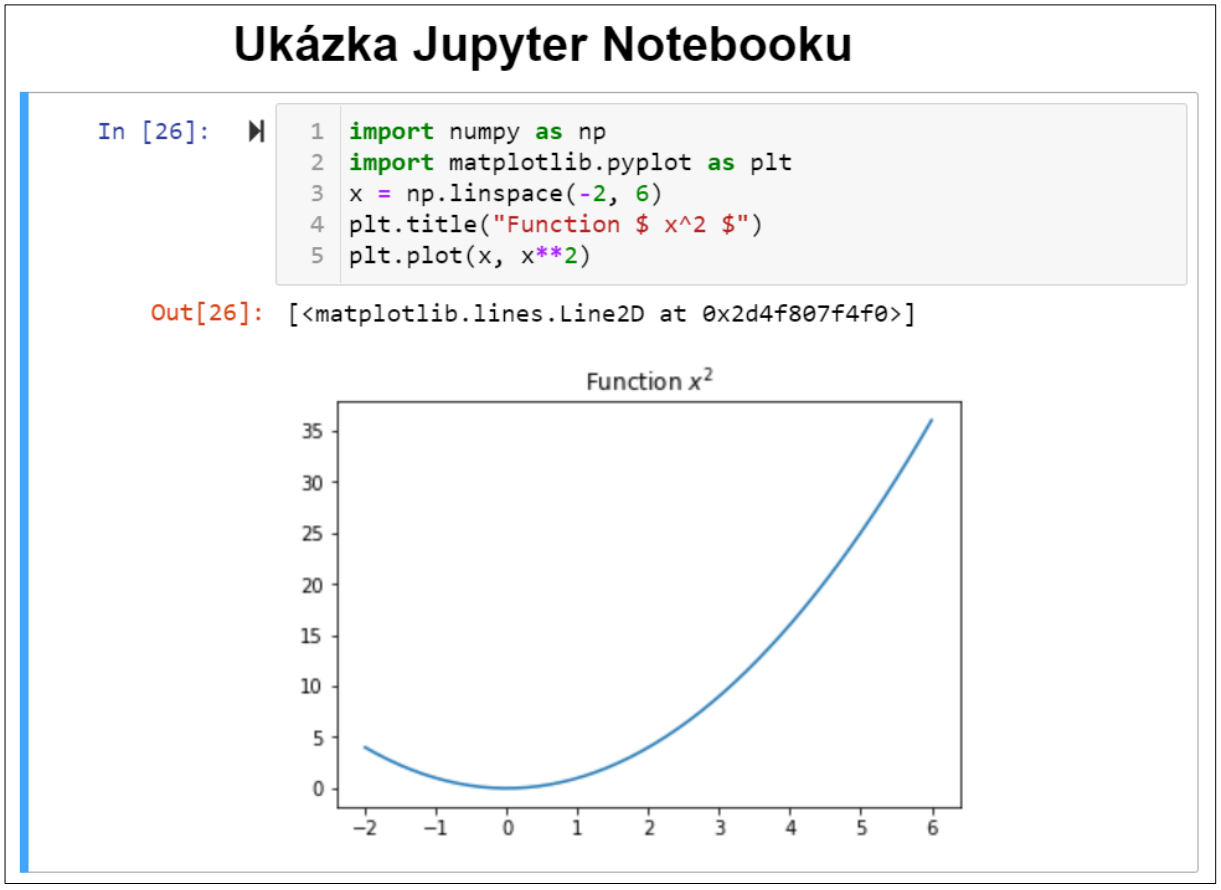
NumPy (Numerical Python) je základní knihovna pro vědecké použití Pythonu. Poskytuje objekt vícerozměrného pole ndarray, matematické operace s tímto objektem a další funkce. Hlavní výhodou této knihovny je její rychlost, je proto používána i jinými knihovnami. [5]

Pandas kombinuje NumPy s prvky relačních databází. Umožňuje manipulovat s daty ve formě tabulek pomocí objektů DataFrame a Series. Mimo samotnou tvorbu a manipulaci s daty se používá i pro jejich ukládání nebo načítání ze souboru. [5]

Matplotlib je knihovna určená pro tvorbu interaktivních grafů a dalších 2D vizualizací dat. Obsahuje širokou škálu možností samotné vizualizace i manipulací s ní. [5]

1.2 Jupyter Notebook

Jupyter Notebook je interaktivní prostředí v rámci webového prohlížeče. Umožňuje psaní kódu rozděleného na úseky, které se dají opakovaně vyhodnocovat. Zároveň dokáže vykreslovat tabulky, grafy, obrázky nebo vypisovat matematické vzorce (jak je vidět na obrázku 1.2.1). Navazuje na projekt IPython, na rozdíl od něj ale podporuje nejen jazyk Python, ale také jazyky MATLAB, Java, C++, C#, PHP a další. [9] Jupyter Notebook lze spouštět lokálně (postup instalace a spuštění je uveden v příloze 1) nebo online pomocí služeb Google Colaboratory nebo JetBrains Datalore.



Obrázek 1.2.1

1.3 Výhody a nevýhody Pythonu

Nespornou výhodou jazyka Python je jeho vysoká popularita, zčásti zapříčiněná jeho jednoduchou syntaxí. Díky jeho rozsáhlé komunitě totiž vzniká mnoho nových knihoven a nástrojů a je často snadné najít řešení chyb a problémů v kódu na internetu. Dále má Python mnoho využití a hodí se nejen pro testování nebo prototypování, ale také pro kompletní implementaci rozsáhlejších projektů (může tak zastávat funkci dvou jazyků). [5]

Hlavní nevýhodou Pythonu je nízká rychlost běhu jeho kódu oproti kompilovaným jazykům, jako je Java nebo C++. Kompenzací této nevýhody může být kratší doba pro psaní samotného kódu. Dále je Python méně vhodný pro programy vyžadující provádění více procesů najednou (multithreading). [5]

2 GeoPandas a Pandas

GeoPandas je knihovna jazyka Python, která umožňuje práci s geometrickými daty, které se mohou vztahovat ke skutečným místům. Jejím základem je knihovna Pandas, rozšířená o možnost manipulace s novými datovými typy. GeoPandas také poskytuje vektorové operace s těmito daty, pro které využívá knihovnu Shapely, a dále možnost zpracování grafického výstupu pomocí knihovny Matplotlib nebo jiných. [10] Tato kapitola se zaměří především na datové struktury a funkce, které GeoPandas převzala z Pandas.

2.1 Základní datové struktury

Jednou z hlavní datových struktur představených knihovnou Pandas je Series. Jedná se o jednorozměrné pole s libovolným typem hodnot. V tomto se podobá struktuře ndarray knihovny NumPy. Series také používá stejné metody pro výběr podmnožin prvků a podporuje stejné matematické operace jako ndarray. Avšak na rozdíl od ndarray, kde k identifikaci konkrétního prvku pole slouží pouze jeho pořadí, Series používá k této identifikaci index, který nutně nemusí odpovídat pořadí prvků. [11] Vychází tak z datové struktury slovník (dictionary), kde jsou tyto hodnoty označované jako klíče, ovšem oproti slovníku nemusí být jedinečné.

DataFrame je tvořen několika objekty Series. Lze si ho představit jako tabulku, ve které jsou sloupci jednotlivé objekty Series. Nezbytnou součástí objektu DataFrame je opět index, který umožňuje rychlý přístup k jednotlivým řádkům. [11] Tvorba a zobrazení objektů Series a DataFrame v Jupyter Notebooku je vidět na obrázku 2.1.1 (indexem je zde pořadí řádků).

```
In [9]:
1 data = numpy.array([5, 1.2, -8, 1])
2 series_object = pandas.Series(data, name="Numbers")
3 series_object
```

```
Out[9]:
0    5.0
1    1.2
2   -8.0
3    1.0
Name: Numbers, dtype: float64
```

```
In [10]:
1 data = numpy.array([5, 1.2, -8, 1])
2 df = pandas.DataFrame({"Numbers": data, "Numbers * 2": data*2})
3 df
```

Out[10]:

	Numbers	Numbers * 2
0	5.0	10.0
1	1.2	2.4
2	-8.0	-16.0
3	1.0	2.0

Obrázek 2.1.1

GeoPandas umožňuje do Series ukládat geometrická data, čímž vzniká GeoSeries. Data v GeoSeries mohou být ve formě bodů, linií nebo polygonů. [12]

GeoDataFrame je DataFrame, který kromě Series obsahuje také jeden nebo více objektů GeoSeries. Jeden z těchto objektů je označován jako *geometry* a jsou na něm prováděny veškeré metody specifické pro GeoSeries. [12] Tyto funkce budou představeny v následující kapitole.

2.2 Základní funkce knihovny Pandas

Pandas disponuje obrovským množstvím funkcí pro manipulaci s objekty DataFrame a Series. Všechny tyto funkce fungují i v GeoPandas. Jelikož se nejedná o hlavní téma této práce, budou zde představeny jen velmi stručně. Informace o všech funkcích jsou k dispozici na webových stránkách oficiální dokumentace Pandas (https://pandas.pydata.org/docs/). [11]

Výběr podmnožin prvků z objektů je umožněn konstrukcemi *loc* (používá k určení prvků index a zahrnuje počáteční i koncový bod) a *iloc* (určuje prvky pomocí jejich pořadí a nezahrnuje ve výčtu koncový bod). Jednoduše lze také vybrat všechny prvky splňující určitou podmínku. [11]

Objekty DataFrame a Series také podporují matematické operace (sčítání, násobení) s číselnými konstantami nebo jinými objekty Series či DataFrame. Také mohou být slučovány, navazovány za sebe nebo porovnávány (kupříkladu metodami *concat, append, merge, compare, reshape, join*). [11]

Další často používanou funkcí je GroupBy. Ta umožňuje seskupit data podle hodnot v určených sloupcích. Na ně pak lze využít tzv. agregační funkce, například pro výpočet součtu nebo průměru hodnot, které spadají do jedné skupiny. Příklad použití GroupBy a agregační funkce *sum* je na obrázku 2.2.1. [11]

In [16]:
```
df = pandas.DataFrame.from_dict(
    {
        "Panda velká": ["Savci", "Medvědovití", 1000],
        "Panda červená": ["Savci", "Pandy malé", 20000],
        "Emu hnědý": ["Ptáci", "Kasuárovití", 725000],
        "Koala medvídkovitý": ["Savci", "Koalovití", 100000]
    },
    orient="index",
    columns=["Třída", "Čeleď", "Populace"]
)
df.groupby("Třída").sum()
```

Out[16]:

Třída	Populace
Ptáci	725000
Savci	121000

Obrázek 2.2.1

Z dalších funkcí zmíním použití Pandas pro práci s chybějícími daty, s časovými a textovými daty nebo možnost využití regulárních výrazů (neboli regex). Také podporuje tvorbu vizualizací (knihovna Matplotlib). [11][13]

2.3 Matematické a statistické funkce

Jelikož se Pandas často používá k analýze velkých souborů dat, jsou statistické a matematické funkce jeho důležitou složkou. Při použití na tabulce s daty (tedy na DataFrame nebo Series) ji některé funkce upravují, jiné vrací jednu hodnotu pro jednotlivé sloupce. Kromě funkcí knihovny Pandas lze na jeho objekty aplikovat také tzv. univerzální funkce (ufuncs) knihovny NumPy. [11][14]

Užitečnou funkcí je *describe*. Ta pro danou tabulku vypíše základní statistické informace podle toho, jaký typ dat se zde vyskytuje. Pro číselné hodnoty například vypíše jejich počet, průměr, směrodatnou odchylku, nejmenší a největší hodnoty a hodnoty v určených percentilech (ve výchozím stavu používá 25., 50. a 75. percentil). [11] Její použití je ukázáno na obrázku 2.3.1.

In [74]:
```
frame = pandas.DataFrame(
    [
        [2.5, 10.0],
        [1.3, 8.7],
        [5.0, 0.8]
    ], columns=["first", "second"])
frame.describe()
```

Out[74]:

	first	second
count	3.000000	3.000000
mean	2.933333	6.500000
std	1.887679	4.978956
min	1.300000	0.800000
25%	1.900000	4.750000
50%	2.500000	8.700000
75%	3.750000	9.350000
max	5.000000	10.000000

Obrázek 2.3.1

Data lze také rozdělit do skupin pomocí funkce *cut* (respektive *qcut* pro rozdělení do kvantilů), určit jejich pořadí při seřazování pomocí *rank* nebo například spočítat jejich procentuální změny. Také je možné zjistit korelaci mezi hodnotami jednotlivých sloupců (funkce *corr*). [11]

2.4 Práce se soubory

Tabulky lze v Pandas vytvářet ručně, v praxi se ale častěji využívá načítání dat ze souboru, po čemž následuje jejich úprava a uložení změn zpět do souboru. Pandas i GeoPandas proto podporuje mnoho formátů souborů pro načítání a ukládání dat.

Základním typem souboru, který se dá načíst jako DataFrame, je CSV (hodnoty oddělené čárkou nebo jiným znakem). Na podobném principu funguje formát dat s pevnou šířkou sloupců, zde ale oddělovač být nemusí. Z textových souborů umí Pandas pracovat ještě s formáty HTML a JSON. Data je možné načítat i ze souborů vytvořených jinými programy, tedy například programy MS Excel, Stata, SAS, Parquet, Feather, ale také soubory SQL nebo OpenDocument. [11]

GeoPandas dokáže načítat většinu formátů vektorových dat, například ESRI Shapefile (shp), GeoJSON, GeoPackage (gpkg), PostGIS, Parquet, Feather, ale i data komprimovaná pomocí archivu zip. Na rozdíl od Pandas využívá GeoPandas pro většinu těchto formátů jedinou funkci (*read_file*). Dále dokáže načítat soubory z webových stránek podle jejich URL. Navíc obsahuje tři vestavěné datové sady, lze ho tedy vyzkoušet i bez použití vlastních dat. [12]

3 Manipulace s geometrickými daty v GeoPandas

Kromě možnosti ukládat a načítat geometrická data přináší GeoPandas také množství vlastních funkcí, kterými manipulaci s těmito daty usnadňují. Všechny funkce a možnosti, které představí tato kapitola, jsou specifické pro GeoPandas, a nelze je proto používat v Pandas (na rozdíl od většiny funkcí z minulé kapitoly, které jsou společné pro obě knihovny).

Všechny funkce, které fungují pro objekt GeoSeries, fungují také pro GeoDataFrame (aplikují se v něm na jeden konkrétní objekt GeoSeries, který je označen jako *geometry*). Tyto funkce jsou většinou převzaté z knihovny Shapely.

3.1 Vytvoření GeoSeries a GeoDataFrame

Jak již bylo popsáno v předchozí kapitole, GeoSeries je jednorozměrné pole s geometrickými daty a GeoDataFrame je tabulka se sloupci tvořená několika sloupci Series a alespoň jedním sloupcem GeoSeries. GeoSeries se vytváří stejnojmennou konstrukční funkcí, která přijímá data ve formě seznamu nebo slovníku. Alternativně lze GeoSeries vytvořit pomocí načtení dat ze souboru funkcí *GeoSeries.from_file*.

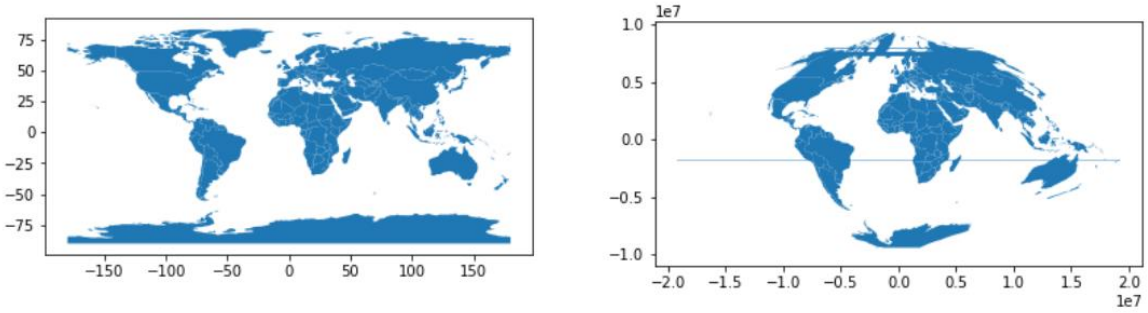
GeoDataFrame se vytváří funkcemi ekvivalentními k GeoSeries. Navíc lze využít metodu *read_file*. Už při načítání se rozsah dat dá omezit vymezením území nebo vybráním konkrétních řádků tabulky. Díky tomu se urychlí práce s velkými soubory. [12]

3.2 Souřadnicový referenční systém

Data v GeoSeries jsou v určitém geografickém souřadnicovém referenčním systému (CRS z Coordinate Reference System). Ten určuje kartografické zobrazení mapy, tedy vztah souřadnic v tabulce k reálným místům na Zemi (nebo jiném tělese). [15] Na obrázku 3.2.1 je zřetelně vidět rozdíl mezi dvěma různými souřadnicovými systémy (vlevo mercator, vpravo sinusoidální zobrazení), které vycházejí ze stejného základu – datové sady vestavěné v GeoPandas. Většinou je CRS nastaven už při načtení dat ze souboru. O tom se lze přesvědčit pomocí příkazu *GeoSeries.crs*, který vrací základní informace o nastaveném CRS.

Pokud není CRS určen, lze jej nastavit pomocí funkce *GeoSeries.set_crs*. Ta přijímá jeho specifikaci ve formě kódu EPSG, WKT, PROJ a dalších. Tyto kódy jsou uvedeny například na stránkách https://spatialreference.org/. Tato funkce nemění souřadnice a v případě, že je CRS v uvedené tabulce již nastaven, neprovede nic. Pro převedení dat do jiného souřadnicového

systému slouží funkce *GeoSeries.to_crs*, která přijímá stejné formáty, ale provádí také úpravu souřadnic. [12]



Obrázek 3.2.1

3.3 Základní geometrické operace

K základní kontrole dat je možné využít funkce, které například zjistí, jaké typy obrazců se v tabulce nacházejí (*geom_type*) nebo jestli se jedná o validní útvary (*is_valid*). Jiné vrátí krajní souřadnice objektů (*bounds* a *total_bounds*) nebo hranice jednotlivých útvarů (*boundary*). Všechny útvary v jednom objektu GeoSeries se mohou spojit pomocí *unary_union*. Naopak *explode* rozdělí útvar tvořený více body, liniemi nebo polygony (ve formě objektů MultiPoint, MultiLine a MultiPolygon) do několika záznamů.

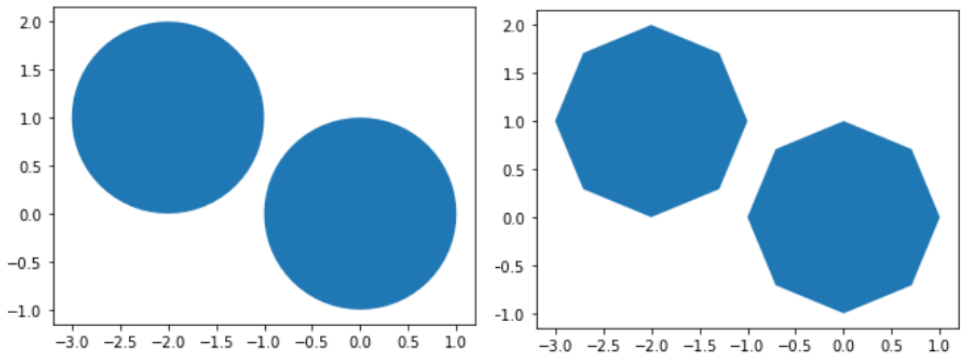
Při tvorbě map je také často důležité měření rozměrů geometrických útvarů. GeoPandas proto umožňuje získat informace o jejich obsahu a obvodu (*area* a *length*). Pro správnost získaných údajů je nutné nastavit tabulku do takového CRS, který používá jako základní jednotku metr (některé používají stupně).

Dále je možné získat bod, který je zaručeně uvnitř daného polygonu, nebo geometrický střed (neboli centroid), což je v rovině bod shodný s těžištěm objektu a nemusí být nutně uvnitř [16] (*representative_point* a *centroid*).

V případě porovnávání dvou geometrických útvarů lze zjistit, jestli se jeden z nich nachází uvnitř druhého (*within*), nebo naopak (*contains*) nebo jestli mají společný průsečík (*intersects*). Také lze určit jejich vzdálenost (*distance*).

Funkce *buffer* zvětší daný objekt o území, které je od něj maximálně v určené vzdálenosti (z bodu tak například v základním nastavení vznikne kruh). Stejně jako u měření obsahu a vzdálenosti je důležité mít souřadnice nastaveny ve správném CRS. Výsledkem této operace je vždy polygon. [17]

Při práci se složitými útvary se může uplatnit jejich zjednodušení (*simplify*). To spočívá v odstranění některých bodů takovým způsobem, aby celkový tvar zůstal přibližně stejný. V GeoPandas je výchozí algoritmus pro zjednodušení pomalejší. Lze použít Douglas-Peuckerův algoritmus, který je mnohem rychlejší, ale nezachovává topologii, a může tak vytvářet méně spolehlivé výsledky. [17] Na obrázku 3.3.1 je použití zjednodušení znázorněné (vlevo dva kruhy vzniklé použitím *buffer(1)* na dva body, vpravo je na ně aplikované *simplify(0.1)*).



Obrázek 3.3.1

3.4 Transformace

Mezi geometrické transformace podporované GeoPandas patří například otočení (*rotate*). To otočí útvar o daný úhel (ve stupních nebo radiánech) ve směru hodinových ručiček. Dále sem patří změna měřítka (neboli škálování, *scale*) podle souřadných os, zkosení (*skew*) a posunutí (*translate*), které na rozdíl od ostatních nemá určený střed transformace.

Všechny jmenované transformace jsou afinní, což znamená, že zachovávají rovnoběžnost, ale ne nutně délky ani velikostí úhlů. [18] Tyto transformace lze také obecně vyjádřit pomocí matice (ta se používá ve funkci *affine_transform*). Tato matice obsahuje šest prvků při používání transformací v rovině a 12 prvků v prostoru. [12][16]

3.5 Operace s překrytím

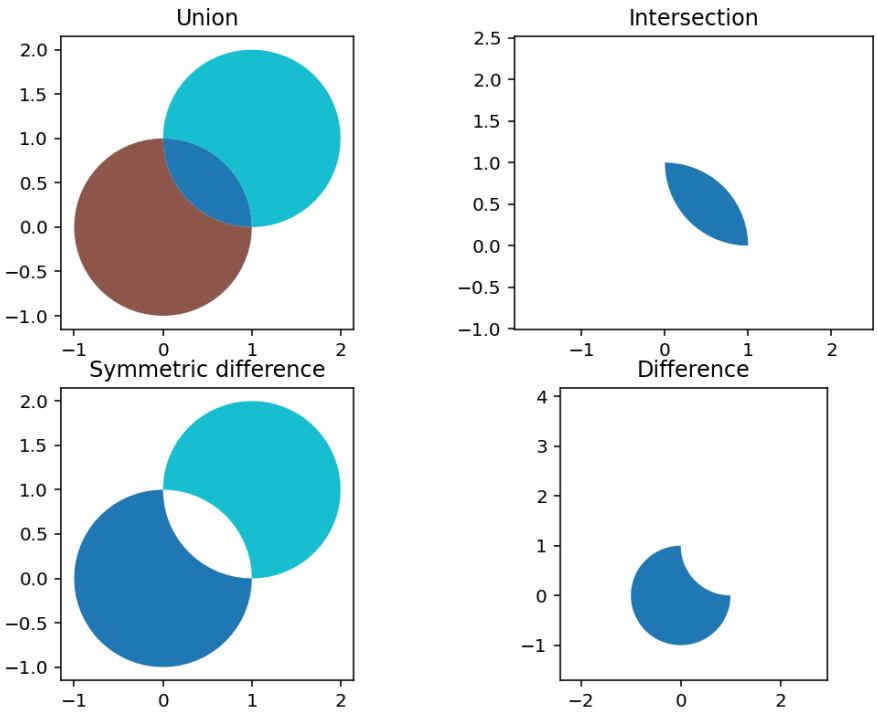
Při používání více datových sad často dochází k tomu, že se některé objekty překrývají. Pro výběr překrývajících se objektů či jejich částí slouží funkce *overlay*. Ta využívá množinové operace – sjednocení, průnik a rozdíl. Zde bude popsáno jejich použití vždy na dvou útvarech.

Sjednocení (*union*) vrátí všechny oblasti, které jsou obsaženy v prvním nebo druhém útvaru, a rozdělí je podle toho, v jakém z počátečních útvarů se nacházejí (na rozdíl od *unary_union* z podkapitoly 3.3 je nespojí do jednoho).

Průnik (*intersection*) vybere pouze ty oblasti, které patří do obou útvarů zároveň. Opakem průniku je symetrický rozdíl (*symmetric_difference*), který vybere oblasti, kde se útvary nepřekrývají.

Rozdíl (*difference*) vybere oblasti z prvního útvaru, které se ale nenacházejí ve druhém útvaru. Ještě existuje možnost *identity*, která vrátí oblasti z prvního útvaru rozdělené podle toho, jestli se nacházejí ve druhém útvaru, nebo ne. [12]

Obrázek 3.5.1 zobrazuje použití různých operací na stejných datech – dvou kruzích (každý útvar má jinou barvu, například po sjednocení tedy vznikají tři útvary).



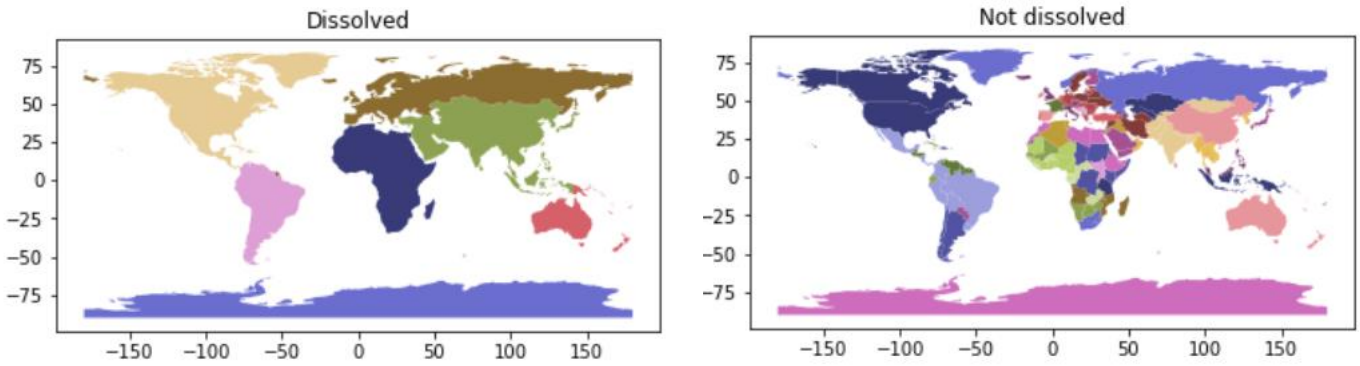
Obrázek 3.5.1

3.6 Agregace a slučování tabulek

Agregace znamená seskupení tabulky podle hodnot v určitém sloupci. V Pandas tuto funkci plnila struktura GroupBy. Ta je podporovaná i v GeoPandas, jejím výsledkem ale není GeoDataFrame, protože nezachovává informace o geometrii. K tomuto je určená funkce *dissolve*, která provádí podobné operace jako GroupBy, ale navíc spojí do jednoho všechny útvary, které po agregaci spadají do jedné skupiny. Hodí se tak například při práci s daty, které

jsou územně více členěné, než je potřeba (například lze spojit data a geometrie pro jednotlivé okresy do krajů). Během tohoto seskupování je také možné využít tzv. agregační funkce. Ty do nové tabulky zapíšou souhrnné informace pro danou oblast. Patří sem výběr prvního nebo posledního prvku, minimální nebo maximální hodnota a také součet, průměr a medián hodnot. [12] Na obrázku 3.6.1 je vidět část tabulky po agregaci podle kontinentů a po použití agregační funkce pro součet hodnot (původní tabulka obsahovala hodnoty pro každou zemi, což je vidět na mapce vpravo).

continent	geometry	population
Africa	MULTIPOLYGON (((32.83012 -26.74219, 32.58026 -...	1219176238
Antarctica	MULTIPOLYGON (((-163.71290 -78.59567, -163.712...	4050
Asia	MULTIPOLYGON (((120.29501 -10.25865, 118.96781...	4389144868



Obrázek 3.6.1

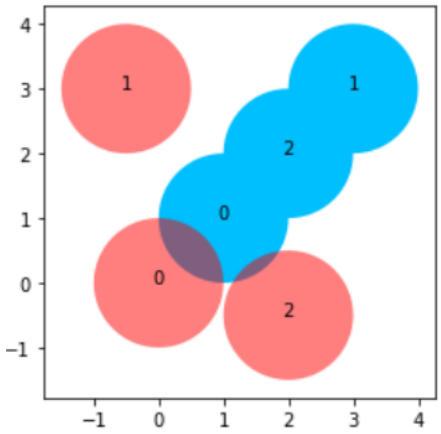
Pro sloučení dvou objektů GeoDataFrame (nebo jednoho GeoDataFrame a DataFrame) slouží funkce *merge* knihovny Pandas. Ta přidává sloupce jedné tabulky ke sloupcům druhé podle určeného sloupce, který mají obě tabulky společný. To vychází z databázového jazyka SQL. Z SQL také přebírá dvě základní možnosti sloučení – vnitřní (*inner*) a vnější, které se dále dělí na levé a pravé (*left* a *right*). U vnitřního sloučení se spojí pouze ty řádky, které se nacházejí v obou tabulkách, u vnějšího se k nim přidají i řádky, které jsou jen v jedné z nich (levé nebo pravé sloučení určuje to, jestli si všechny řádky zachová levá, nebo pravá tabulka). V případě vnějšího sloučení se řádky, které byly pouze v jedné z tabulek, doplní o prázdné hodnoty. [11]

V GeoPandas je také funkce *sjoin*, která slučuje tabulky na základě toho, jak se jejich data překrývají. Zachová vždy geometrii z první tabulky a přidá k nim sloupce z druhé tabulky podle různých kritérií. Porovnává tak například, jestli se jednotlivé útvary dotýkají, mají společný průsečík atd. (Tyto možnosti porovnávání jsou zmíněny ve čtvrtém odstavci podkapitoly 3.3). I zde se uplatňuje možnost vnitřního a vnějšího sloučení. [12]

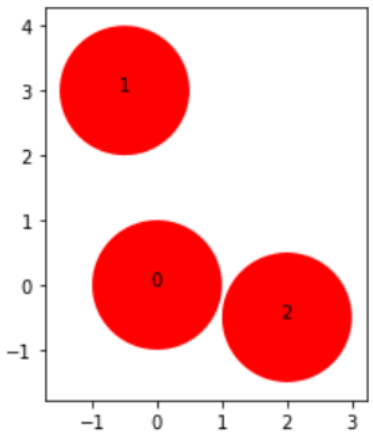
Na obrázku 3.6.2 je vidět levé vnější sloučení (*sjoin*) podle toho, jestli mají útvary společný průsečík. Nahoře jsou vidět dvě tabulky a jejich grafické zobrazení před spojením. Pod nimi je výsledná tabulka a pod ní její grafické zobrazení. Jelikož se jedná o levé sloučení, jsou ve výsledné tabulce všechny útvary z první tabulky (i červený kruh 1, který se nedotýká žádného útvaru z druhé tabulky, ve výsledné tabulce se k němu proto vložily prázdné hodnoty). Modrý kruh 0 se dotýká jiných kruhů, jeho hodnoty se tedy překopírují k nim.

	Column 1 - red	geometry
0	10	POLYGON ((1.00000 0.00000, 0.99518 -0.09802, 0...
1	50	POLYGON ((0.50000 3.00000, 0.49518 2.90198, 0....
2	20	POLYGON ((3.00000 -0.50000, 2.99518 -0.59802, ...

	Column 1 - blue	geometry
0	19	POLYGON ((2.00000 1.00000, 1.99518 0.90198, 1....
1	89	POLYGON ((4.00000 3.00000, 3.99518 2.90198, 3....
2	259	POLYGON ((3.00000 2.00000, 2.99518 1.90198, 2....



	Column 1 - red	geometry	index_right	Column 1 - blue
0	10	POLYGON ((1.00000 0.00000, 0.99518 -0.09802, 0...	0.0	19
1	50	POLYGON ((0.50000 3.00000, 0.49518 2.90198, 0....	NaN	NaN
2	20	POLYGON ((3.00000 -0.50000, 2.99518 -0.59802, ...	0.0	19



Obrázek 3.6.2

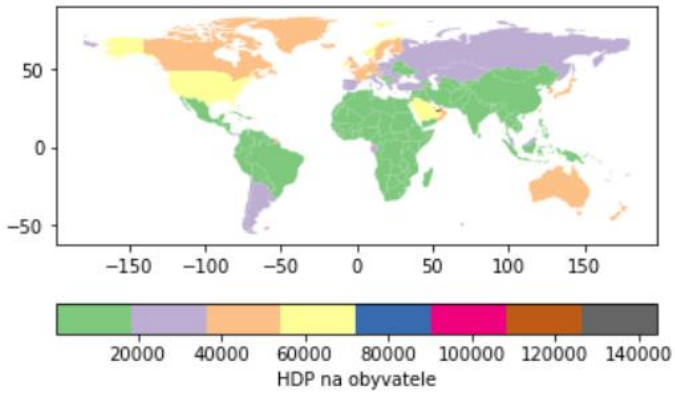
4 Vizualizace map v GeoPandas

Vizualizace je významnou součástí analýzy dat, protože se často jedná o její hlavní výsledek. Zároveň se ale může použít i pro kontrolu toho, že jsou data, s kterými se pracuje, v pořádku. Metod vizualizace dat GeoPandas existuje více, v této kapitole budou představeny dvě nejpoužívanější – použití knihoven *Matplotlib* a *Folium*.

4.1 Matplotlib

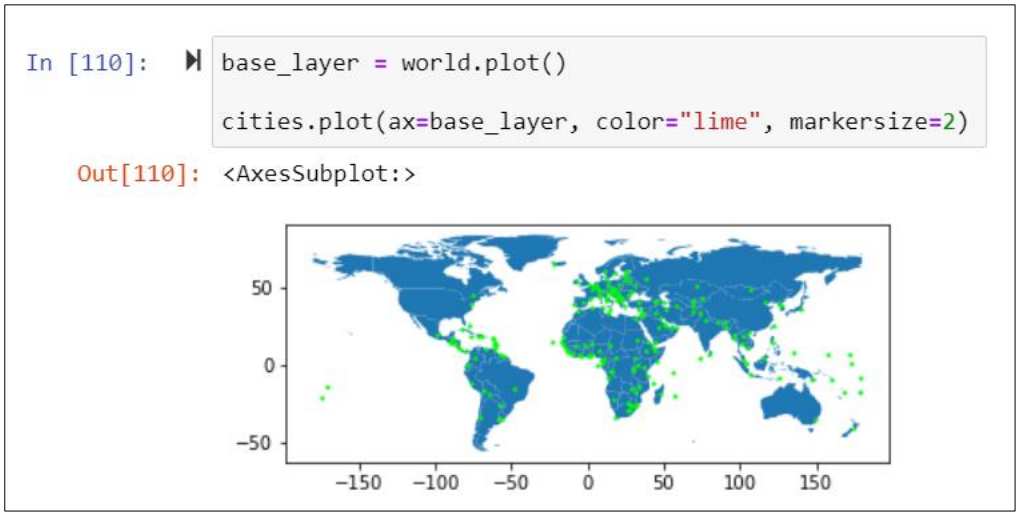
Knihovna Matplotlib je výchozí způsob tvorby grafické mapy v GeoPandas. V GeoPandas je totiž zabudovaná funkce *plot*, která vytvoří vizualizaci pomocí Matplotlib (u knihovny Folium se vizualizace nedělá přímo funkcí GeoPandas, ale funkcemi Folium). V prostředí Jupyter Notebook lze výstupní graf nebo mapu zobrazit mezi jednotlivými buňkami s kódem jako obrázek, nebo v samostatném okně. U druhé možnosti lze s mapou dále manipulovat, přibližovat ji nebo ji uložit jako obrázek do počítače. Knihovna Matplotlib se ale nespecializuje jen na vytváření map (na rozdíl od Folium), proto je některé funkce obtížnější provádět. Také kvůli tomu je podle mě méně vhodný pro tvorbu výstupních map.

Tato knihovna dokáže kromě základního zobrazení dat vytvářet také barevné kartogramy (choropletové mapy), které zobrazují relativní hodnoty určitého jevu. [19] Pro tento účel obsahuje mnoho barevných škál, od postupného barevného přechodu až k tzv. kvalitativním škálám, které mají jasně vymezené hranice mezi různými odstíny. [20] Dále je při výběru barev možné využít předdefinovaná schémata, k tomu je ale nutné nainstalovat knihovnu *mapclassify*. Různé hodnoty lze ale odlišit také pomocí rastru (vzorku). Speciálně pak lze stylizovat oblasti, pro které data chybí. [12] Ke kartogramu lze dále přidat legendu nebo popisky k osám či k jednotlivým oblastem na mapě. Ukázka jednoduchého kartogramu vytvořeného pomocí datové sady vestavěné v GeoPandas je na obrázku 4.1.1.



Obrázek 4.1.1

Při vizualizaci několika tabulek najednou se zobrazují data v jednotlivých vrstvách. Ovládat pořadí vrstev se dá ovládat vytvořením základní vrstvy, ke které se přidávají další. Ty se totiž v daném pořadí vykreslí nad ni. Tento způsob včetně zdrojového kódu je znázorněn na obrázku 4.1.2 (na vrstvu světových zemí je nanesena bodová vrstva s hlavními městy). Při použití více vrstev je ale důležité, aby byly všechny ve stejném CRS. [12]



Obrázek 4.1.2

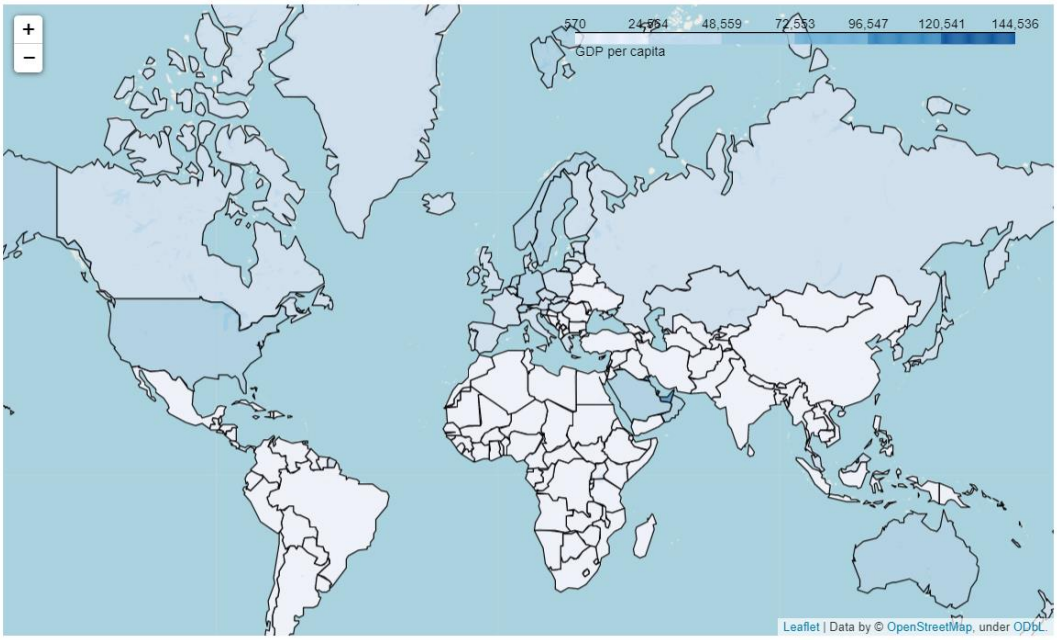
Funkce *plot* také podporuje tvorbu jiných typů grafů, například koláčového, sloupcového, liniového grafu nebo korelačního diagramu. [12] Tyto grafy ale nelze vykreslit na mapě.

4.2 Folium

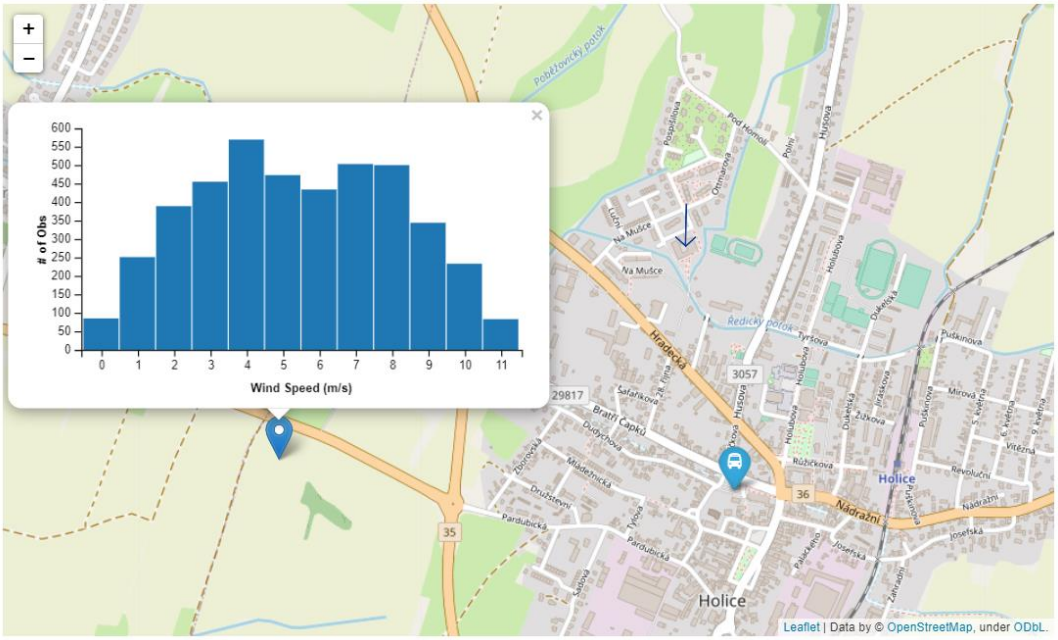
Funkce knihovny Folium vychází z knihovny Leaflet programovacího jazyka JavaScript, která se používá pro tvorbu map na webových stránkách. Výhodou knihovny Folium je možnost využívat podkladové mapy OpenStreetMap, Mapbox a Stamen. [21] Na těchto podkladových mapách pak lze zobrazovat objekty vytvořené například v GeoPandas. Je určená přímo pro tvorbu map, proto obsahuje i funkce, které v jiných knihovnách chybí nebo jsou složitější. Mapy jsou navíc interaktivní, velmi podrobné a je možné je různě stylizovat. K vykreslení podkladu je ale třeba připojení k internetu.

Kartogramy se zde vytvářejí podobně jako v knihovně Matplotlib. Při zobrazování hodnot je ale nutné specifikovat dvě tabulky. Jedna z nich obsahuje geometrická data a je převedena do

formátu GeoJSON. Druhá obsahuje hodnoty, které jsou barvami zobrazené na mapě. Na obrázku 4.2.1 je kartogram vytvořený pomocí stejných dat jako na obrázku 4.1.1.



Obrázek 4.2.1



Obrázek 4.2.2

Jednotlivá místa na mapě lze opatřit interaktivními značkami. Po najetí či kliknutí myši mohou zobrazovat text nebo jiné informace. Po kliknutí tak například zobrazí graf vytvořený v jazyce Vega, který ve formátu JSON popisuje design interaktivních vizualizací. [22] Folium obsahuje pro značky ohromné množství ikon (jsou převzaté z Font Awesome a Bootstrap 3), navíc je

možné vytvořit si vlastní ikony. Dále je tu možnost nechat uživatele přidávat vlastní značky, měnit jejich pozici nebo je odebírat. Na obrázku 4.2.2 je ukázka vlastní ikony (šipka ukazující na Gymnázium Dr. Emila Holuba), ikony Font Awesome (ikona autobusu) a ikony s grafem Vega (data k tomuto grafu pocházejí z ukázkových příkladů v dokumentaci Folium a nevztahují se k reálnému místu, kde je umístěná ikona).

Folium také nabízí další funkce ve formě pluginů. Jedním z nich je teplotní mapa (heat mapa), která zobrazuje intenzitu nějakého jevu v různých oblastech pomocí spojité barevné škály. [23] Dalšími jsou například možnost na mapu kreslit perem nebo zjistit souřadnice místa, kde se uživatel nachází. K mapě lze také přidat časový posuvník, a zobrazit tak změny určité hodnoty v čase. [21]

5 Příklady použití knihovny GeoPandas

V této kapitole využiji informace z předešlých kapitol k vytvoření několika malých map, kterými ukážu různé možnosti využití knihovny GeoPandas, Matplotlib a Folium. U každé je podrobně popsán postup její tvorby. Jejich zdrojový kód je veřejně dostupný, cesta k němu je uvedena v příloze 2.

5.1 Mapa vlajek světových zemí

Tato mapa zobrazuje vlajky většiny světových zemí vykreslené na jejich území. Cílem její tvorby bylo ukázat další možnosti stylizace map v knihovně Matplotlib, konkrétně zobrazování obrázků. Jako podklad jsem využil datovou sadu *naturalearth_lowres*, která je součástí GeoPandas a která kromě jiných informací obsahuje tvary území, názvy a zkratky jednotlivých světových zemí. Obrázky vlajek jsem stáhl ze stránky https://flagpedia.net/, všechny s pevnou šířkou 320 pixelů ve formátu PNG [24].

Nejprve bylo nutné zjistit, jak lze propojit jednotlivé záznamy v tabulce se soubory obrázků vlajek. Naštěstí nebylo nutné přejmenovávat všechny soubory s obrázky, protože byly pojmenovány dvoumístným kódem země. Pomocí knihovny *pycountry* jsem tedy jen převedl zkratky v tabulce z trojmístných kódů na dvoumístné (pycountry totiž obsahuje informace o ISO kódech zemí, ale i o jejich podrobnějším dělení, jejich jazycích a měnách). Některé země ale v původní tabulce neměly uvedené zkratky, proto jsem je ručně doplnil.

Po této přípravě bylo už možné vytvořit samotnou mapu. Nejprve se jako základní vrstva vykreslily všechny státy z tabulky. Pro každý řádek v tabulce se poté podle zkratky země načetl příslušný obrázek. Poté se u zemí, které byly tvořeny několika polygony, vybral ten největší, aby se na něj vlajka vykreslila. Díky tomu byly všechny útvary ve stejném formátu (MultiPolygony se převedly na Polygony). Zároveň to zajistilo, aby byla z vlajky vidět co největší část. Zároveň se tím ale vyfiltrovaly některé oblasti, které jsou na výsledné mapě zakreslené modrou barvou (proto se na začátku musela vykreslit také spodní vrstva, bez ní by tyto oblasti nebyly zakreslené vůbec).

Pro každý řádek v tabulce se následně vykreslil obrázek jeho vlajky. Ten se upravil tak, aby jeho horní strana byla v nejsevernějším bodě území, levá v nejzápadnějším atd., kvůli čemuž se ale změnil poměr jeho stran. Za použití vnějšího okraje území pak vznikl objekt, pomocí něhož byl obrázek vlajky oříznut.

Jak je vidět na obrázku 5.1.1 (a na detailu Jižní Ameriky na obrázku 5.1.2), výsledná interaktivní mapa zobrazuje vlajky všech zemí, i těch, které nejsou samostatnými státy (Grónsko, Antarktida). U některých vlajek je ale výrazně změněn poměr stran nebo nejsou vidět jejich hlavní části. Při velkém přiblížení na interaktivní mapě se navíc obrázky nevykreslují správně. Tato mapa však slouží jako dobrá ukázka toho, že knihovnu Matplotlib lze díky její univerzálnosti použít i pro funkce, pro které nebyla přímo určena. Navíc lze tento program využívat pro vykreslení jakýchkoliv obrázků na jakémkoliv mapovém podkladu. Samozřejmě by jej bylo možné vylepšit. Například po najetí myši zobrazit popisky zemí a další informace, zachovat poměr stran u všech obrázků nebo je otočit tam, kde je vzdálenost od severu k jihu území větší než vzdálenost od východu k západu.

Obrázek 5.1.1

Obrázek 5.1.2

5.2 Mapa oblíbených umělců ze Spotify

Pomocí druhé mapy bych chtěl ukázat další způsob získávání dat v Pythonu. Konkrétně se jedná o výběr mých oblíbených umělců na Spotify pomocí Spotify API a jejich zobrazení na mapě podle zemí, z kterých pocházejí. Zdrojový kód k této mapě je rozdělen na dvě části. První se zabývá získáním dat, druhá tvorbou samotné mapy.

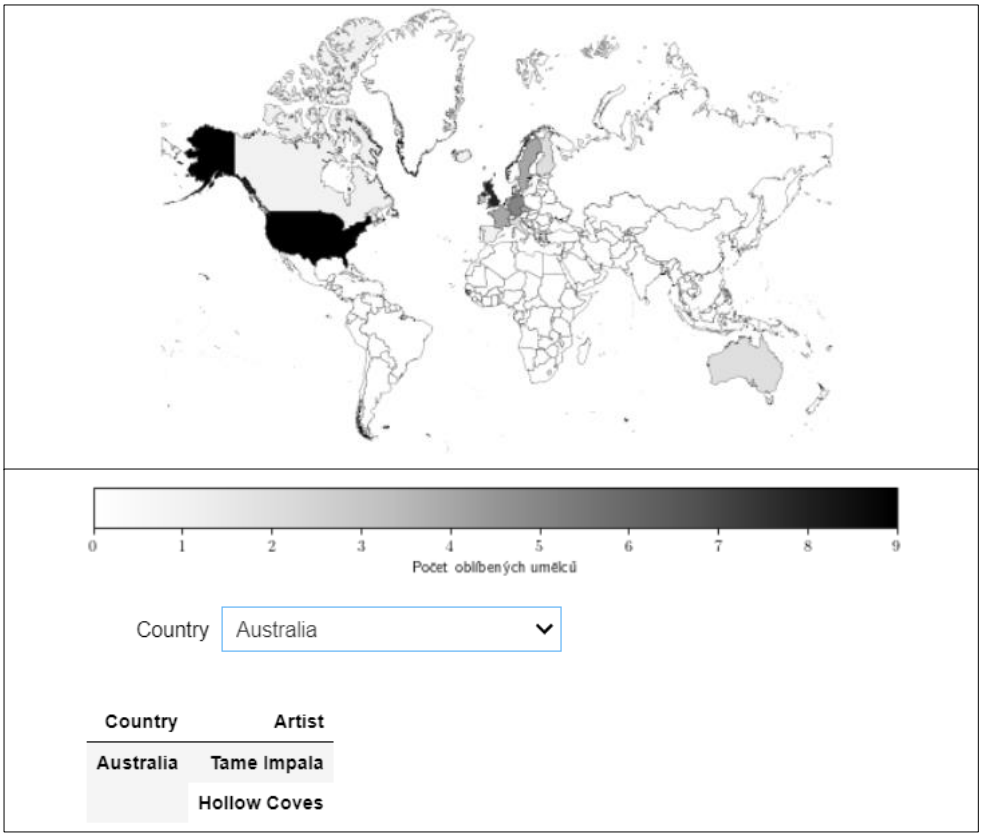
Pro připojení ke Spotify API jsem použil knihovnu *spotipy*. Pomocí ní jsem vygeneroval seznam mých nejoblíbenějších umělců. Pro ně jsem následně pomocí knihovny *musicbrainzngs* vyhledal v internetové databázi MusicBrainz země, odkud pocházejí. Země umělců, kteří nebyli nalezeni v databázi, jsem doplnil ručně. Data o umělcích a jejich zemích se poté uložila ve formátu CSV do samostatného souboru. Před spuštěním první části kódu je totiž nutné provést nastavení v účtu Spotify a získat token a identifikační číslo pro připojení ke Spotify API. Data jsou proto uložena do souboru, a není nutné je znovu generovat. Pro tvorbu mapy tak stačí pouze spustit druhou část kódu.

Druhá část kódu začíná načtením dat z uvedeného souboru a dále načtením mapy světových zemí ze souboru Shapefile. Nepoužil jsem vestavěnou datovou sadu *naturalearth_lowres* jako u minulé mapy, protože se v ní názvy států neshodují s názvy v databázi MusicBrainz. Po načtení dat jsem zkontroloval, že se všechny země z databáze nacházejí na mapě, a přejmenoval ty, které byly špatně pojmenované.

Dalším krokem bylo spojení obou datových sad. Nejprve jsem pomocí funkce GroupBy rozdělil umělce do skupin podle zemí. Následně jsem tyto skupiny pomocí levého sloučení (popsáno v podkapitole 3.6) spojil s tabulkou světových zemí. Výsledná tabulka tak obsahovala informace o názvu, tvaru území a celkovém počtu oblíbených umělců v jednotlivých zemí. V posledním sloupci pak byli jednotliví umělci vypsaní.

Mapa je vytvořena podobným způsobem jako minule. Jsou k ní ale přidané anglické popisky, které se zobrazí po kliknutí myší na jakoukoliv zemi. Ze souřadnic kurzoru myši se vytvoří bod a následně se zjistí, jestli se daný bod nachází uvnitř nějakého objektu v tabulce. Pokud ano, zviditelní se popisek s názvem země a s umělci, kteří z dané země pocházejí. Po kliknutí pravým tlačítkem nebo kolečkem myši popisek opět zmizí. Pod buňkou s kódem je také rozbalovací menu se všemi státy, kde je alespoň jeden umělec. Po výběru položky z menu se daná oblast na mapě přiblíží a pod buňku se vypíšou všichni umělci z dané země.

Na obrázku 5.2.1 je vidět mapa vykreslená v buňce Jupyter Notebooku, která ale není interaktivní a neumožňuje vytvářet popisky po kliknutí ani přiblížení oblastí. Na obrázku 5.2.2 je detail Evropy na interaktivní mapě. Tato mapa už umožňuje všechny funkce zmiňované výše.



Obrázek 5.2.1

Obrázek 5.2.2

5.3 Mapování vývoje populace světových zemí

Třetí mapou chci ukázat další způsob získávání dat a také to, jak se dají jednotlivé součásti knihovny Folium propojit na jedné mapě. Data jsem převzal ze seznamu zemí podle minulé a předpokládané budoucí populace na anglické Wikipedii. [25] Na mapě jsem pomocí časově proměnné mapy zobrazil (skutečné a předpokládané) průměrné roční procentuální změny v počtu obyvatel. U vybraných států lze po kliknutí zobrazit graf vývoje jejich obyvatelstva ve sledovaném období.

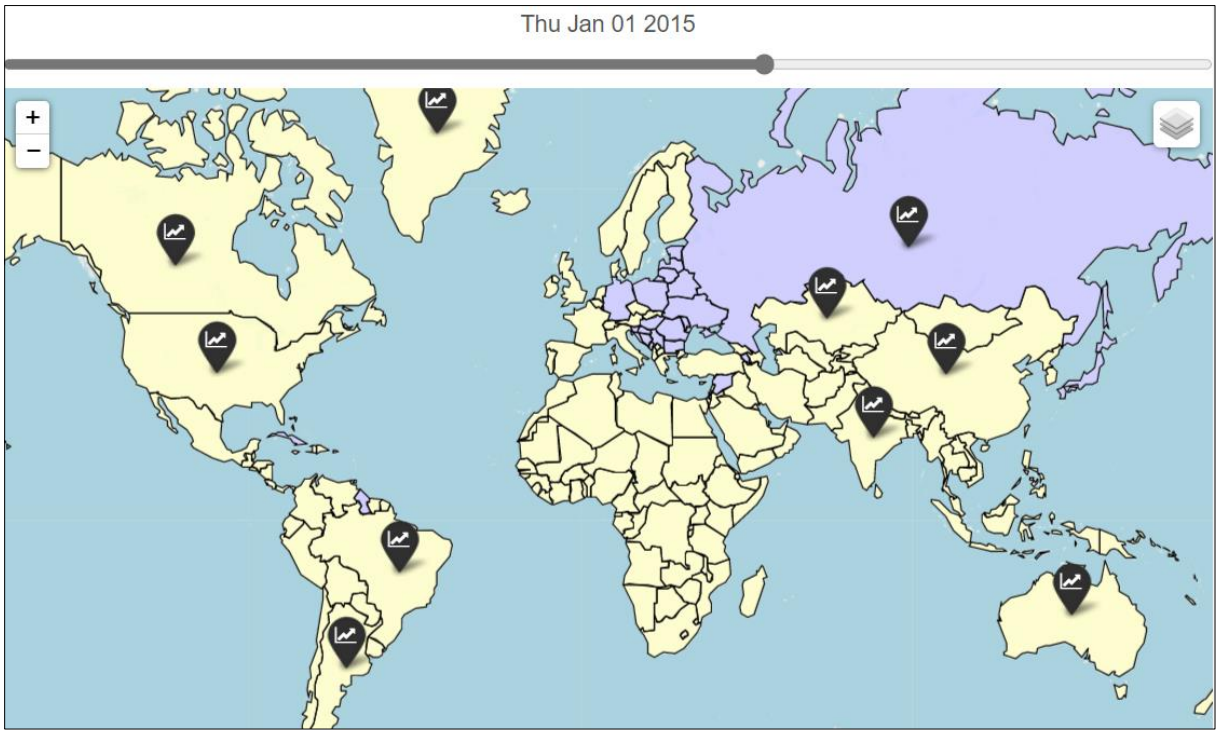
Data z tabulek na webových stránkách lze získat pomocí funkce Pandas (*read_html*). Jelikož jsou na výše zmíněné stránce tři oddělené tabulky, sloučil jsem je do jedné. Smazal jsem z nich přebytečná data a ponechal pouze informace o procentuálních změnách. Dále jsem tuto tabulku spojil s daty v *naturalearth_lowres* (zmíněno v 5.1). Předtím jsem ale ručně upravil názvy některých států, aby se při slučování jejich data neztratila. Tím vznikla jedna tabulka s informacemi o všech zemích, včetně jejich zobrazení na mapě. Hodnoty vyjádřené v procentech jsem rozdělil na pět skupin, podle toho, jakou barvou budou na výsledné mapě zobrazeny.

Folium bohužel nedokáže automaticky vytvořit požadovanou časovou mapu pouze pomocí číselných hodnot pro daná kalendářní data. Místo toho je nutné pro každý časový údaj nutné specifikovat styl vykreslení jednotlivých území (například barvu nebo průhlednost). Proto jsem do tabulky dosadil pět barev pro jednotlivé skupiny hodnot. Letopočty jsem následně převedl na milisekundy. Tato data jsem převedl do formátu JSON, aby s nimi knihovna Folium mohla pracovat.

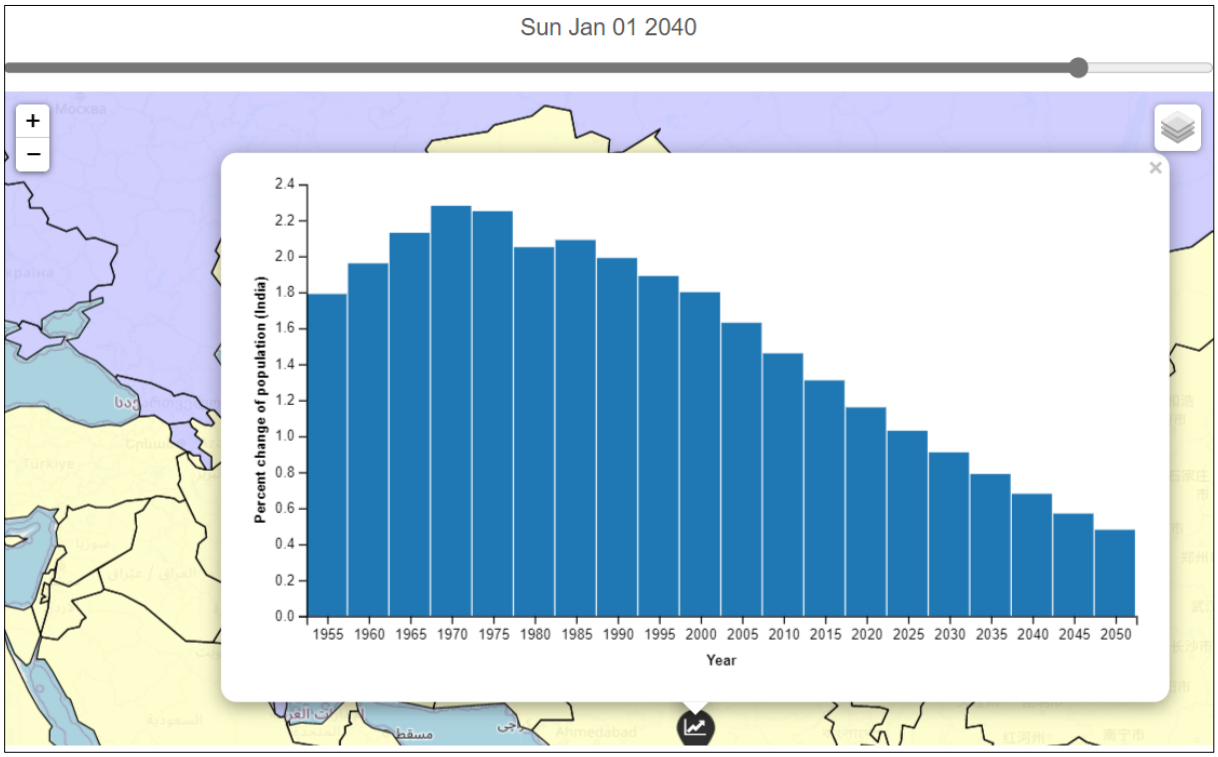
Jako podkladovou mapu jsem využil OpenStreetMap. To není nezbytné, výsledná mapa může fungovat i bez podkladu. Dále jsem vytvořil časově proměnnou mapu (*TimeSliderChoropleth*), která pomocí různých barev ukazuje vývoj obyvatelstva v čase (modrá ukazuje úbytek obyvatel, žlutá až zelená přírůstek). Dále jsem pro každé území zobrazil jeho hranice a název, který se objeví po najetí myší. Na deset největších jsem také umístil značku, která po rozkliknutí zobrazí sloupcový graf vývoje obyvatel na daném území. (Tyto značky by se daly umístit na všechna území, ale poté by byla mapa nepřehledná.) Samotné sloupcové grafy jsou vytvořené v knihovně *vincent*, která umožňuje vytvářet grafy ve formátu Vega. V pravém horním rohu mapy je tlačítko pro zobrazování nebo skrývání jednotlivých skupin objektů na mapě.

Na obrázku 5.3.1 a na detailu grafu pro Indii 5.3.2 jsou vidět jednotlivé objekty a funkce výsledné mapy. Barevná škála je zvolena poměrně nahodile, vhodné by nejspíš bylo jemnější

členění hodnot do více skupin. Nahodile jsou také zvolena území, na kterých je značka s grafem (toto se v kódu ale dá celkem jednoduše změnit). Pro zobrazení vývoje změn obyvatelstva by také bylo vhodnější použít spojnicový graf. Mapa tak nepředstavuje konečný výsledek analýzy daných dat, ale ukazuje různé možnosti využití knihovny Folium, což byl její hlavní účel.



Obrázek 5.3.1



Obrázek 5.3.2

Závěr

Cílem této práce bylo stručně představit knihovnu GeoPandas, což se jí dle mého názoru podařilo. Myslím si, že výsledný text může být vhodným počátečním zdrojem informací o GeoPandas pro ty, kteří se o práci s daty a výrobu map zajímají. Těm, kteří nemají žádné zkušenosti s programováním, jej může přiblížit ukázka map v praktické části a zdrojový kód v příloze.

Osobně si myslím, že knihovna GeoPandas a další knihovny jazyka Python jsou pro svoji flexibilitu a vzájemnou propojenost velmi užitečné nástroje nejen pro programátory. V určitých situacích tak mohou být vhodnou alternativou k velkým počítačovým programům určeným pro analýzu a vizualizaci dat (přestože použití GeoPandas je poměrně specifické), ale i jiné procesy.

Seznam použité literatury

[1] *History and License — Python 3.9.1 documentation.* [online]. Copyright © [cit. 15.02.2021]. Dostupné z: https://docs.python.org/3/license.html

[2] *Lekce 1 - Úvod do Pythonu. itnetwork.cz* - Ajťácká sociální síť a materiálová základna pro C#, Java, PHP, HTML, CSS, JavaScript a další. [online]. Copyright © 2021 itnetwork.cz. [cit. 16.02.2021]. Dostupné z: https://www.itnetwork.cz/python/zaklady/python-tutorial-uvod-do-pythonu-a-zakladni-matematicke-operace

[3] *PYPL PopularitY of Programming Language index.* [online]. Copyright © Pierre Carbonnelle, 2020 [cit. 15.02.2021]. Dostupné z: https://pypl.github.io/PYPL.html

[4] *10 Famous Websites Built Using Python - Learn to code in 30 Days.* Learn to code in 30 Days - OneMonth.com [online]. Dostupné z: https://learn.onemonth.com/10-famous-websites-built-using-python/

[5] MCKINNEY, Wes. Python for Data Analysis. Sebastopol (California): O'Reilly Media, 2012. ISBN 9781449319793.

[6] *Sunsetting Python 2 | Python.org.* Welcome to Python.org [online]. Copyright ©2001 [cit. 15.02.2021]. Dostupné z: https://www.python.org/doc/sunset-python-2/

[7] *34 Open-Source Python Libraries You Should Know About.* [online]. Copyright © 2020 Great Learning All rights reserved [cit. 18.02.2021]. Dostupné z: https://www.mygreatlearning.com/blog/open-source-python-libraries/

[8] *PyPI · The Python Package Index* [online]. Copyright © 2021 [cit. 18.02.2021]. Dostupné z: https://pypi.org/

[9] *Root.cz - informace nejen ze světa Linuxu* [online]. Dostupné z: https://www.root.cz/clanky/jupyter-notebook-nastroj-pro-programatory-vyzkumniky-i-lektory/

[10] *Open Source Spatial: GeoPandas Part 1 | Spatial Vision.* Spatial Vision | Geospatial Intelligent Solutions | Get In Touch Today [online]. Copyright © Spatial Vision 2020 [cit. 11.03.2021]. Dostupné z: https://spatialvision.com.au/blog-open-source-spatial-geopandas-part-1/

[11] *User Guide — pandas 1.2.3 documentation.* pandas - Python Data Analysis Library [online]. Copyright © Copyright 2008 [cit. 14.03.2021]. Dostupné z: https://pandas.pydata.org/docs/user_guide/

[12] *User Guide — GeoPandas 0.9.0 documentation. GeoPandas 0.9.0 — GeoPandas 0.9.0 documentation* [online]. Copyright © Copyright 2013 [cit. 15.03.2021]. Dostupné z: https://geopandas.org/docs/user_guide.html

[13] VANDERPLAS, Jacob T. Python data science handbook: essential tools for working with data. Beijing: O'Reilly, 2016. ISBN 978-1-491-91205-8.

[14] *NumPy user guide — NumPy v1.20 Manual. NumPy* [online]. Copyright © Copyright 2008 [cit. 17.03.2021]. Dostupné z: https://numpy.org/doc/stable/user/index.html

[15] *8. Coordinate Reference Systems — QGIS Documentation documentation.* [online]. Copyright © Copyright 2008 [cit. 21.03.2021] Dostupné z: https://docs.qgis.org/3.16/en/docs/gentle_gis_introduction/coordinate_reference_systems.html

[16] *Geometrický střed – Wikipedie.* [online]. Dostupné z: https://cs.wikipedia.org/wiki/Geometrick%C3%BD_st%C5%99ed

[17] *The Shapely User Manual — Shapely 1.8a2 documentation.* [online]. Copyright © Copyright 2008 [cit. 25.03.2021] Dostupné z: https://shapely.readthedocs.io/en/latest/manual.html

[18] *07 Maticové transformace* [online]. Copyright © [cit. 26.03.2021]. Dostupné z: https://saint-paul.fjfi.cvut.cz/base/sites/default/files/POGR/POGR2/07.maticove_transformace.pdf

[19] *Kartogram* [online]. Copyright © [cit. 31.03.2021]. Dostupné z: https://www.natur.cuni.cz/geografie/geoinformatika-kartografie/ke-stazeni/projekty/moderni-geoinformacni-metody-ve-vyuce-gis-a-kartografie/kartogram/

[20] *Matplotlib: Python plotting — Matplotlib 3.4.0 documentation. Matplotlib: Python plotting — Matplotlib 3.4.0 documentation* [online]. Copyright © Copyright 2002 [cit. 31.03.2021]. Dostupné z: https://matplotlib.org/stable/index.html

[21] *Folium — Folium 0.12.1 documentation. GitHub Pages* [online]. Copyright © Copyright 2013, Rob Story. [cit. 01.04.2021]. Dostupné z: https://python-visualization.github.io/folium/

[22] *A Visualization Grammar* | Vega. Vega [online]. Dostupné z: https://vega.github.io/vega/

[23] *Heat mapa; Heatmap | Marketingový výzkum a analýza dat - STEM/MARK*. [online]. Copyright © 2020 STEM [cit. 01.04.2021]. Dostupné z: https://www.stemmark.cz/encyklopedie-heat-mapa-heatmap/

[24] *Download flag bitmap images of all countries in the world | Flagpedia.net. Flags of the World | Flagpedia.net* [online]. Copyright © 2008 [cit. 04.04.2021]. Dostupné z: https://flagpedia.net/download/images

[25] *List of countries by past and projected future population - Wikipedia*. [online]. Dostupné z: https://en.wikipedia.org/wiki/List_of_countries_by_past_and_projected_future_population

Přílohy

Příloha 1

Instalace a spouštění uvedených technologií (na systému Windows)

Spouštění kódu online

Většina zdrojového kódu se dá spustit pomocí online Jupyter Notebooku bez instalace Pythonu. Osobně doporučuji využít JetBrains Datalore (https://datalore.jetbrains.com/), kde je knihovna GeoPandas a některé další instalovány předem (ovšem ne knihovna folium, ta se dá nainstalovat příkazem *!pip install folium*). Dále je možné Jupyter Notebook vyzkoušet v Google Colab (*https://colab.research.google.com/*) nebo na stránce *https://jupyter.org/try*. Při použití online Jupyteru Notebooku můžete ostatní body v této příloze přeskočit.

Ověření instalace Pythonu

Úspěšná instalace Pythonu (ze stránek www.python.org) se dá ověřit zapsáním příkazu *python* (respektive *py* nebo *python3*) v programu Příkazový řádek. Pokud se po zadání příkazu vypíše verze Pythonu, proběhla instalace v pořádku.

Stahování knihoven pomocí správce pip a virtuální prostředí

Přímo z Příkazového řádku lze instalovat knihovny pomocí příkazu *python -m pip install nazevKnihovny*. Pro lepší kontrolu nad staženými knihovnami se využívá virtuální prostředí. Uchovává totiž knihovny v daných verzích, a tak se nemůže stát, že po instalaci nových přestanou některé programy fungovat. Prostředí se vytváří příkazem *python -m venv nazevSlozky* (více informací naleznete na https://naucse.python.cz/lessons/fast-track/install/). Aktivuje se pomocí příkazu nazevSlozky\Scripts\activate (všimněte si zpětného lomítka).

Spuštění Jupyter notebooku offline

Pro lokální použití Jupyter noteboku je třeba ho nainstalovat (pomocí *python -m pip install notebook* v Příkazovém řádku nebo *conda install -c conda-forge notebook* v terminálu Anaconda). Po zapsání příkazu *jupyter notebook* do Příkazového řádku (či jiného terminálu) se ve vybrané složce otevře prostředí Notebooku.

Instalace knihovny GeoPandas

GeoPandas závisí na mnoha dalších knihovnách a ve Windows je jeho přímá instalace přes pip problematická. Osobně se mi osvědčilo použití kroků popsaných v https://geoffboeing.com/2014/09/using-geopandas-windows/.

Příloha 2

Zdrojový kód k práci

Zdrojový kód ke všem částem práce je k dispozici online na stránce: https://github.com/standakozak/Seminarni-prace-geopandas.