Univerzita Hradec Králové
Fakulta informatiky a managementu
Katedra informatiky a kvantitativních metod

Využití jazyka Python pro vizualizaci dat

Bakalářská práce

Autor: David Továrek
Studijní obor: Aplikovaná informatika

Vedoucí práce: Ing. Bruno Ježek, Ph.D.

Hradec Králové Březen 2022

Prohlášení:

Prohlašuji, že jsem bakalářskou práci zpracoval samostatně a s použitím uvedené literatury.

V Hradci Králové dne 24.4.2022 David Továrek

Poděkování:
Děkuji vedoucímu bakalářské práce Ing. Bruno Ježkovi, Ph.D. za vedení práce, cenné rady a doporučenou literaturu.

Anotace

Tato bakalářská práce se zabývá možnostmi vizualizace dat v programovacím jazyce Python. Teoretická část obsahuje základní principy vizualizace a popis často používaných knihoven pro zpracování dat a jejich následnou vizualizaci. Praktická část následně ukazuje využití jazyka Python pro načítání dat z různých zdrojů, navrhuje možná zadání úloh napomáhajících ve výuce práce s vizualizačními knihovnami a prezentuje ukázková řešení. Poslední částí práce je hodnocení vybraných vizualizačních knihoven a možností jejich využití pro různé účely.

Annotation

Title: Data visualization using Python

This Bachelor Thesis deals with data visualization in the Python programming language. The theoretical part describes basic visualization principles and libraries often used for the processing of data and the following visualization. The practical part shows the use of Python in data gathering and presents exercises along with example solutions that could be helpful in teaching basics of work with these libraries. The last part of the thesis contains an evaluation of visualization libraries considering their use for different purposes.

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

Programovací jazyk Python se díky své jednoduchosti a flexibilitě stal velmi populárním a pronikl do nejrůznějších oblastí, včetně vědeckého výzkumu. Jedním z důležitých aspektů jakéhokoliv výzkumu je zpracování a vizualizace dat, jak z důvodu prezentace výsledků, tak hledání nových poznatků. Vzniklo tedy velké množství knihoven a nástrojů pro jazyk Python, které se zabývají právě touto problematikou, od všeobecných knihoven usnadňující práci s daty různých formátů, až po vysoce specializované nástroje uplatnitelné pouze v některých oborech.

Tato bakalářská práce si klade za cíl seznámit čtenáře se základními principy datové vizualizace a s knihovnami jazyka Python, které je možné k tomuto účelu využít. Práce se dále zabývá příklady získávání dat a jejich zpracováním v podobě ukázek a praktických úloh, které mohou najít uplatnění ve výuce předmětů oboru Datové vědy.

V neposlední řadě jsou také zhodnoceny vybrané obecné vizualizační knihovny jazyka Python s ohledem na jejich použití v různých situacích. Výsledky jsou porovnávány převážně subjektivně na základě zkušeností autora, vybrané aspekty jsou kvantifikovány a shrnuty v odděleném objektivnějším hodnocení.

2 Vizualizace

2.1 Vizualizace obecně

Pojem „vizualizace dat/informací“ není úplně jednoznačně definován a různé zdroje nabízí velmi odlišná vysvětlení. Kniha „Lecture Notes in Computer Science“ z roku 1970 popisuje, že „*Vizualizace informací využívá počítačové grafiky a interakce, aby napomohla lidem v řešení problémů*“ [1]. Vizualizace však nemusí nutně sloužit pouze k tomuto účelu, může najít uplatnění například ve školní výuce. „*Matematické koncepty, jako čísla, funkce, nebo vektory […] nejsou součástí intuitivního chápání, tak jako reálné fyzické objekty*“ [2]. Vizualizací je však možné pochopení těchto konceptů studentům usnadnit. „*Informační vizualizace nabízí možnost snáze a lépe ukázat klasické vizuální reprezentace matematických formátů, ale také je obohatit o prvky pohybu a interaktivity […] informační vizualizace tedy plní didaktické funkce nezbytné pro výuku matematiky.*“ [2]

Vizualizace může plnit i dekorativní roli, Lorène Fauvelle v článku „Data visualization: definition, examples, tools, advice“ popisuje „Datové umění“ jako nejvyšší formu vizualizace kde „*Získávání informací a postřehů již nestačí, výsledek musí být také vizuálně atraktivní…* “ [3] v rozhovoru s Nicholasem Rougeux popisuje, že „*Datový umělec se nesnaží pouze informovat, ale hlavně vyvolat emoce*“. [4]

Většina zdrojů se však shoduje na dvou základních konceptech „redukce“ a „prostoru“, které popisuje i Lev Manovich v článku „What is Visualization?“: „*Vizualizace informací využívá jednoduchých grafických elementů jako zástupce reálných objektů a vazeb mezi nimi – nezáleží na tom, jestli se jedná o lidi, ceny na burze, příjmy států, nezaměstnanost, nebo cokoliv jiného. Skrze tyto grafické elementy informační vizualizace odhaluje vzorce a struktury v datech. Nicméně, cenou za tuto možnost je extrémní úroveň schematizace. Zahazujeme 99 % toho, co je na objektech specifické, ve snaze nalezení vzorců na zbývajícím 1 % charakteristik.*“ [7]
„*Co mají všechny vizualizační techniky společné, kromě redukce? Všechny využívají prostoru (umístění, velikosti, tvaru a v poslední době i zakřivení a pohybu) pro reprezentaci klíčových rozdílů v datech a zobrazení nejdůležitějších vzorců*

a vztahů. […] Ostatní, méně důležité vlastnosti objektů jsou pak reprezentovány jinými vizuálními prostředky – odstínem, barvou, vzorem stínování nebo i průhledností. “ [7]

2.2 Principy datové vizualizace

Základním cílem vizualizace je zjednodušit porozumění datům, ať už hmatatelným (jako jsou počty produktů ve skladu, hladiny řek v čase, nebo rozmístění zastávek autobusu), ale i čistě abstraktním, které nemají fyzickou podobu a jsou tak složitější k pochopení. Aby bylo možné lépe vytvářet grafy zobrazující data srozumitelným způsobem, je nejprve nutné pochopit jakým způsobem člověk zpracovává zrakové vjemy. K tomuto účelu může dle Encyclopedia of Human-Computer Interaction využít zákonů Gestaltismu (česky také „Tvarové psychologie“). Konkrétně jsou při vytváření vizualizací uplatnitelné následující principy:

Princip blízkosti – Objekty nacházející se blízko u sebe, jsou vnímány jako skupina
Princip podobnosti – Objekty stejného, nebo podobného tvaru, barvy či jiných vlastností jsou vnímány jako skupina
Princip ohrady – Objekty ohraničené čarou, nebo na stejnobarevném poradí jsou považovány za skupinu
Princip uzavření – Vnímání tvaru jako celku i v případě, že obsahuje mezery
Princip návaznosti – Objekty zarovnané do čáry/křivky jsou považovány za jeden celek
Princip spojitosti – Všechny objekty spojené čarou jsou považovány za jeden celek [18]

Využitím těchto principů lze čtenáře upozornit na konkrétní závislosti nalezené v grafu, ale také vyhledat a upravit části vizualizace, které by mohly vést k nepravdivým závěrům.

2.3 Barvy v datové vizualizaci

Dalším důležitým aspektem vizualizace jsou použité barvy a barevné palety. Ačkoliv mnoho zdrojů se shoduje, že účelem barev není estetika, i ta je pro mnohé uživatele aspektem pro volbu specifických barev v grafu. [20]

Barevné palety lze rozdělit do třech základních skupin. Každá z těchto skupin je vhodná pro vizualizaci jiného typu dat. Sekvenční palety se obvykle skládají z různých odstínů jedné barvy, které se liší svým jasem. Taková paleta je vhodná pro data, která jsou nějakým způsobem příbuzná, protože dle principu podobnosti, popsaném v předcházející kapitole, budou vnímána jako jedna skupina. [19] Zvláštní podmnožinou sekvenčních palet jsou takzvané „pro lidské vnímání uniformní“ (ang. „perceptually uniform“) barevné palety, kde není využita pouze jedna barva, ale několik barev s proměnlivým jasem tak, aby vnímaný rozdíl mezi stejně vzdálenými hodnotami byl stejný ve všech částech palety a předešlo se tak ztrátě informací, nebo naopak nalezení zdánlivých anomálií v částech s vyšším vnímaným rozdílem. [21]

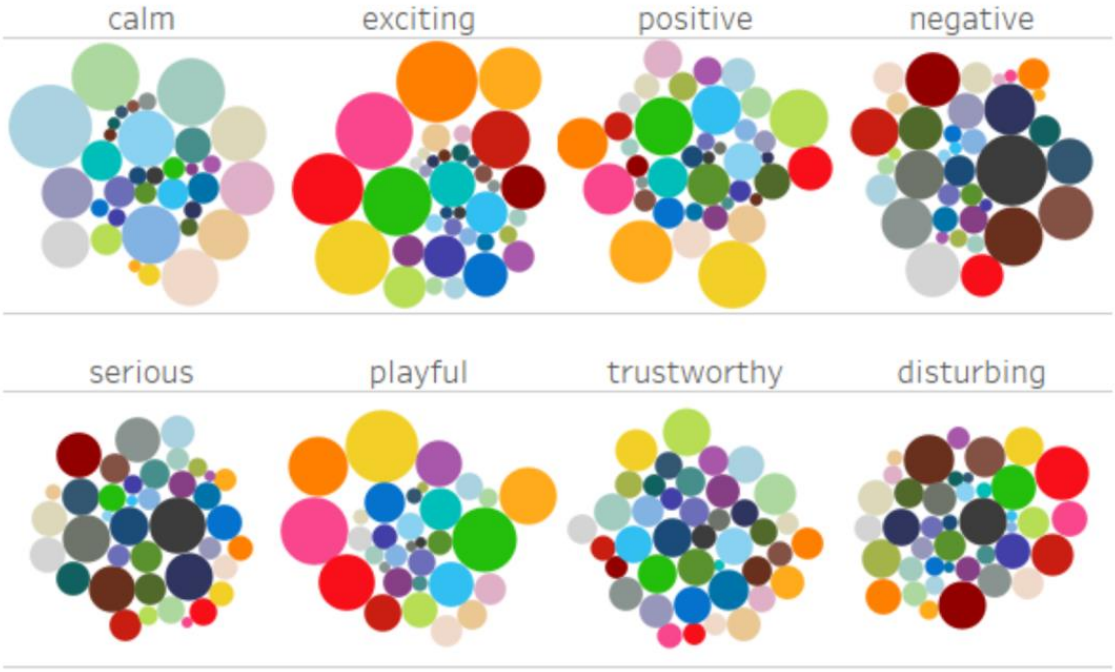
Obr. 1 Pro lidské vnímání uniformní sekvenční palety knihovny Matplotlib
Zdroj: https://matplotlib.org/stable/tutorials/colors/colormaps.html

Dalším typem jsou divergující (rozcházející se) barevné palety, které přechází z jedné kontrastní barvy do druhé přes „inflexní bod“ v jejich středu (obvykle se jedná o velmi světlou barvu, často bílou). Tato barevná paleta je vhodná pro data, která se nějakým způsobem dělí na dvě hlavní skupiny (například kladné a záporné hodnoty, nebo hodnoty, které se dají prezentovat jako positivní a negativní). [19] [20]

Posledním typem jsou kategorické (někdy také kvalitativní, či nominální) barevné palety, používané pro data, které nemají žádné souvislosti a jsou navrženy tak, aby je nebylo možné vnímat jako posloupnost. [20] Ačkoliv některé z těchto palet mohou být velmi obsáhlé, pro zachování přehlednosti se doporučuje nepoužívat více než 8 různých kategorií. [19]

Při výběru palety je také nutné myslet na to, jak budou barvy na čtenáře působit. Dle průzkumu popsaném v práci „Affective Color in Visualization“ je možné rozdělit

barevné palety do skupin podle pocitu, který vyvolávají. Konkrétně uklidňující palety se skládají převážně ze světlých, chladných barev s nízkou sytostí (například různé odstíny modré a zelené). Naopak vzrušující se vyznačují vysokou sytostí a využívají hlavně teplých barev, jako je červená, žlutá, nebo oranžová. Asi nejdůležitějším aspektem je, jestli daná vizualizace celkově působí positivním, nebo negativním dojmem. Jako positivní jsou obvykle vnímány syté barvy, často obsahující odstíny modré, zelené, ale i žluté či oranžové, naopak negativně jsou vnímány tmavé barvy, především odstíny červené, hnědé a šedé. [22]

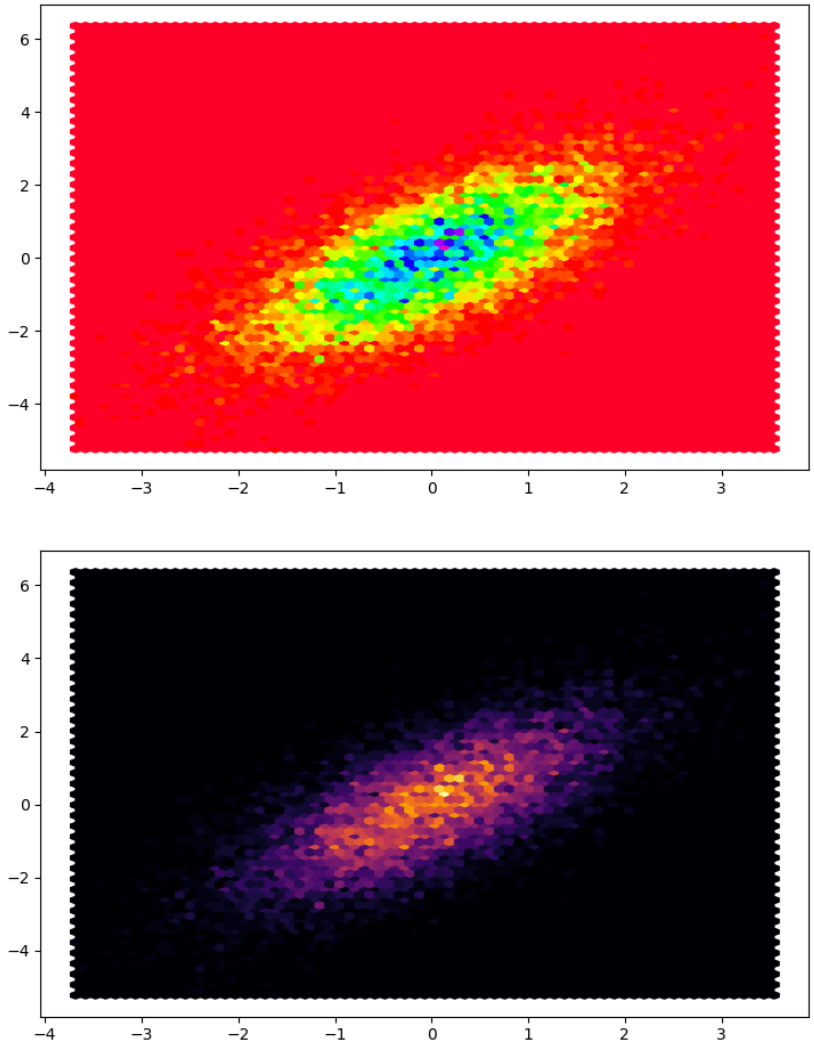


Obr. 2 Skupiny barev dle asociovaných pocitů
Zdroj: Lyn Bartram, Abhisekh Partra a Maureen Stone: Affective Color in Visualization

2.4 Nevhodná volba barevné palety

Volba barevné palety je velmi důležitým krokem při tvorbě jakékoliv vizualizace. Jak již bylo zmíněno v předchozích kapitolách, tato volba může pomáhat s nalezením závislostí a vlastností zobrazovaných dat, zvýraznit nejdůležitější poznatky čtenáři, nebo dokonce přidat emocionální prvek. Je tedy zároveň velmi snadné vybrat barevnou paletu, která bude pro daný typ vizualizace nevhodná, bude zakrývat důležité informace, nebo působit rušivým dojmem. Asi nejčastěji používanou barevnou paletou nevhodnou pro vizualizaci většiny dat je duhová paleta.

Tato paleta s sebou nese hned několik problémů: Pořadí barev je sice všeobecně známé, ale nelze intuitivně rozlišit vyšší a nižší hodnoty. Druhým problémem je vnímání rozdílu mezi barvami, například odstíny světle modré působí dojmem „rychlejší“ změny než odstíny zelené. Posledním důležitým problémem, který není zdaleka omezený na duhovou paletu, je složité vnímání takových palet pro čtenáře se zhoršeným barvocitem. [26] Navzdory těmto chybám jsou duhové barevné palety často součástí vizualizačních programů a knihoven, někdy dokonce jako výchozí.



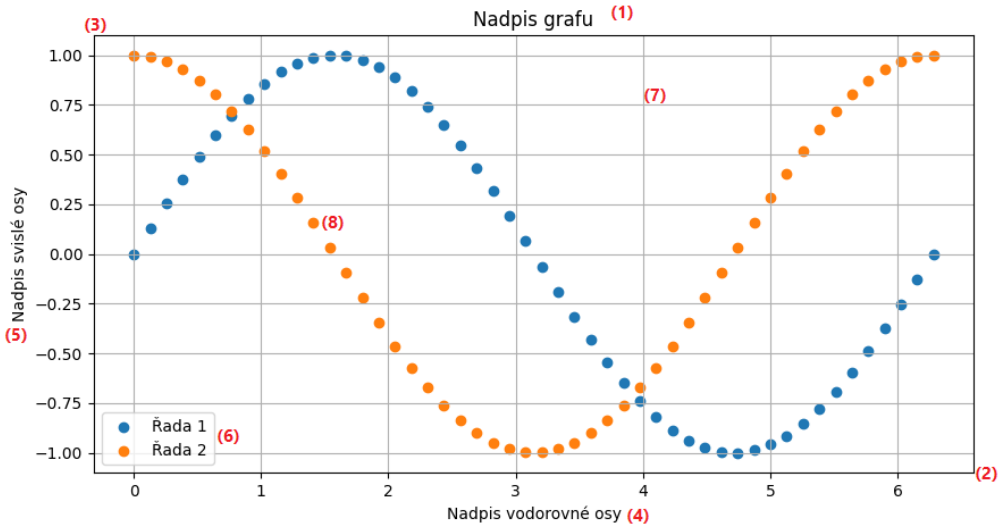
Obr. 3 Porovnání barevných palet
Zdroj: vlastní zpracování – Knihovna Matplotlib

Grafy na obrázku X jsou vytvořeny na základě stejných dat (10000 hodnot normálního rozdělení pro osu X a Y rovno x + náhodná hodnota normálního rozdělení), ale využívají různé barevné palety. Horní graf využívá duhovou paletu

„gist_rainbow“, zatímco dolní využívá pro lidské vnímání uniformní paletu „inferno“, obě dostupné v knihovně Matplotlib. Červená barva pro nulové hodnoty horního grafu působí velmi rušivým dojmem. Odstíny modré a zelené ve středu grafu vytváří zdání skupin s rozdílnými vlastnostmi na základě principu podobnosti, ačkoliv dle dolního grafu je patrné, že zde dochází k plynulému zvyšování četnosti hodnot, bez výrazného zlomu, jak by horní graf mohl naznačovat.

2.5 Anatomie grafu

V této kapitole budou popsány základní součásti grafu, společné pro většinu typů. Elementy specifické pro určité typy grafů budou popsány v následující kapitole.



Obr. 4 Obecný graf
Zdroj: vlastní zpracování – Knihovna Matplotlib

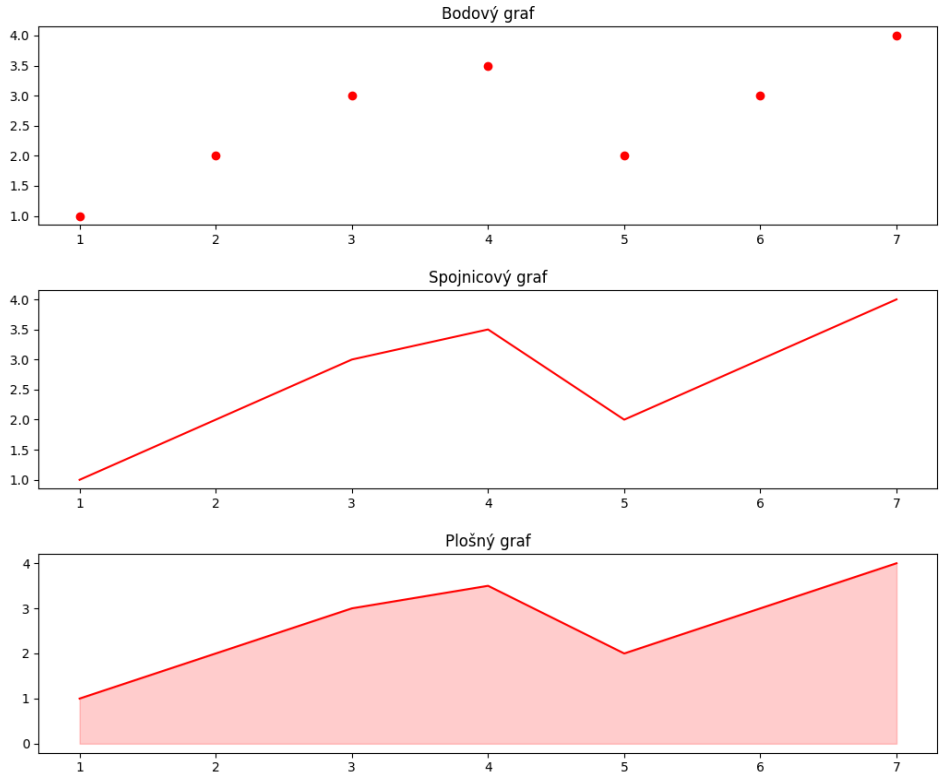
Nadpis grafu (1) může být umístěn kdekoliv na ploše grafu, zpravidla se však nachází nad zobrazovanou oblastí grafu. Osa Y, nejčastěji svislá, (3) obvykle zobrazuje hodnoty závislé na ose X (2). U některých grafů však hodnoty na osách nemusí být závislé. Nadpisy os (4 a 5) mohou poskytnout čtenáři více informací o zobrazovaných datech jako například jednotky. Legenda (6) slouží k popsání barev, které byly přiřazeny zobrazovaným datovým řadám. Mřížka (7) prodlužuje značky os pro snadnější určení hodnot jednotlivých datových bodů (8).

2.6 Nejčastější typy grafů

Existuje mnoho způsobů, jak rozdělit typy grafů do skupin, ať už podle grafických elementů, které využívají (body, čáry, plochy), nebo oborů, ve kterých se převážně využívají (finance, věda, zpravodajství). V této kapitole budou grafy rozděleny do skupin dle jejich funkce, stejně jako je rozděluje Mike Yi v příručce „How to Choose the Right Data Visualization“ [23], nebo Severino Ribecca v projektu „Data Visualisation Catalogue“ [24].

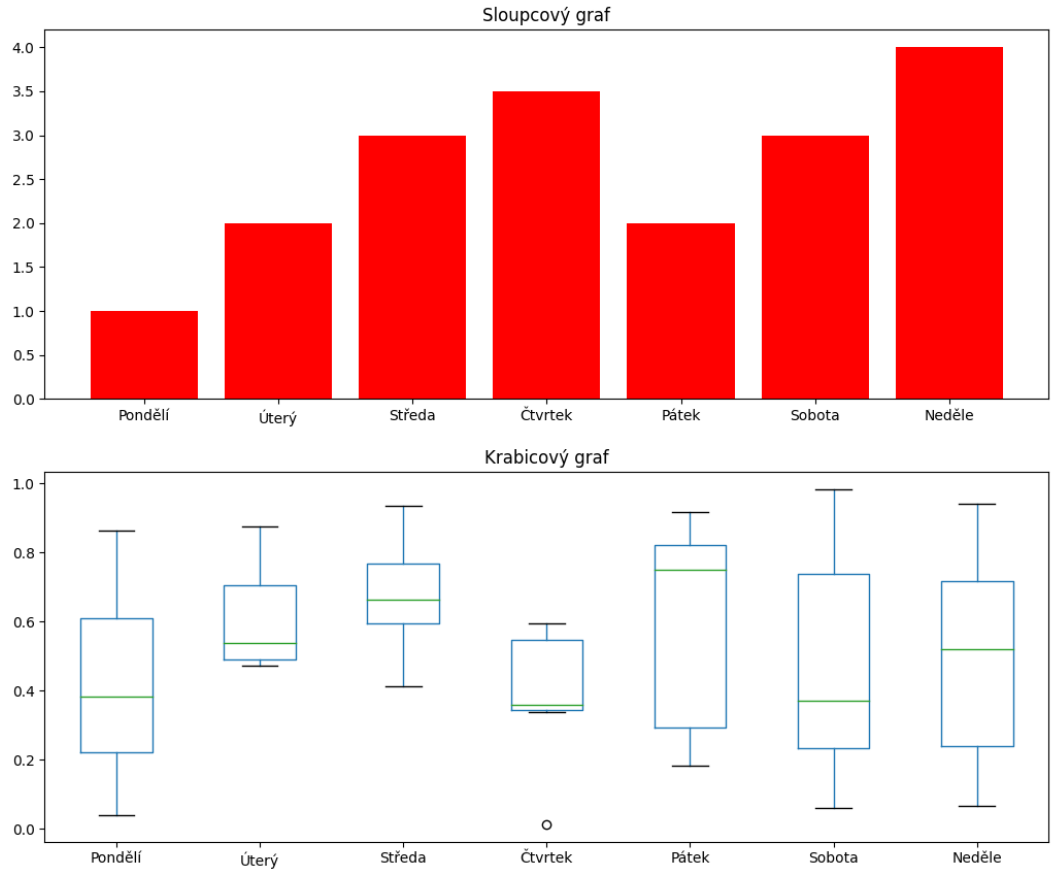
2.6.1 Grafy vyjadřující změnu hodnoty v čase

Častým využitím datové vizualizace je porovnání jedné, nebo více proměnných, které se mění v čase. Pro takové případy je vhodné využít bodový, spojnicový, nebo plošný graf v závislosti na počtu měření a počtu sledovaných veličin. Plošný graf je vhodnější pro vyjádření změny v čase nebo rozdílu mezi několika veličinami, spojnicový naopak více zvýrazňuje konkrétní hodnoty, nicméně může být zavádějící v případě rychlých změn. [24] [25]



Obr. 5 Ukázka grafů vyjadřujících změnu v čase
Zdroj: vlastní zpracování – Knihovna Matplotlib

Pro vizualizaci změny za delší časová období lze také využít sloupcový graf, kde každý sloupec vyjadřuje součet, nebo průměr za dané období. V případě, že je nutné předat více informací o nasbíraných datech, je také možné využít krabicový graf. [23]



Obr. 6 Ukázka využití sloupcového a krabicového grafu pro vyjádření změny v čase
Zdroj: vlastní zpracování – Knihovna Matplotlib

2.6.2 Grafy vyjadřující poměr hodnot

Dalším velmi častým úkolem vizualizace je zobrazení poměru různých kategorií nebo jejich rozložení v rámci celku. Asi nejčastějšími příklady grafů využívajících kategorií jsou sloupcový a koláčový graf, kde koláčový také ukazuje poměr vůči celku, zatímco u sloupcového obvykle není tato informace příliš patrná. Mezi další specifické druhy těchto grafů patří například takzvaný „donut“, což je koláčový graf s chybějícím středem. Takové uspořádání dovoluje umístit popisky do středu grafu a vytvořit tak kompaktnější vizualizaci, zároveň napomáhá při porovnávání více

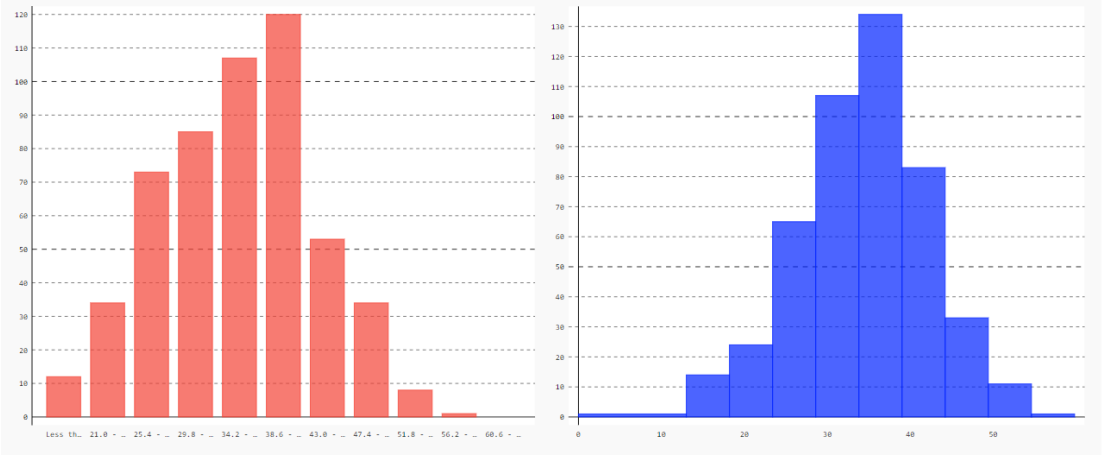
grafů tohoto typu, protože kladou menší důraz na plochu. [24] Některé zdroje však nedoporučují využívat koláčových grafů a „donutů“ a tvrdí, že pro lidské vnímání je složité překládat úhly na hodnoty. [25]

Detailnějším grafem pro zobrazení poměrů hodnot, které tvoří určitou stromovou hierarchii (například rozložení dat v souborovém systému nebo zisky z různých odvětví průmyslu), je takzvaná stromová mapa:

Obr. 7 Ukázka využití stromové mapy pro zobrazení souborového systému
Zdroj: vlastní zpracování – knihovna Pygal

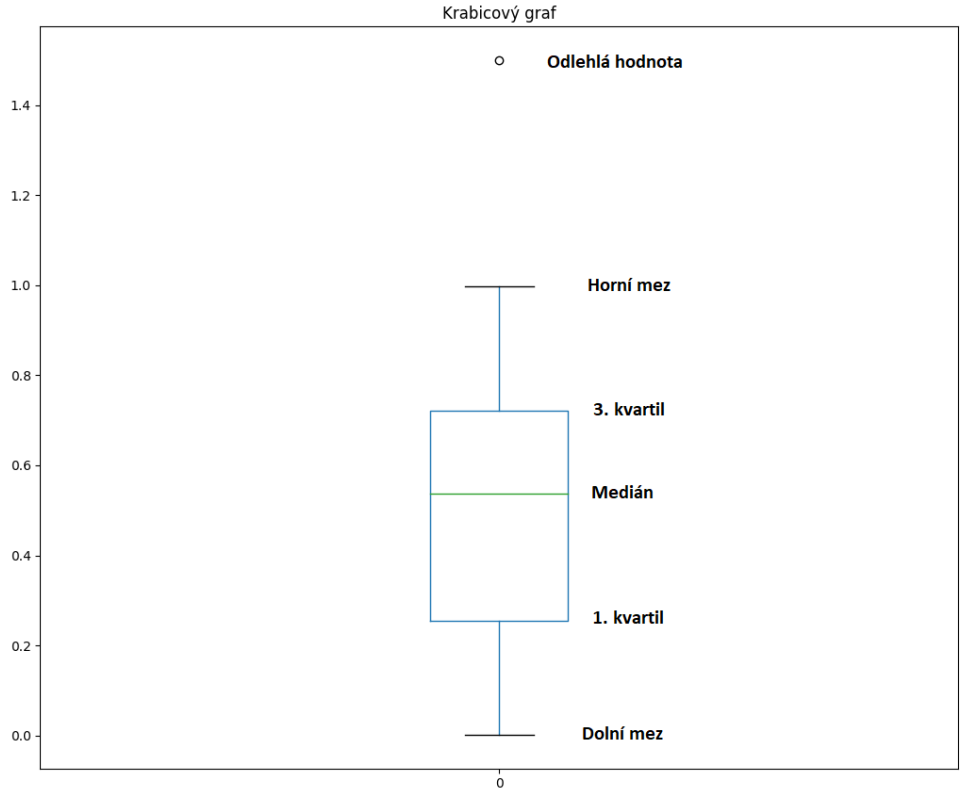
2.6.3 Grafy vyjadřující rozdělení hodnot

Pro vizualizaci rozdělení hodnot lze využít již zmíněné sloupcové grafy, ale také jim velmi podobné histogramy. V některých případech jsou tyto typy grafů téměř identické, ale pokud histogram obsahuje pouze numerické hodnoty, může u některých histogramů být šířka sloupce proměnlivá v závislosti na velikosti intervalů (někdy také nazývaných „třídy“).



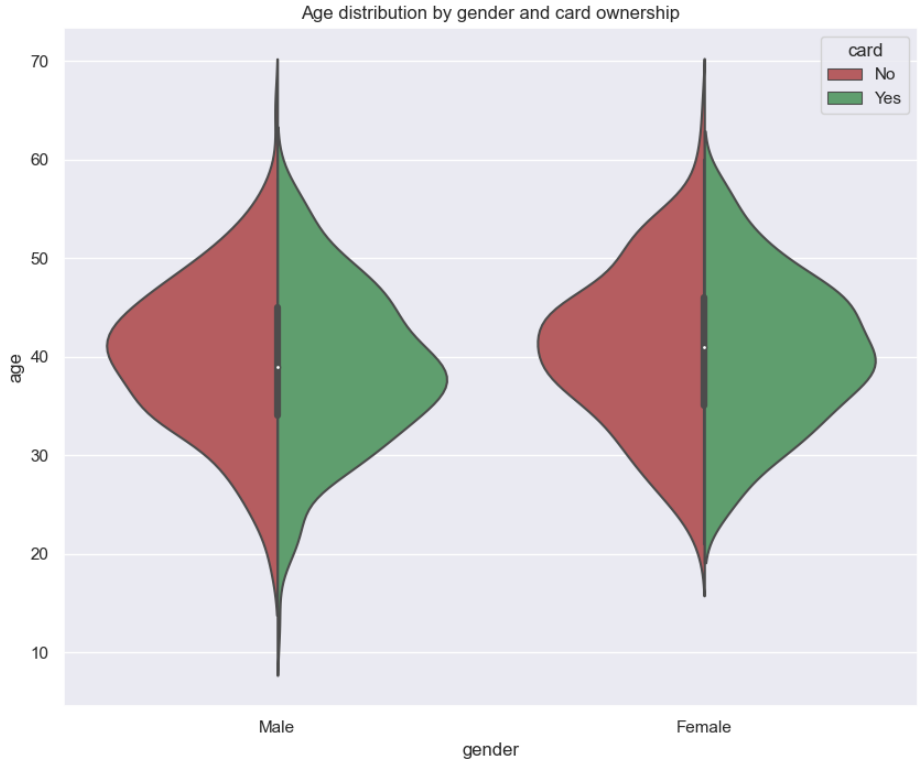
Obr. 8 Histogram jako sloupcový graf s kategoriemi intervalů (červený) a s proměnlivou šířkou sloupce (modrý)
Zdroj: vlastní zpracování – knihovna Pygal

Dalším již zmiňovaným grafem sloužícím k vizualizaci rozložení je krabicový graf, nejčastěji používaný ve statistice. Jednotlivé jeho části ukazují medián, kvartily, horní a dolní mez a případné odlehlé hodnoty. [24]



Obr. 9 Krabicový graf doplněný o popis součástí
Zdroj: vlastní zpracování – knihovna Matplotlib

Zajímavou kombinací krabicového grafu s křivkou ukazující hustotu pravděpodobností hodnot je takzvaný houslový graf/diagram. Stejně jako krabicový graf zobrazuje medián a kvartily, pomocí značek ve středu „houslí“. V některých případech nemusí být symetrický, pokud data mohou být dále dělena do dvou kategorií. Z tohoto grafu je také dobře porovnatelné rozložení hodnot mezi více skupinami.



Obr. 10 Houslový graf s nesymetrickým rozdělením podle skupin
Zdroj: vlastní zpracování – knihovna Seaborn

3 Jazyk Python

Jazyk Python je interpretovaný objektově orientovaný programovací jazyk, který v dnešní době patří mezi jeden z nejpoužívanějších. V listopadu roku 2021 byl na prvním místě žebříčků PYPL a TIOBE index [27] [28], hodnotící programovací jazyky podle počtu kurzů pro daný jazyk vyhledaných online a počtu programátorů využívajících daný jazyk. Využití nachází ve vědě a výzkumu, výuce, webových aplikacích, ale i při vývoji jiných aplikací pro testování a management. [30]

Jeho popularita se zakládá hlavně na jeho jednoduchosti, dobré čitelnosti kódu, přenositelnosti programů mezi platformami, velkému množství rozšiřujících knihoven a snadné integraci s komponentami v jiných jazycích. Nevýhodou jazyka Python je především jeho rychlost, která je oproti kompilovaným jazykům nižší, což však nemusí být problémem při mnoha využitích tohoto jazyka. Problém s rychlostí lze také řešit oddělením náročnějších částí programu do kompilovaných rozšíření, následně volaných z hlavního interpretovaného programu. [29] Další možnou nevýhodou může být pro některé programátory nezvyklá syntaxe využívající odsazení kódu a konce řádků.

4 Knihovny jazyka Python pro zpracování dat

4.1 NumPy

Asi nejčastěji používanou rozšiřující knihovnou jazyka Python je NumPy. Jedná se o knihovnu přidávající podporu n-dimenzionálních polí a práce s nimi. Na rozdíl od seznamů jazyka Python mají tato pole definovanou velikost a datový typ. Tento rozdíl je způsoben kompilovaným kódem jazyka C, který NumPy využívá pro velmi rychlé výpočty s velkými objemy dat. Právě díky této rychlosti se tato knihovna stala velmi oblíbeným nástrojem pro vědecké výpočty a je využívána velkým množstvím dalších knihoven. Kromě základní manipulace n-dimenzionálních polí a výpočtů s nimi nabízí tato knihovna také efektivní algoritmy pro jejich třídění, výpočty lineární algebry, statistické operace, diskrétní Fourierovy transformace i generování náhodných rozložení. Pro složitější operace je možné využít nadstavby SciPy. [31] [29]

4.2 SciPy

SciPy je rozšířením knihovny NumPy, zaměřujícím se převážně na efektivní implementaci složitějších matematických algoritmů v oblastech lineární algebry, včetně řešení rovnic, statistiky, optimalizačních úloh, interpolace nebo výpočtu určitých integrálů. Krom toho umí SciPy také pracovat se signály a obrazovými daty a provádět operace jako filtrování, konvoluce nebo gradientní operátory a nabízí i možnost práce se soubory programu MATLAB. [32]

4.3 pandas

Často využívanou knihovnou při tvorbě vizualizací je knihovna pandas, sloužící k manipulaci s tabulkovými datovými strukturami pomocí objektů Series (pro jednodimenzionální data stejného typu) a DataFrame (pro obecná tabulková data). Dovoluje snadno přidávat a odebírat sloupce dat, kombinovat datové soubory nebo vytvářet výběry na základě podmínek. Velmi praktická je také možnost vytvářet objekty DataFrame přímo ze souborů a databází, konverze textových záznamů na datové typy vyjadřující čas a práce s časovými řadami. Knihovna také disponuje schopností vytvářet vizualizace přímo z objektů DataFrame pomocí knihovny Matplotlib, častěji je však DataFrame použit jako zdroj dat pro jiné vizualizační knihovny. [33]

5 Vizualizační knihovny jazyka Python

5.1 Matplotlib

Pravděpodobně nejznámější vizualizační knihovna jazyka Python byla vytvořena okolo roku 2003 Johnem Huterem a byla původně určena k vizualizaci dat zaznamenaných elektrokortikografií při výzkumu epilepsie. Laboratoř, ve které John Hunter pracoval, měla v té době pouze jednu licenci na softwarový balíček pro analýzu dat, o který se museli všichni výzkumníci dělit. Hunter se tedy rozhodl vytvořit náhradu v prostředí MATLAB. Tato aplikace však nebyla ideální pro vizualizaci dat z mnoha zdrojů (kromě elektrokortikografie například i EEG a magnetické rezonance) uložených na několika serverech. Hunter proto začal vyvíjet novou aplikaci v jazyce Python „EEG viewer and analyzer“, která se postupem času změnila v dnešní knihovnu Matplotlib. [5]

V současné době se jedná o volně dostupný open-source projekt, sloužící k tvorbě statických, animovaných i interaktivních vizualizací nejčastěji ve 2D, knihovna však podporuje i trojrozměrné vizualizace. [6]

5.2 Seaborn

Seaborn vznikl jako řešení nejčastějších nedostatků knihovny Matplotlib, konkrétně defaultních nastavení vizualizace před verzí 2.0, která byla založena na vizualizacích MATLABu, nižší úrovně API u Matplotlibu, která často měla za následek nadbytečný kód a složitost využití datových struktur knihovny pandas. [9]

Seaborn buduje na základech knihovny Matplotlib, ke které poskytuje vysokoúrovňové rozhraní, také dokáže využívat datové struktury knihovny pandas. Seaborn na základě specifikace typu grafu zvládne automaticky provázat hodnoty v datech s vizuálními atributy, jako je barva, velikost a styl, propočítat statistické transformace a doplnit ke grafu informativní štítky a legendu. Díky tomu, že Seaborn dokáže vytvořit kompletní vizualizace jedním voláním funkce s minimálním počtem argumentů, je ideálním nástrojem pro explorační analýzu dat. [8]

5.3 Bokeh

Stejně jako Matplotlib je knihovna Bokeh open source projektem, finančně podporovaným neziskovou organizací NumFOCUS, která se zaměřuje na financování a propagaci, open source nástrojů ve vědě a výzkumu. [10] Oproti knihovnám Matplotlib a Seaborn však Bokeh nevytváří kompletní grafy jednou funkcí, místo toho nabízí širokou škálu nástrojů pro manipulaci s elementy vektorové grafiky (nazývanými „glyphs“), ze kterých lze následně vytvářet jednotlivé vrstvy vizualizace. To dává uživateli mnohem větší kontrolu nad vzhledem finální vizualizace, za cenu větší složitosti kódu, potřebného na její vytvoření. Bokeh dále dovoluje do vizualizace přidávat interaktivní elementy, jako slidery, tlačítka nebo drop-down menu. Knihovna také umožňuje vytvoření „Bokeh serveru“, který zajišťuje možnost streamování dat a složitější uživatelské interakce.

Knihovna se vlastně skládá ze dvou, Bokeh pro Python – sloužící k vytváření vizualizací a BokehJS – knihovna jazyka JavaScript sloužící k renderování vizualizace a zajištění interaktivity ve webovém prohlížeči. [11]

5.4 Plotly

Další open source knihovnou pro datovou vizualizaci je Plotly. Tato knihovna však není omezena pouze na jeden jazyk, lze ji používat v jazycích Python, R, Julia, MATLAB a existují i projekty zpřístupňující ji v jazyce Java a jazycích .NET frameworku. Implementace v jazyce Python je však její zdaleka nejoblíbenější variantou a existují zde dvě možnosti, jak knihovnu používat. Jednodušší vysokoúrovňový Plotly Express, který s minimem kódu vytváří kompletní vizualizace a složitější modul Plotly Graphic Objects, který uživateli poskytuje mnohem větší kontrolu nad vytvářenou vizualizací a je pomocí něho možné vytvořit některé typy grafů, nepodporované modulem Express. [12] Existují také komerční produkty založené na této knihovně, Dash Enterprise – platforma pro analýzu trhu, datové vědy a výzkum v oblasti umělé inteligence a Chart Studio Enterprise – nástroj pro rychlou tvorbu vizualizací přímo ze souborů nebo databází a jejich vkládání do webových stránek. [12]

5.5 Holoviews

Na rozdíl od předchozích vizualizačních knihoven, není knihovna Holoviews schopna sama vizualizovat data a je v tomto ohledu závislá na knihovnách Matplotlib, Bokeh a Plotly. Místo toho se Holoviews soustředí na co možná nejjednodušší manipulaci s grafy jako objekty, které jsou až do chvíle zobrazení nezávislé na jakékoliv vizualizační knihovně. Tyto objekty je také možné skládat do složitějších vizualizací pomocí seznamů a jednoduchých operátorů jako + (pro rozložení grafů vedle sebe) nebo * (pro překrytí grafů). Práce s daty je také zjednodušena, u většiny typů grafů stačí data popsat a předat konstruktoru objektu grafu informace o veličinách, které mají být vizualizovány. Dle popisu knihovny tato zjednodušení dovolují uživateli „*soustředit se na to, jakým způsobem chce data prozkoumat a co se snaží ukázat namísto samotného procesu tvorby grafu*“ [13]

5.6 Pygal

Knihovna Pygal je poslední z obecných vizualizačních knihoven jazyka Python popisovaných v této práci. Zaměřuje se převážně na jednoduchou tvorbu interaktivních vizualizací, ideálních pro vložení do webových stránek díky exportu

do formátu vektorové grafiky. Nabízí 14 různých typů grafů, včetně geografických vizualizací a možnosti přizpůsobení v podobě vestavěných stylů a možnosti definovat vlastní styly. [14]

5.7 MidiTime

Knihovna MidiTime neslouží ke klasické vizualizaci dat, naopak se zabývá oborem, který by se dal nazvat „sonifikace dat“. Jedná se o vyjádření dat rozložených v čase jako zvuk (v případě této knihovny jde o formát MIDI, který je možné využít v syntetizátorech a jiných elektronických nástrojích). Knihovnu vytvořil Michael Corey v roce 2015 za účelem znázornění počtu zemětřesení v americkém státě Oklahoma pro rádiové zpravodajství. [15] Takové znázornění dat by mohlo najít využití nejen pro média, jako je rádio nebo podcast, ale mohlo by přispět i k přístupnosti informací na webových stránkách pro lidi se zrakovým postižením nebo dokonce ke tvorbě hudby.

5.8 Geoplotlib

Ačkoliv některé z předcházejících knihoven podporují zobrazování dat na mapě, Geoplotlib se specializuje pouze na zobrazování geografických dat využívající OpenStreetMap jako podklad a OpenGL pro rychlé renderování grafických elementů na mapový podklad. Knihovna podporuje několik různých typů grafů jako body, teplotní mapy, histogramy, nebo i vykreslení tvarů v podobě geografických shapefiles a geoJSON. [16]

5.9 WordCloud

Poslední vizualizační knihovna se zabývá spíše estetickou než informativní vizualizací, konkrétně v podobě takzvaného Word cloudu (v češtině někdy také „Slovního mraku“). Jedná se o vizualizaci založenou na četnosti slov v textu. Čím častěji se slovo v textu vyskytuje, tím větší bude ve výsledném „mraku“. Kromě velikosti slov dovoluje knihovna WordCloud manipulovat s barvou slov (ať už na základě četnosti, délky, či jiného uživatelem definovaného pravidla) a s celkovým rozvržením „mraku“ pomocí masky založené na libovolném obrázku. [17]

6 Metodika zpracování

Cílem práce je předvést získávání dat a jejich zpracování v jazyce Python v podobě ukázek a praktických úloh, vyzkoušet použití jednotlivých vizualizačních knihoven a jejich následné porovnání. Hodnocení je rozděleno na dvě části, subjektivnější slovní hodnocení a jeho objektivnější kvantifikovanou podobu.

6.1 Hodnocení knihoven

Aby bylo možné využité knihovny snáze porovnávat, budou jejich funkcionality hodnoceny dle následující tabulky:

Tabulka 1 Ukázková tabulka hodnocení

Závislosti knihovny	Seznam knihoven, které jsou nutné pro fungovaní dané vizualizační knihovny
Podporované vstupní formáty	Formáty a datové struktury, které je knihovna schopná zpracovat
Podporované výstupní formáty	Formáty, ve kterých je možné vytvořené vizualizace ukládat
Poskytované typy grafů	Které typy grafů knihovna poskytuje? Existují nějaké často používané grafy, které v této knihovně chybí?
Poskytované možnosti přizpůsobení	Jakým způsobem lze vytvořené vizualizace upravovat?
Poskytované možnosti interaktivity	Jakým způsobem může uživatel s vytvořenou vizualizací interagovat?
Jednoduchost použití	Jaká je složitost potřebného kódu pro vytvoření a přizpůsobení vizualizace?
Přehlednost a atraktivita výchozích nastavení	Jak přehledná a vizuálně atraktivní je výsledná vizualizace při minimálním použití přizpůsobení?

Pro každou obecnou vizualizační knihovnu bude vytvořeno několik skriptů podle společného vzoru (složka Templates v kódech práce). Budou otestovány

podporované vstupní formáty, způsoby exportu vizualizace, nabízené možnosti přizpůsobení a hodnocení schopnosti rozlišit větší množství dat v jednom grafu. Aby bylo možné knihovny objektivněji porovnávat, budou tyto aspekty testovány pomocí často používaných typů grafů podporovaných všemi knihovnami, tedy graf sloupcový, spojnicový a bodový.

6.2 Kvantifikace hodnocení

Aby bylo možné porovnání knihoven vizualizovat, je nutné převést některé aspekty do číselného formátu. Velmi subjektivní části hodnocení, jako je „Jednoduchost použití“ a „Přehlednost a atraktivita výchozích nastavení“ nebudou zahrnuty. Procentuální hodnoty jsou v případě desetinných čísel zaokrouhleny. Body budou zaznamenány do následující tabulky:

Tabulka 2 Ukázková tabulka kvantifikovaného hodnocení

Kategorie	Bodové hodnocení	Procentuální hodnocení
Vstupní formáty	0-5 bodů	0-100 %
Výstupní formáty	0-3 body	0-100 %
Typy grafů	0-2 body	0-100 %
Možnosti přizpůsobení	0-2 body	0-100 %
Možnosti interaktivity	0-2 body	0-100 %
Celkové hodnocení	-	0-100 %

Body jsou přidělovány podle těchto pravidel:

Vstupní formáty

- Za každý z testovaných vstupních formátů je udělen jeden bod

Výstupní formáty

- Export v rastrovém formátu + 1 bod
- Export ve vektorovém formátu + 1 bod
- Export ve formátu .pdf + 1 bod

Typy grafů

- Pouze základní typy grafů = 0 bodů
- Některé specializované typy grafů = 1 bod
- Široký výběr různých typů grafů = 2 body

Možnosti přizpůsobení

- Žádné možnosti přizpůsobení = 0 bodů
  Knihovna nenabízí možnosti přizpůsobení vizualizace
- Minimální možnosti přizpůsobení = 1 bod
  Knihovna nabízí základní možnosti přizpůsobení, jako jsou barevné palety, či přiřazení barvy datové řadě
- Pokročilé možnosti přizpůsobení = 2 body
  Knihovna nabízí hlubší možností přizpůsobení, například definici vlastních barevných palet, nebo úpravy na úrovni jednotlivých grafických elementů

Možnosti interaktivity

- Žádné možnosti interaktivity = 0 bodů
  Nelze interagovat s vytvořenou vizualizací
- Minimální možnosti interaktivity = 1 bod
  Lze měnit zobrazovaná data, vybírat intervaly os, nelze však definovat vlastní chování
- Pokročilé možnosti interaktivity = 2 body
  Lze definovat vlastní chování pomocí uživatelského rozhraní, či událostí kliknutí, tažení apod.

7 Návrh a implementace vizualizačních úloh

7.1 Způsoby získávání dat pomocí jazyka Python

Kapitola popisuje různé způsoby získávání dat pomocí jazyka Python. Ke všem příkladům jsou zároveň vytvořeny soubory Jupyter Notebook, které jsou pro snadné prohlížení výsledků vyexportovány do formátu .pdf. Je tedy možné vidět výstupy programu bez nutnosti mít k dispozici zdrojová data nebo databázi.

7.1.1 Načtení dat ze souboru

Jedním ze způsobů získání dat pro vizualizaci je jejich načtení přímo z pevného disku počítače. Následující kapitoly popisují způsoby importu dat uložených v souborech běžných typů.

7.1.1.1 Sobory typu CSV

Soubory s příponou .csv (comma-separated values) jsou jednoduchým způsobem, jak uchovávat data organizovaná do sloupců a jsou hojně využívány například pro zveřejňování takzvaných „Otevřených dat“. Pro jejich načtení a zpracování je možné využít modul `csv` jazyka Python. Pomocí `csv.reader()` je vytvořen objekt, přes který lze iterovat v cyklu. Každá iterace vrací jeden řádek CSV souboru ve formátu seznamu (pokud soubor obsahuje hlavičku, nachází se na prvním místě a je s ní pracováno jako s jakýmkoliv jiným řádkem souboru). Následné zpracování dat je tedy poměrně náročné na množství potřebného kódu a při velkém množství dat i na rychlost.

Snazším a rychlejším způsobem práce se soubory ve formátu CSV je využití knihovny pandas. Pomocí volání `pd.read_csv()` lze ze souboru vytvořit DataFrame, se kterým lze dál snadno manipulovat. Není nutné soubor otevírat pomocí funkce `open()`, pro načtení stačí cesta. Důležitým rozdílem oproti csv modulu jazyka Python je nutnost specifikovat separátor pomocí `sep='<znak>'`, pokud je pro oddělení hodnot použit jiný znak než čárka.

7.1.1.2 Bitmapové soubory

Pro načtení obrazových dat ve formátech jako jsou .png, .jpg, apod. je možné použít knihovnu Python Imaging Library, respektive její aktuálně používanou verzi Pillow. Po načtení pomocí `PIL.Image.open()` získáme objekt typu Image, který lze dále zpracovávat. Dostupné jsou funkce jako ořezávání, změna velikosti, geometrické transformace a kombinace více souborů. Asi nejdůležitějšími jsou však metoda `.convert()` pro konverzi reprezentací pixelů mezi RGB, CMYK nebo převedení na odstíny šedi a možnost převést obrazová data na vícerozměrné pole knihovny NumPy pomocí `np.array()`. Tvar a uspořádání pole závisí na zvolené reprezentaci pixelů. V případě více kanálů jsou pixely polem hodnot, zatímco u obrazových dat konvertovaných na odstíny šedi jsou pixely tvořeny jedinou hodnotou. Takto vytvořená pole je možné dále zpracovávat manuálně, nebo využít knihoven jako je například SciPy. Zpracovaná data lze převést zpět na Image objekt pomocí `PIL.Image.fromarray()` a následně uložit, nebo vizualizovat pomocí vhodné knihovny, například Matplotlib (pro data ve formě odstínů šedi lze využít pseudobarev specifikací ColorMap).

7.1.1.3 Zvukové soubory

Vizualizace zvuku může být praktická v některých oblastech výzkumu, ale může sloužit i k čistě estetickým účelům. Knihovna Librosa je jednou z možných knihoven pro práci se zvukovými soubory. Kromě jejich načtení jako pole knihovny NumPy pomocí `librosa.load()` umí také provádět operace s načtenými daty, jako krátkodobé Fourierovy transformace, nebo dokonce generovat nové signály. Krom toho lze pomocí knihovny Librosa vizualizovat načtený zvuk jako průběh signálu v čase, nebo spektrogram. Vizualizace jsou založeny na knihovně Matplotlib a lze je provést pomocí `librosa.display.waveshow()` a `.specshow()`.

Krom této knihovny lze také využít SciPy, konkrétně `scipy.io.wavfile.read()`, který je schopen přečíst soubory ve formátu .wav a vrátit ho také jako NumPy pole. Výběr knihovny záleží převážně na způsobu dalšího zpracování dat.

7.1.2 Načtení dat z webové stránky

Hojně využívaným nástrojem pro práci s obsahem webových stránek (především výběr a zpracování dat, někdy také „web scraping“) je knihovna BeautifulSoup. Díky ní lze snadno procházet strukturu webové stránky, vyhledávat specifické elementy, třídy a identifikátory a extrahovat z nich text, nebo atributy (například atribut href v případě ele mentu odkazu může být užitečný pro další automatické procházení stránky).

V případě načítání tabulky z webové stránky není nutné procházet jednotlivé řádky a sloupce, stačí tabulku načíst pomocí BeautifulSoup a předat ji v podobě řetězce metodě `pd.read_html()` knihovny pandas, která z tabulky vygeneruje DataFrame. Pokud má tabulka hlavičku, pandas se pokusí tyto řetězce dosadit jako názvy sloupců.

7.1.3 Načtení dat z databáze

Způsoby získání dat z databáze se pochopitelně liší dle použitých technologií. Pro ukázku byly zvolena MySQL databáze, jako reprezentace relační databáze a nerelační (NoSQL) MongoDB.

MySQL

Pro interakci s libovolnou databází je nejprve nutné získat odpovídající ovladač. V případě MySQL se jedná o oficiální connector dostupný z webových stránek MySQL. Dalším krokem po importu ovladače je vytvoření spojení s databází, zde pomocí `mysql.connector.connect()`, kde je nutné doplnit parametry s adresou, uživatelským účtem a databází. Dotazy na databázi jsou pak vytvářeny jako string, do kterého lze doplnit zástupné symboly `%s` pro pozdější doplnění parametrů, například z uživatelského vstupu. Hodnoty načítané z databáze jsou vraceny jako uspořádané n-tice (Tuple) představující jeden řádek, popřípadě jejich seznam.

MongoDB

Pro připojení k MongoDB databázi je vhodné využít ovladače `pymongo`. Podobně jako u MySQL je prvním krokem vytvoření spojení s databází, zde pomocí `pymongo.MongoClient()`, kde argumentem je „connection string“ obsahující adresu, uživatelské jméno a databázi k otevření. Každá databáze MongoDB se může skládat

z mnoha kolekcí. Výběr databáze i kolekce lze provádět buď pomocí tečkové notace (`client.nazev_databaze` a `nazev_databaze.nazev_kolekce`), nebo pomocí (`client["nazev_datanaze"]`) v případě, že název by nebylo možné zapsat do kódu. Operace s kolekcemi se pak provádí voláním odpovídajících funkcí, jako `.find()`, `.insert_one()`. Záznamy z databáze vrácené po vyhledání se chovají jako slovníky, lze tedy snadno přistupovat k jejich atributům.

7.1.4 Načtení dat pomocí API

Pro dotazy na Application Programming Interface (API) přes HTTP protokol slouží modul jazyka Python `requests`. Získání dat pak probíhá pomocí metody `requests.get()` kde argumentem je url dotazu (pro tvorbu je vždy nutné konzultovat dokumentaci dotazovaného rozhraní). Vrácen je objekt Response, který nese data, a některé doplňující informace jako kódování, čas potřebný pro dokončení dotazu a asi nejdůležitěji HTTP stavový kód (200 v případě úspěšného dokončení). Aplikační rozhraní s architekturou REST, vrací odpovědi v podobě JSON nebo vzácněji XML dokumentu. K datům ve formátu JSON lze pak v jazyce Python snadno přistupovat pomocí indexů hodnot, protože se chová jako slovník.

7.2 Ukázkové úlohy

Pro odzkoušení funkčnosti jednotlivých vizualizačních knihoven byly navrženy zadání testovacích úloh kombinující různé druhy získávání, zpracovávání a vizualizace dat. Vytvořené ukázky kódu představují jejich možné řešení. Některé aspekty zadání jsou ponechány jako volitelné.

7.2.1 Vizualizace dat získaných pomocí REST API knihovnou Matplotlib

Zadání: „Pomocí knihovny Matplotlib vizualizujte vhodná kategorická data získaná pomocí zvoleného REST API. Využijte sloupcového grafu, aplikujte vlastní barevné schéma.“

Ukázkové řešení: TmxREST.py

V ukázkovém řešení jsou vizualizovány časy jezdců z kompetitivní závodní hry „Trackmania“, dostupné na webové stránce Trackmania Exchange

(https://trackmania.exchange/). Pomocí jednoduchého rozhraní je od uživatele získán identifikátor tratě k vizualizaci. Následně jsou pomocí REST API načteny informace o rekordech na trati a popis tratě samotné. Barevné schéma je tvořeno zlatou, stříbrnou a bronzovou barvou pro první tři pozice, zbylé sloupce jsou obarveny šedou barvou. Po vytvoření objektu grafu je doplněn nadpis s názvem trati a pro každý sloupec doplněn popisek s časem v sekundách. Výsledná vizualizace je zobrazena v samostatném okně, je tedy možné zadat nový identifikátor a vizualizovat více tratí.

7.2.2 Vizualizace EXIF metadat knihovnou Matplotlib

Zadání: „Pomocí knihovny Matplotlib vizualizujte ohniskové vzdálenosti fotografií. Předpokládají se soubory ve formátu .jpg. Využijte vhodný typ grafu."

Ukázková řešení: ExifData.py a ExifData_Bar.py

Pomocí jednoduchého dialogu pro výběr složky z modulu Tkinter je získána cesta, na které jsou následně nalezeny všechny soubory typu .jpg. Je vytvořen seznam všech ohniskových vzdáleností postupným čtením EXIF informací. K přečtení je využito knihovny Python Image Library. Seznam intervalů byl definován nerovnoměrně dle konzultace s fotografem, ponechat intervaly rovnoměrné by však nebylo chybou. V samotném bodu vizualizace se řešení liší. V ExifData.py je využit graf typu histogram z knihovny Matplotlib, ten však přizpůsobuje šířku sloupců dle velikosti intervalu na ose, to znamená, že některé sloupce jsou velmi úzké a těžko čitelné. Tento problém je řešen v ExifData_Bar.py, kde je histogram vytvořen pomocí funkce `np.histogram()` a je následně vizualizován jako standardní sloupcový graf.

7.2.3 Získání textu z webové stránky a jeho vizualizace

Zadání: „Pomocí BeautifulSoup, nebo jiné knihovny pro čtení webových stránek získejte text ze stránky a vizualizujte četnost slov ve formě „mraku". V datech by se neměly vyskytovat části stránky jako menu, nebo zápatí. Volitelně uspořádejte vizualizaci pomocí masky."

Ukázková řešení: NMNM.py a WordCloudJP.py

Při tvorbě vizualizace obsahu oficiálních stránek Nového Města nad Metují jsou nejprve nalezeny všechny odkazy pomocí knihovny BeautifulSoup. Poté je

na každé takto nalezené stránce vybrán element typu div s textovým obsahem. Takto je odfiltrován obsah záhlaví, zápatí, menu a jiných součástí stránky, které by se ve výsledném textu opakovaly. Pomocí knihovny PIL je vytvořeno pole definující masku pro výsledný mrak, díky které bude mít tvar novoměstského zámku. Ačkoliv knihovna WordCloud zvládne vynechávat spojky a částice v textu, děje se tak jen pro anglický text. Před vizualizací je tedy nutné definovat pole přeskočených slov manuálně. Na výslednou vizualizaci je aplikována funkce přebarvující jednotlivá slova dle grafického manuálu města.

Obr. 11 Výsledná vizualizace byla využita jako náhled pro záznam zasedání zastupitelstva města
Zdroj: https://www.novemestonm.cz/

7.2.4 Geografická vizualizace pro bodová data

Zadání: „Pomocí knihovny určené k vizualizaci geografických dat (např. GeoPlotLib nebo Plotly) vizualizujte soubor bodových dat dle vlastního výběru (zastávky hromadné dopravy, dopady meteoritů apod.)“

Ukázková řešení: NASAMeteorite_GeoPlotLib.py, NASAMeteorite_Plotly.py, NASAMeteoriteHeatmap_GeoPlotLib.py a NASAMeteoriteHeatpam_Plotly.py

Tato vizualizace využívá volně dostupných dat poskytovaných agenturou NASA [34] ve formátu .csv. Zpracování je tedy velmi jednoduché, jedná se pouze o načtení řádků ze souboru a vytvoření objektu DataFrame (v případě knihovny GeoPlotLib je z DataFrame dále vytvořen DataAccessObject). Vizualizace jsou pro obě knihovny vytvořeny jako bodová mapa s popisky a takzvané „teplotní mapy“, která znázorňuje hustotu nalezených meteoritů. Zajímavostí datového souboru je dopad, který se při vizualizaci knihovnou GeoPlotLib vyskytuje mimo mapu, označený jako „Meridiani Planum“, tedy lokace nacházející se na Marsu. Několik dalších bodů se nachází mimo mapu v oblasti Antarktidy, nejspíše kvůli deformaci u okrajů mapy.

7.2.5 Vizualizace dat z SQL databáze zákazníků

Zadání: „Firma provozující obchod s oblečením si udržuje databázi svých zákazníků, ve které jsou zaznamenány základní informace, jako pohlaví, věk, jestli vlastní zákaznickou kartu apod. Načtěte data pomocí SQL dotazu a vizualizujte je pomocí libovolné vizualizační knihovny.“

Ukázková řešení: Složky MarketingNoDb a MarketingSQL. Složka NoDb obsahuje identické vizualizace, ale data nejsou načítána z databáze, aby bylo možné je lokálně spustit.

Úloha předpokládá využití MySQL databáze, k připojení je tedy použit `mysql.connector`. Získání a zpracování dat z databáze je ve všech případech stejné, dotaz z databáze vrátí seznam uspořádaných n-tic (i v případě, že se jedná o jeden sloupec), je tedy převeden na seznam hodnot funkcí `untuple()` (tato funkce se nevyskytuje ve variantě programů bez databáze). Následné zpracování těchto seznamů již závisí na použité vizualizační knihovně. V případě Matplotlib stačí jen ze seznamu velikostí vytvořit slovník s četností pomocí objektu `Counter()` a seřadit ho od nejpočetnější velikosti pomocí funkce `.most_common()`. Výsledné seznamy jsou pak použity při volání funkcí pro samotnou tvorbu grafů. Obdobně je tomu u knihovny Seaborn. Zde je pouze knihovnou nepodporovaný koláčový graf nahrazen houslovým, s dodatečným rozdělením zákazníků podle toho, jestli vlastní zákaznickou kartu. Velmi jednoduché zpracování je také v případě knihovny Plotly, která se liší především kompaktnější definicí jednotlivých grafů. U knihovny Pygal je pak využito tvorby histogramu pomocí `np.histogram()`, který je následně

pro porovnání vizualizován jako sloupcový graf i graf typu histogram poskytovaný knihovnou, rozdílem mezi těmito grafy je podobně jako u knihovny Matplotlib variabilní šířka sloupce u histogramu. Největší rozdíly jsou u knihovny Bokeh, kde grafy jako histogram a koláčový graf jsou tvořeny z jednotlivých grafických elementů. Histogram je tvořen z „quads“ a je nutné u nich specifikovat všechny rozměry, spodní strana je 0 a dotýká se tak osy x, horní odpovídá hodnotě na ose y, levá a pravá hrana je pak získána z hranic intervalů. Koláčový graf je vytvořen z kruhových výsečí, je pro ně tedy dopočítán úhel v radiánech.

7.2.6 Geografické vizualizace pro státy

Zadání: „Vizualizujte data o produkci automobilů evropských států. Barva státu by měla vyjadřovat produkci.“

Ukázková řešení: GeoPlotCars.py, PlotlyCars.py

V případě GeoPlotCars je využito knihovny GeoPlotLib, dat ve formátu JSON a souboru s hranicemi států ve formátu GeoJSON. Tento soubor také obsahuje název země, díky kterému je možné k ní přiřadit produkci z datového souboru, země jsou tedy vyplněny barvou, která odpovídá této produkci. Knihovna GeoPlotLib má však problém s vyplněním složitějších území definovaných GeoJSON souborem a často jsou tak hranice nepřesné. Nejspíše se nejedná o chybu v definici hranic, ale v knihovně samotné. Pokud nejsou území vyplněna a jsou zobrazeny pouze hranice, nedochází k žádným chybám. Chyby při vyplňování jsou viditelné i na ukázkových projektech knihovny, ačkoliv jsou kvůli tvaru použitých států USA méně patrné.

V druhém ukázkovém řešení je využito knihovny Plotly Express a zjednodušené mapy světa, kterou poskytuje. Tvary území jsou součástí samotné knihovny a není tedy třeba využívat externích GeoJSON souborů. Data jsou identifikována podle kódů států definovaných normou ISO 3166-1.

7.2.7 Vizualizace využití hardware v reálném čase

Zadání: „Vizualizujte využití hardwarových zdrojů, jako je procesor, či paměť v reálném čase pomocí vhodně zvolené knihovny.“

Ukázkové řešení: BokehStreaming.py

V této vizualizaci je využito schopnosti knihovny Bokeh přidat ke grafům periodicky volanou funkci a měnit zobrazovaná data za běhu programu. Po vytvoření základních datových struktur a jednotlivých grafů je každých 100 ms volána funkce `update()`, která pomocí balíčků psutil a GPUtil načte aktuální stav procesoru, paměti, grafické karty a přenesených paketů přes síť a přidá hodnoty do již vytvořených slovníků. Díky definovanému maximálnímu počtu, se po dosažení 50 nebo 100 hodnot v souboru přestanou data přidávat a jsou postupně nahrazována, graf si tedy uchovává stejné měřítko. Pro zobrazení této vizualizace je nutné ji spustit jako Bokeh Server příkazem v terminálu.

7.2.8 Vizualizace zvuku ve formátu wave

Zadání: „Vizualizujte zvukový soubor ve formátu wave. Výsledný graf může být jak spektrogram, tak průběh signálu (waveform).“

Ukázkové řešení: LibrosaMusicVisualisation.py

Podobně jako vizualizace dat získaných z EXIF informací je prvním krokem tohoto programu zobrazení dialogu s výběrem složky a následné získání cest ke všem souborům formátu .wav, které se v ní nachází. Soubory jsou následně zpracovány knihovnou Librosa pomocí krátkodobé Fourierovy transformace a je vytvořen spektrogram s měřítkem v decibelech. Na základě těchto dat jsou dále vytvořeny grafy, ke kterým je doplněn název souboru. Během zpracovávání dat je do konzole vypisován aktuální postup, aby byl uživatel informován o průběhu operací. Nakonec je z vytvořených grafů vytvořen .pdf soubor pro snazší prohlížení a případné sdílení výsledků.

7.2.9 Vizualizace struktury složek na pevném disku

Zadání: „Vytvořte vizualizaci znázorňující strukturu složek na pevném disku počítače. Do vizualizace zahrňte informaci o velikosti jednotlivých složek.“

Ukázkové řešení: FolderTreeStatsPygal.py

Pro tento typ vizualizace je ideální graf typu „stromová mapa“ (ang. TreeMap), jednou z knihoven schopných této vizualizace je Pygal. U této knihovny je také možné využít snadný způsob vytváření popisků pro data, výsledný graf tak bude obsahovat informaci o názvu složky, jejím umístění a její velikost v megabytech.

Získání dat probíhá pomocí rekurzivního sčítání velikostí souborů jednotlivých podsložek. Funkce sloužící k tomuto účelu, `get_size()`, má na vstupu příznak, který určuje, zda by měla vrátit pouze velikost souborů ve složce, či velikost souborů ve složce a všech jejích podsložkách. Díky tomu lze v grafu zobrazit dvě úrovně zanoření od uživatelem specifikované složky.

7.2.10 Interaktivní vizualizace matematické funkce

Zadání: „Vytvořte interaktivní vizualizaci libovolné matematické funkce. Uživatel by měl mít možnost měnit parametry funkce“

Ukázkové řešení: BokehInteractive.py

Matematická funkce v této ukázce je

$$x_{n+1} = rx_n(1 - x_n)$$

S počátečním parametrem r = 3 a hodnotou x = 0,5. Je vypočítáno 100 následujících hodnot funkce. Zobrazená vizualizace je doplněna o dva widgety typu „slider“ pro úpravu hodnoty r a x. Pokud je uživatelem změněna hodnota jednoho ze sliderů, je volána funkce `update_values()`, která obstarává výpočet nových 100 hodnot k zobrazení.

7.2.11 Vizualizace dat získaných z aplikace pro sledování aktivity

Zadání: „Vytvořte vizualizaci pro data o aktivitě získaná z mobilního telefonu, chytrých hodinek, nebo podobných zařízení.“

Ukázkové řešení: SamsungHealthHoloviews.py

Pro vizualizaci jsou využita data vyexportovaná z aplikace Samsung Health. Prvním krokem je získání cesty ke složce s daty od uživatele. Program předpokládá, že získaný .zip archiv byl již extrahován. Následně jsou nalezena data pro tep, saturaci kyslíku a počet kroků za den. Pro každý z nalezených souborů je vytvořen DataFrame knihovny pandas, u kterého jsou dlouhé názvy sloupců nahrazeny novými, pro člověka čitelnými. Dále jsou data seřazena dle dne, kdy byla zaznamenána a u sloupce s časem je změněn datový typ na „datetime“. Nakonec jsou tyto DataFrames předány k vizualizaci knihovnou Holoviews. Program zvládne také upravit soubory, ve kterých se může nacházet nadbytečná hlavička, která by

způsobila chybu při načtení knihovnou pandas. V případě, že došlo k problémům při zpracování dat, či daný soubor nebyl nalezen, jsou odpovídající grafy přeskočeny.

7.2.12 Vizualizace stavu českých řek pomocí Jupyter Notebook

Zadání: „Na základě dat dostupných z webových stránek Povodí Labe vytvořte vizualizaci stavů, nebo průtoků jednoho či více toků. Využijte prostředí Jupyter Notebook.“

Ukázkové řešení: PLARiverVisualisation.ipynb

Program nejprve pomocí knihovny BeautifulSoup získá odkazy na jednotlivé měřící stanice. Poté na každé stránce nalezne tabulku s hodnotami pro výšku hladiny a průtok, ta je načtena do objektu DataFrame pomocí metody `read_html()`, neoznačený sloupec obsahující datum a čas je přejmenován a převeden do správného formátu. Takto vytvořený DataFrame je poté vizualizován knihovnou Plotly. Grafy jsou nakonec zobrazeny do Notebooku.

7.2.13 Vizualizace dat pomocí zvuku

Zadání: „Pokuste se navrhnout a implementovat způsob datové vizualizace pro uživatele se zrakovým postižením.“

Ukázkové řešení: TemperatureMIDI_Pygame.py a TemperatureMIDI_Timidity.py

Tento program využívá knihovny MIDITime pro převedení dat předpovědi počasí získaných z webového rozhraní na noty, které následně zapíše do formátu MIDI. Ukázková řešení se v tomto případě liší jen způsobem následného přehrání tohoto souboru. Kromě knihoven Pygame a Timidity by bylo možné výsledný soubor využít i pro tvorbu hudby ve specializovaném software.

7.3 Porovnání vizualizačních knihoven

7.3.1 Matplotlib

7.3.1.1 Závislosti

Knihovna Matplotlib má následující závislosti:

- FreeType –sloužící k renderování fontů
- libpng –pro manipulaci se soubory typu png
- NumPy – knihovna pro práci s daty ve formátu vícerozměrných polí
- cycler – objekt pro nekonečné cyklické procházení seznamů
- dateutil – rozšíření DateTime modulu jazyka Python
- kiwi – rychlé řešení soustav rovnic a nerovnic

7.3.1.2 Vstupní formáty

Matplotlib zvládne data zpracovávat jako standardní seznamy a uspořádané n-tice (tuple) jazyka Python i pole knihovny Numpy. Objekty typu DataFrame z knihovny pandas zvládne zpracovat přímo pouze pro některé typy grafů, u jiných (například u sloupcového grafu), lze volat metodu `.plot()` tohoto objektu, která vrátí objekt Axes, který lze snadno vložit jako podgraf do vytvářené vizualizace. Pro vytvoření grafu nelze přímo použít slovník jazyka Python, je nutné jeho části předat jako seznamy.

7.3.1.3 Výstupní formáty

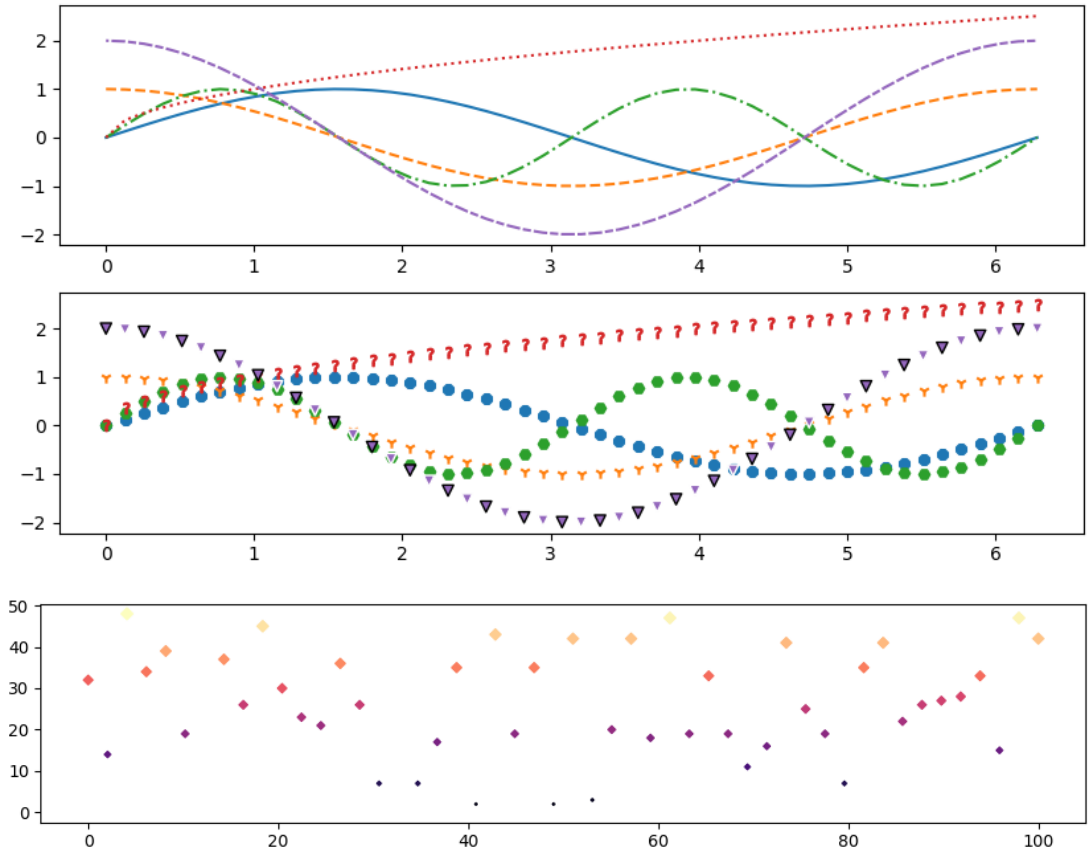
Knihovna Matplotlib zvládne vytvořené vizualizace zobrazovat do okna, které nabízí uživateli určitou míru interaktivity (přibližování grafů, pohyb s osami, úprava podgrafů), export do souboru lze provést přímo z kódu voláním metody `plt.savefig()`, nebo z již zmiňovaného okna přes souborový dialog. Matplotlib nabízí široký výběr formátů, od často používaných jako jsou .png, .jpeg, .svg, .pdg až po vzácnější formáty, například .raw, .rgba a .tiff.

7.3.1.4 Typy grafů

Podporovány jsou nejrůznější typy dvourozměrných vizualizací od jednoduchých grafů po vizualizace založené na obrazových datech, ale i specializované typy grafů jako je Sankeyův diagram. Vytvářet lze i velké množství trojrozměrných vizualizací. V případě, že nabízené typy vizualizací nejsou dostačující, lze vytvářet i vlastní skládáním existujících vizualizací a manipulací se základními geometrickými tvary.

7.3.1.5 Přizpůsobení vizualizací

Možnosti přizpůsobení vizualizací jsou velmi hluboké, od rychlého přepnutí celkového stylu grafu, přes jednoduché změny velikosti a barvy objektů při vytváření grafu, až po možnost jednotlivě manipulovat s libovolným elementem hotové vizualizace. Je možné snadno měnit barvu a velikost elementů v závislosti na hodnotách, díky speciálním argumentům a tzv. „colormaps“, které definují gradient barev, ze kterého lze vybírat na základě zobrazovaných dat. Některé možnosti přizpůsobení mohou být někdy mírně nekonzistentní ve způsobu jejich použití, například při vytváření sloupcového grafu, lze nastavit barvy jednoduchým doplněním seznamu barev do argumentu „color“, stejným způsobem však nelze použít seznam šrafování pro argument „hatch“, ten přijímá pouze jeden typ šrafování. Pro použití různých šrafování je nutné vytvořit více grafů v jednom diagramu, nebo přistupovat přímo k vlastnost „hatch“ u jednotlivých grafických elementů grafu.



Obr. 12 Možnosti rozlišení hodnot v knihovně Matplotlib
Zdroj: vlastní zpracování

7.3.1.6 Interaktivita

Jak již bylo zmíněno, určitá míra interaktivity je možná pomocí zobrazení v okně knihovny Matplotlib. Pro hlubší interaktivitu je možné využít událostí, které dovolují detekovat klikání, pohyb a tažení myši a reagovat na ně. Je také možné opakovaně volat „animační funkci“, pozměňující zobrazovaná data a vytvářet tak animované grafy.

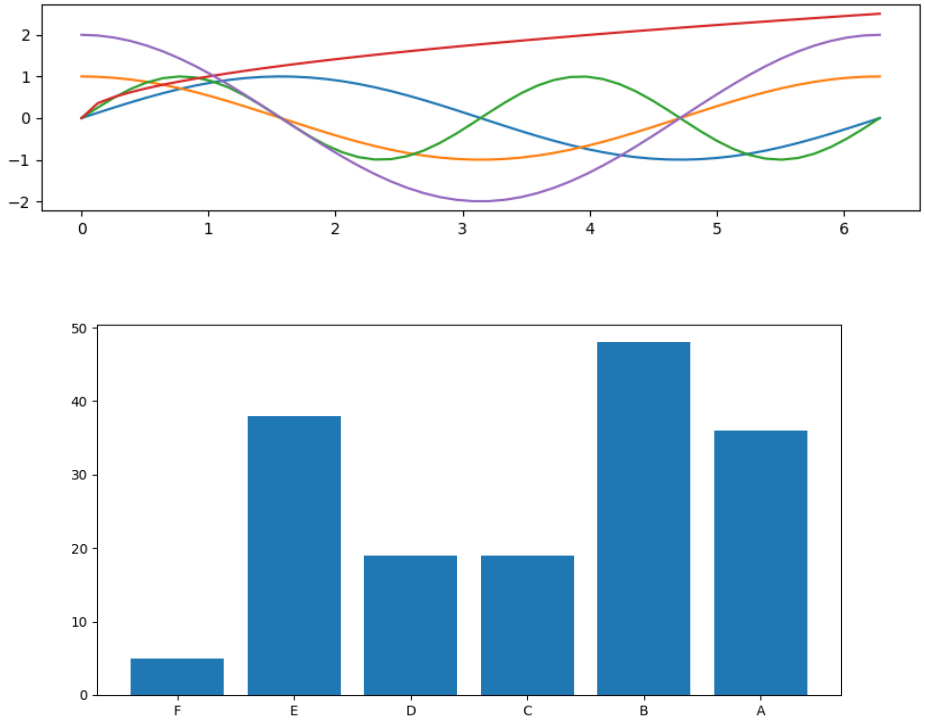
7.3.1.7 Jednoduchost použití

Knihovna Matplotlib je velmi jednoduchá na použití, k vytvoření vizualizace obvykle není třeba více než jen několik řádků kódu. Až na vzácné případy, jako například šrafování popsané v dřívější kapitole, se způsob volání metod a dosazování argumentů řídí jednoduchými a snadno pochopitelnými pravidly. Ke knihovně je dostupná obsáhlá a velmi detailně zpracovaná dokumentace popisující všechny

její součásti, od grafů jako celku až po vlastnosti a metody jednotlivých částí grafu a grafických elementů. V případě, že by informace v dokumentaci nebyly dostačující, je díky rozšířenosti této knihovny velmi snadné je dohledat na jiných webových stránkách.

7.3.1.8 Výchozí nastavení

Výchozí nastavení knihovny pro většinu typů grafů je dostačující k nalezení některých závislostí mezi zobrazovanými daty, ale v mnoha případech nejsou příliš atraktivní pro čtenáře, vzhled lze však snadno změnit výběrem z mnoha zabudovaných stylů.



Obr. 13 Výchozí vzhled grafů knihovny Matplotlib
Zdroj: vlastní zpracování

7.3.1.9 Shrnutí

Tabulka 3 Hodnocení knihovny Matplotlib

Závislosti knihovny	FreeType, libpng, NumPy, cycler, Dateutil, kiwi
Podporované vstupní formáty	Seznam, tuple, Numpy pole, pandas DataFrame (u některých grafů)
Podporované výstupní formáty	png, jpeg, svg, pdf, pgf, ps, tiff, raw, rgba
Poskytované typy grafů	Velmi široký výběr 2D a 3D grafů, možnost tvorby vlastních vizualizací
Poskytované možnosti přizpůsobení	Široké možnosti úprav všech elementů, colormaps, předem definované styly, anotace a legendy
Poskytované možnosti interaktivity	Handlery pro události myši a klávesnice, periodicky volané funkce pro animace
Jednoduchost použití	Snadno pochopitelný zápis kódu, velmi dobře zpracovaná dokumentace
Přehlednost a atraktivita výchozích nastavení	Výchozí vizualizace nejsou příliš atraktivní, nicméně jsou pro mnoho účelů dostačující

Tabulka 4 Kvantifikované hodnocení knihovny Matplotlib

Kategorie	Bodové hodnocení	Procentuální hodnocení
Vstupní formáty	4 / 5 bodů	80 %
Výstupní formáty	3 / 3 bodů	100 %
Typy grafů	2 / 2 bodů	100 %
Možnosti přizpůsobení	2 / 2 bodů	100 %
Možnosti interaktivity	2 / 2 bodů	100 %
Celkové hodnocení	-	**96 %**

7.3.2 Seaborn

7.3.2.1 Závislosti

Seaborn je vizualizační knihovna, založená na knihovně Matplotlib, má tedy i všechny její závislosti. Krom těchto knihoven dále vyžaduje:

- pandas – knihovna pro manipulaci s daty ve formě tabulek
- SciPy – rozšíření knihovny NumPy

7.3.2.2 Vstupní formáty

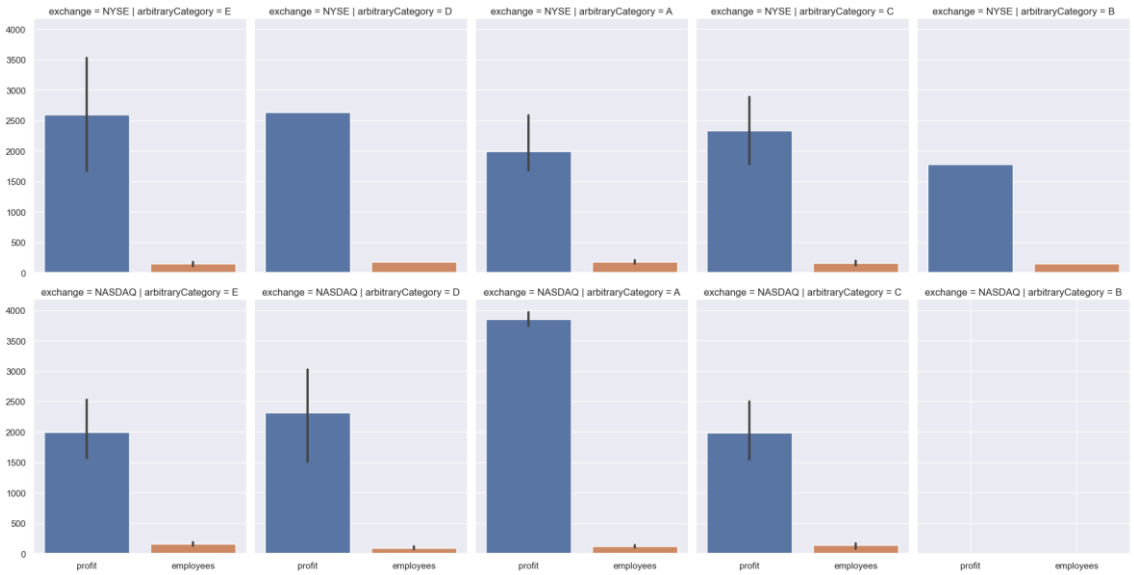
Oproti knihovně Matplotlib zvládne Seaborn lépe pracovat s daty ve formátu slovníků a DataFrame knihovny pandas. Pro tyto vstupy se při volání metody vytvářející vizualizaci daný objekt dosadí do parametru „data“ a definují se klíče (v případu slovníku) nebo názvy sloupců (u DataFrame) do parametrů „x“ a „y“ pro definici os. U některých typů vizualizací lze dále využít „hue“ a „size“ pro vyjádření dalších dat pomocí odstínu a velikosti grafických elementů. Zvláštností je, že uspořádané n-tice (Tuple) nelze přímo využít při volání vizualizační metody a musí být přetypovány na seznam, ačkoliv tak lze učinit v knihovně Matplotlib.

7.3.2.3 Výstupní formáty

Export vizualizací je založen na exportu z knihovny Matplotlib voláním `plt.savefig()`. Formáty exportu jsou tedy identické s touto knihovnou.

7.3.2.4 Typy grafů

Seaborn umožňuje generovat mřížky vizualizací třech základních typů: vztahu více proměnných (metoda `.relplot()`), rozložení hodnot (meotda `.displot()`) a hodnoty rozložené do kategorií (metoda `.catplot()`). Krom toho lze využít metod, které vytváří jeden specifický typ grafu. Jedním z často využívaných grafů, který v této knihovně chybí je koláčový graf a jeho varianty.



Obr. 14 Vizualizace pomocí FacetGrid knihovny Seaborn. Firmy jsou děleny do řádků dle burzy a sloupců dle kategorie
Zdroj: vlastní zpracování

7.3.2.5 Přizpůsobení vizualizací

Pro přizpůsobení vytvářených vizualizací lze využít všech parametrů a funkcí dostupných v knihovně Matplotlib. Seaborn samotný navíc umožňuje rychlý výběr vzhledů a okolo 180 barevných palet s možností definice vlastních a úpravy existujících.

7.3.2.6 Interaktivita

Seaborn nerozšiřuje interaktivitu knihovny Matplotlib, jsou tedy dostupné stejné funkce a události.

7.3.2.7 Jednoduchost použití

Seaborn zjednodušuje práci hlavně s objekty typu DataFrame a umožňuje rychle a jednoduše porovnávat obsáhlé soubory data pomocí FacetGrid vizualizací, které lze vytvořit jediným řádkem kódu. Díky základu v knihovně Matplotlib je způsob tvorby vizualizací snadno pochopitelný, pokud uživatel již má zkušenosti s touto knihovnou.

7.3.2.8 Výchozí nastavení

Výchozí vzhled Seaborn připomíná knihovnu Matplotlib. Pro lépe vypadající vizualizace lze zvolit styl voláním `.set_theme()`. Pokud není specifikován parametr, je nastaven „výchozí“ vzhled knihovny Seaborn, který se až do verze 0.8 automaticky aplikoval při importu knihovny.

7.3.2.9 Shrnutí

Tabulka 5 Hodnocení knihovny Seaborn

Závislosti knihovny	SciPy, pandas, závislosti knihovny Matplotlib
Podporované vstupní formáty	Seznam, slovník, NumPy pole, Pandas DataFrame,
Podporované výstupní formáty	png, jpeg, svg, pdf, pgf, ps, tiff, raw, rgba
Poskytované typy grafů	Většina základních 2D grafů kromě koláčového
Poskytované možnosti přizpůsobení	Stejné jako u knihovny Matplotlib, rozšířené o palety a styly
Poskytované možnosti interaktivity	Stejné jako Matplotlib
Jednoduchost použití	Stejné jako Matplotlib. Snadno pochopitelná a přehledná dokumentace
Přehlednost a atraktivita výchozích nastavení	Díky jednoduché aplikaci stylů je atraktivnější než u knihovny Matplotlib

Tabulka 6 Kvantifikované hodnocení knihovny Seaborn

Kategorie	Bodové hodnocení	Procentuální hodnocení
Vstupní formáty	4 / 5 bodů	80 %
Výstupní formáty	3 / 3 bodů	100 %
Typy grafů	1 / 2 bodů	50 %
Možnosti přizpůsobení	2 / 2 bodů	100 %
Možnosti interaktivity	2 / 2 bodů	100 %
Celkové hodnocení	-	**86 %**

7.3.3 Bokeh

7.3.3.1 Závislosti

K použití knihovny Bokeh jsou vyžadovány následující závislosti:

- Jinja2 – knihovna pro vytváření šablon
- NumPy – knihovna pro práci s daty ve formátu vícerozměrných polí
- packaging – knihovna nástrojů pro balíčky jazyka Python
- pillow – knihovna pro práci s obrazovými formáty
- PyYAML – parser serializačního jazyka YAML
- tornado – framework pro webové aplikace
- typing_extensions – zpřístupnění modulu typing pro starší verze jazyka Python

7.3.3.2 Vstupní formáty

Knihovna Bokeh bez problému zpracovává nejrůznější vstupní formáty, ať už přímým zadáním dat ve formě seznamů, polí a n-tic, nebo využitím pojmenovaných dat a argumentu `source`, kterým lze předat knihovně slovník, DataFrame, nebo i objekt ColumnDataSource, který knihovna nabízí. Využitím tohoto nového objektu lze snadno sdílet data mezi grafy, přidávat i odebírat sloupce, či aktualizovat data v reálném čase.

7.3.3.3 Výstupní formáty

Jak již bylo zmíněno v předchozích kapitolách, Bokeh se skládá ze dvou části, vizualizační knihovny pro jazyk Python a BokehJS, který zajišťuje renderování a interaktivitu v prohlížeči. Hlavním výstupním formátem této knihovny jsou tedy soubory .html. Možný je také export do formátů .png a .svg, který však ke svému fungování potřebuje dodatečné závislosti v podobě knihovny selenium a ovladače (např. geckodriver prohlížeče Firefox). Vytvořené vizualizace lze také spustit v podobě serveru, což dovoluje značně rozšířit interaktivitu vizualizací.

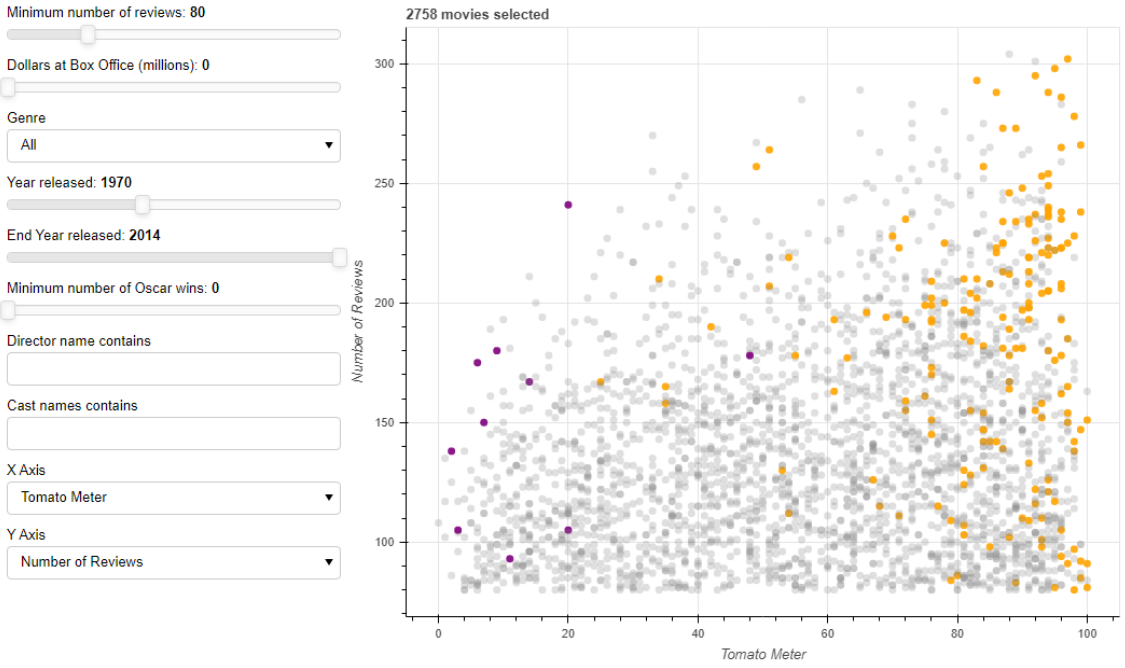
7.3.3.4 Typy grafů

V rámci objektu Figure lze volat funkce pro vytvoření konkrétních vizualizací, jako sloupcového, spojnicového, či bodového grafu. Hlavním způsobem tvorby vizualizací pomocí knihovny Bokeh je však manipulace základních grafických elementů (například pro vytvoření koláčového grafu je nutno využít grafického elementu „Wedge“ a výpočtu správných úhlů na základě dat. Obdobně pro tvorbu histogramu by bylo použito elementů „Quad“, nebo „Patch“).

7.3.3.5 Přizpůsobení vizualizací

Díky možnosti libovolně manipulovat s grafickými elementy vizualizace dovoluje knihovna Bokeh velmi široké možnosti přizpůsobení. Pokud je však důraz kladen na rychlost tvorby grafů, je lepší volbou využít předpřipravených stylů a palet (lze i definovat vlastní pro použití ve více vizualizacích, či v celé organizaci).

Pro vyjádření více proměnných ve vizualizaci jsou k dispozici funkce jako `factor_cmap()` nebo `factor_hatch()`, dovolující přiřadit barvu či typ šrafování různým hodnotám.



Obr. 15 Pomocí knihovny Bokeh lze vytvářet velmi složité interaktivní vizualizace
Zdroj: Dokumentace knihovny Bokeh [11]

7.3.3.6 Interaktivita

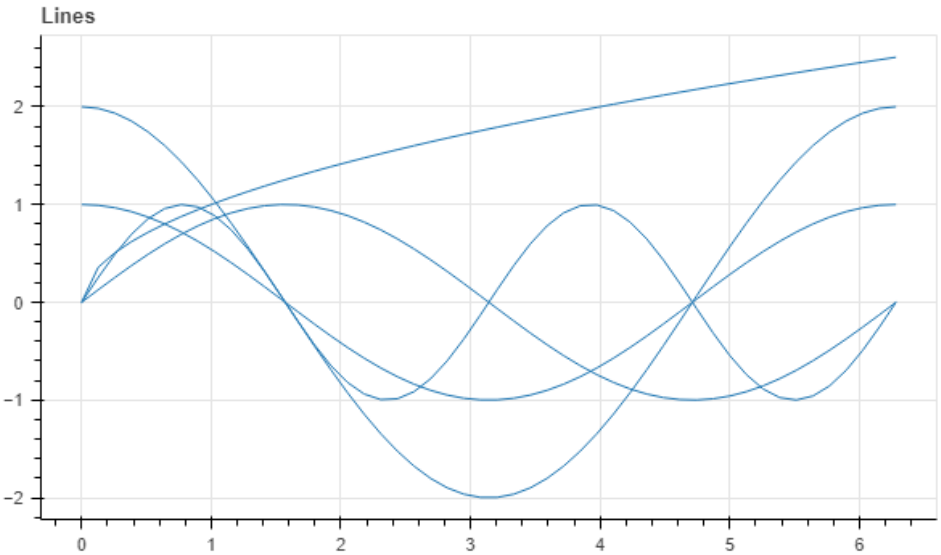
Pro vytváření interaktivních vizualizací je možné přidat do výsledné stránky nejrůznější „widgety“, jako tlačítka, slidery, textová pole, nebo menu. V případě, že jsou tyto komponenty nevyhovující, lze také vkládat vlastní widgety vytvořené v jazyce JavaScript.

7.3.3.7 Jednoduchost použití

Oproti předcházejícím knihovnám Matplotlib a Seaborn je Bokeh mnohem složitější a vyžaduje větší množství kódu ke tvorbě vizualizace. Ke každé součásti knihovny je však k dispozici rozsáhlá dokumentace včetně ukázek. Všechny objekty a funkce mají také vyplněné detailní informace o jejich správném použití, které je možné zobrazit přímo ve vývojovém prostředí. V některých případech jsou tyto příklady použití zahrnuty i do chybových zpráv, je tedy jednoduché opravit chybu vzniklou špatným použitím některé z funkcí.

7.3.3.8 Výchozí nastavení

Výchozí nastavení na rozdíl od předchozích knihoven žádným způsobem nezasahuje do barev (a to ani po použití předpřipraveného stylu), jsou tedy značně nepřehledné, pokud je takto vizualizováno více elementů v jednom grafu. Vždy je nutné doplnit elementům barvy manuálně, či je přiřadit dle existující palety.



Obr. 16 Výchozí nastavení knihovny Bokeh může být nepřehledné
Zdroj: vlastní zpracování

7.3.3.9 Shrnutí

Tabulka 7 Hodnocení knihovny Bokeh

Závislosti knihovny	Jinja2, NumPy, packaging, pillow, PyYAML, tornado, typing_extensions
Podporované vstupní formáty	Seznam, n-tice, slovník, NumPy pole, Pandas DataFrame, ColumnDataSource
Podporované výstupní formáty	.html, .png, .svg
Poskytované typy grafů	Spojnicový, plošný, sloupcový, bodový + tvorba vlastních vizualizací
Poskytované možnosti přizpůsobení	Palety, styly, možnost úpravy libovolného elementu
Poskytované možnosti interaktivity	Možnost vložení widgetů i definice vlastních
Jednoduchost použití	Mnohem složitější než Matplotlib, či Seaborn
Přehlednost a atraktivita výchozích nastavení	Vždy nutné definovat alespoň barvy pro přehlednost grafu

Tabulka 8 Kvantifikované hodnocení knihovny Bokeh

Kategorie	Bodové hodnocení	Procentuální hodnocení
Vstupní formáty	5 / 5 bodů	100 %
Výstupní formáty	2 / 3 bodů	66.7 %
Typy grafů	2 / 2 bodů	100 %
Možnosti přizpůsobení	2 / 2 bodů	100 %
Možnosti interaktivity	2 / 2 bodů	100 %
Celkové hodnocení	-	**93 %**

7.3.4 Plotly

7.3.4.1 Závislosti

Závislosti knihovny Plotly se liší podle způsobu používání. Pokud je v programu použita knihovna Plotly, jsou potřeba pouze dvě závislosti:

- six – knihovna pro kompatibilitu mezi Python 2 a 3
- tenacity – knihovna zajišťující opětovné spouštění v případě nespolehlivých operací

Při využití high-level nadstavby Plotly Express jsou však nutné další závislosti:

- NumPy – knihovna pro práci s daty ve formátu vícerozměrných polí
- SciPy – rozšíření knihovny NumPy
- statsmodels – dodatek ke SciPy pro výpočty v oblasti statistiky
- pandas – knihovna pro manipulaci s daty ve formě tabulek
- patsy – knihovna pro popis statistických modelů

Dále je pak pro export do obrazových formátů jako jsou png a jpeg požadována knihovna Kaleido.

7.3.4.2 Vstupní formáty

Podporované vstupní formáty také závisí na použité knihovně. V případě Plotly Express lze data dodat v prakticky libovolném formátu. Pro datové struktury jako jsou slovníky, nebo DataFrame knihovny pandas lze specifikovat osy pomocí názvů sloupců. Další aspekty dat lze pak vizualizovat pomocí barvy, velikosti, nebo šrafování.

Při využití Plotly graph_objects je nutné data předávat ve formátu seznamů, NumPy polí, nebo pandas Series.

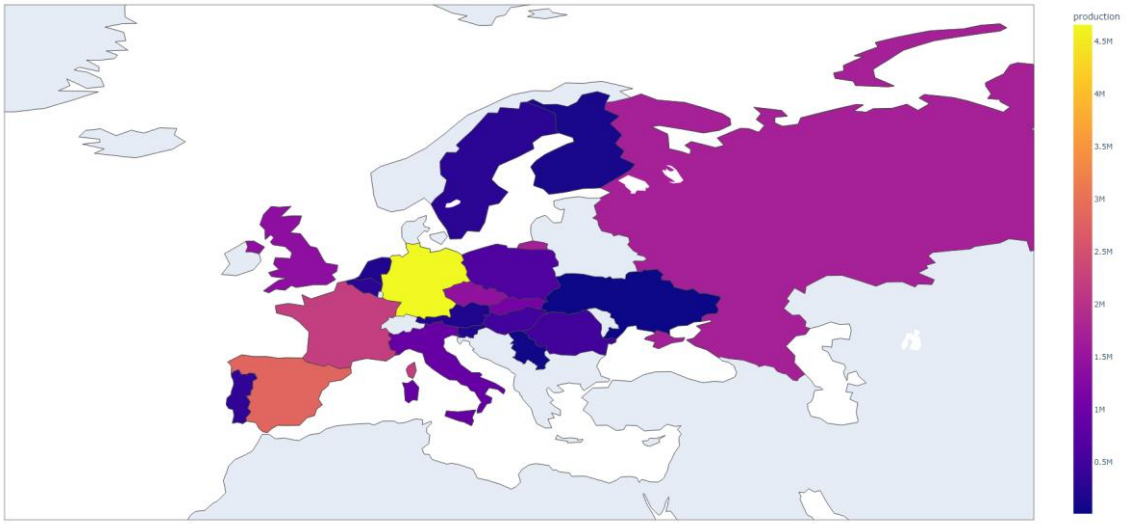
7.3.4.3 Výstupní formáty

Výstup knihovny Plotly je zobrazován ve webovém prohlížeči. Pro export do souboru je k dispozici metoda `fig.write_image()`, pomocí které lze grafy uložit

ve formátech .png, .jpeg, .webp, .pdf a .svg. V případě některých formátů lze také namísto uložení na disk získat v podobě bytů, které lze následně využít v jiných částech programu.

7.3.4.4 Typy grafů

Knihovna Plotly nabízí poměrně široký výběr možných vizualizací, od základních grafů až po specializované vědecké, ekonomické nebo geografické vizualizace. Pomocí Plotly graph_objects lze pak také vytvářet trojrozměrné grafy a mřížky podgrafů.



Obr. 17 Plotly nabízí i vizualizace na zjednodušené mapě
Zdroj: vlastní zpracování – Plotly

7.3.4.5 Přizpůsobení vizualizací

Možnosti přizpůsobení grafů Plotly Express nejsou zdaleka tak obsáhlé, jako u jiných knihoven. Pro úpravu vzhledu lze využít hlavně zabudovaných stylů a barevných palet. U Plotly graph_objects je však možné specifikovat vlastnosti jednotlivých elementů grafu, primárně pomocí slovníků, obsahujících výčet vlastností, jako barvy, tvaru apod. V případě obou knihovnen lze přiřadit vlastnosti dat k vizuálním aspektům a obohatit tak výsledný graf o dodatečné informace.

7.3.4.6 Interaktivita

Díky zobrazení vizualizací v prohlížeči lze interagovat s daty i bez nutnosti speciálních komponent a widgetů. Plotly dovoluje uživateli vybírat, která data chce zobrazit nebo skrýt, omezovat intervaly os k zobrazení, či prozkoumávat hlouběji strukturovaná data, jako například stromy.
Pro hlubší interakci lze přidat vlastní tlačítka, slidery a dropdown menu, kterými lze nastavit libovolné chování. Mohou sloužit například k výběru barevného schéma, nebo dokonce přepnutí typu grafu.

7.3.4.7 Jednoduchost použití

Tvorba vizualizací knihovnou Plotly Express je opravdu velmi jednoduchá. Za předpokladu, že data jsou ve vhodném formátu, může být většina grafů vytvořena jedním voláním funkce, bez nutnosti dalšího kódu, a to i v případech rozsáhlejších datových souborů. Naopak u Plotly graph_objects je nutné předpřipravit data pro vizualizaci a některé úpravy vyžadují více vlastního kódu a detailnější studium dokumentace. To však dává uživateli mnohem větší kontrolu, nad vzhledem výsledného grafu a možnost přesněji definovat některé aspekty vizualizace.

7.3.4.8 Výchozí nastavení

Výchozí nastavení obou knihoven je plně dostačující pro běžné použití a lze ho dále zlepšit pomocí zabudovaných stylů. Knihovna dokonce rozezná případy, kdy by se grafické elementy jako popisky os, či jiných elementů překrývaly a vhodně je orientuje, což značně zjednodušuje práci s některými typy grafů.

7.3.4.9 Shrnutí

Tabulka 9 Hodnocení knihovny Plotly

Závislosti knihovny	**Plotly**: six, tenacity **Plotly Express**: NumPy, SciPy, patsy, statsmodels, pandas
Podporované vstupní formáty	pole, seznamy, n-tice, slovníky, Dataframe
Podporované výstupní formáty	.png, .jpeg, .svg, .webp, .pdf
Poskytované typy grafů	Mnoho typů 2D grafů specializovaných pro různé oblasti
Poskytované možnosti přizpůsobení	**Plotly Express:** Omezené možnosti přizpůsobení, hlavně styly a palety **Plotly:** Hlubší přizpůsobení jednotlivých objektů (marker, line atd.)
Poskytované možnosti interaktivity	Interaktivní vizualizace v prohlížeči, a rozšíření o vlastní ovládací prvky
Jednoduchost použití	**Plotly Express:** Velmi jednoduchá a rychlá tvorba grafů **Plotly:** Složitější, poskytuje však mnohem více možností, včetně podgrafů
Přehlednost a atraktivita výchozích nastavení	V obou případech poměrně přehledné, lze zlepšit styly

Tabulka 10 Kvantifikované hodnocení knihovny Plotly

Kategorie	Bodové hodnocení	Procentuální hodnocení
Vstupní formáty	5 / 5 bodů	100 %
Výstupní formáty	3 / 3 bodů	100 %
Typy grafů	2 / 2 bodů	100 %
Možnosti přizpůsobení	2 / 2 bodů	100 %
Možnosti interaktivity	2 / 2 bodů	100 %
Celkové hodnocení	-	**100 %**

7.3.5 Holoviews

7.3.5.1 Závislosti

Knihovna Holoviews má ve své „minimální instalaci“ velmi málo závislostí:

- NumPy – knihovna pro práci s daty ve formátu vícerozměrných polí
- pandas – knihovna pro manipulaci s daty ve formě tabulek
- Param – knihovna pro kontrolu parametrů volaných funkcí

Standardní instalace pomocí manažeru pip, dále obsahuje:

- panel – balíček pro tvorbu tzv. „dashboards“
- pyviz-comms – balíček pro komunikaci mezi jazyky Python a JavaScript
- colorcet – knihovna barevných palet pro vizualizace
- Markdown – nástroj pro konverzi textu na HTML

Dále je stažena knihovna Bokeh jako základní „backend“ pro vizualizaci a její závlislosti.

7.3.5.2 Vstupní formáty

Formát dat pro vizualizaci se pro Holoviews liší na základě použitého grafu. Například pro sloupcový graf nelze data dodat ve formátu seznamu, nebo uspořádané n-tice tak, jak tomu bylo u jiných vizualizačních knihoven. Holoviews zde předpokládá seznam n-tic ve formátu klíč + hodnota. Nejlepším vstupním formátem pro tuto knihovnu jsou pojmenovaná data, tedy slovníky a DataFrames knihovny pandas.

7.3.5.3 Výstupní formáty

K dispozici jsou výstupní formáty jako .png, .svg, .pdf, .html pro interaktivní vizualizace a .gif nebo .mp4 pro animace. Nabídka výstupních formátů a závislosti, které jsou nutné pro export závisí na knihovně vybrané pro renderování.

7.3.5.4 Typy grafů

Holoviews nabízí velmi širokou nabídku grafů z knihoven Bokeh, Matplotlib a Plotly, některé z nich jsou však k dispozici jen při použití specifické „backend“ knihovny. Pro snazší vyhledávání dostupných typů je k dispozici „referenční galerie“ grafů na stránkách holoviews.org.

7.3.5.5 Přizpůsobení vizualizací

Přizpůsobení vytvořených grafů se provádí pomocí volání `.opts()` na objektu grafu, kde jsou dosazeny parametry, které uživatel chce v daném grafu změnit. Problémem však je závislost na knihovně použité pro renderování výsledné vizualizace. Parametr `cmap` pro vybrání barevné palety lze vybrat při renderování pomocí Bokeh, ale u jiných knihoven skončí takové volání chybou. Parametry, které lze u grafu měnit jsou také v některých případech omezeny, například u již zmiňované knihovny Bokeh nelze v `.opts()` nastavit parametr `hatch_pattern`, sloužící ke specifikaci šrafování.

7.3.5.6 Interaktivita

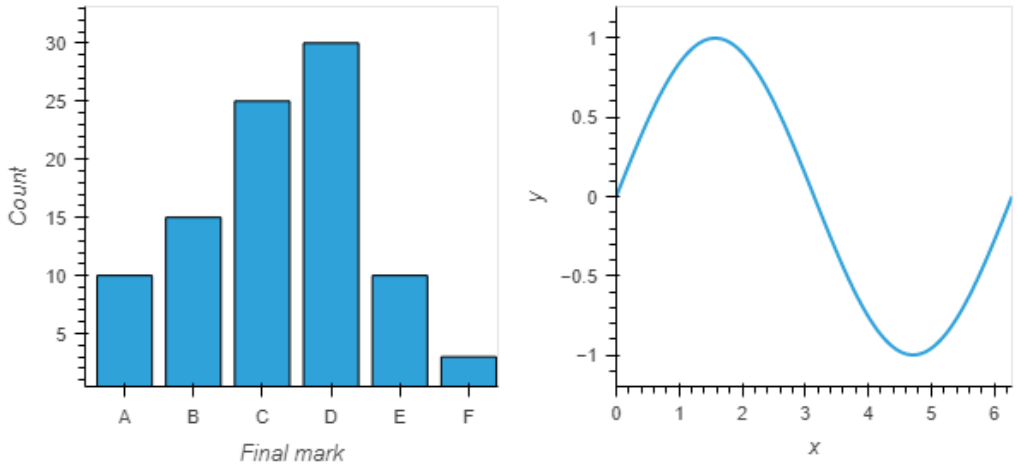
Míra interaktivity dostupná ve vizualizacích je daná použitou knihovnou (viz. příslušné kapitoly)

7.3.5.7 Jednoduchost použití

Holoviews může sloužit jako nástroj pro zjednodušení práce s jinými vizualizačními knihovnami. Problémem je však poměrně nekonsistentní pojmenování parametrů pro metodu `.opts()`, které často neodpovídá názvům z použité knihovny. Výpis dostupných parametrů pro daný typ grafu pomocí `hv.help()` by v tomto případě mohl být velmi užitečný, bohužel však při volání z konzole prostředí Python, nebo ze skriptu končí chybou a funguje správně pouze v prostředí Jupyter Notebook. Dokumentace na webových stránkách knihovny je často nedostačující a v některých případech dokonce nedostupná (například kapitola dokumentace „Plotting with plotly“). Pro knihovny Plotly a Matplotlib je tedy často jednodušší využít přímo těchto knihoven. Pro některé případy užití knihovny Bokeh může být Holoviews užitečným nástrojem, zjednodušující práci s daty i za cenu některých omezení.

7.3.5.8 Výchozí nastavení

Výchozí nastavení odpovídají nastavení použitých knihoven, v mnoha případech je však změněn poměr stran grafu na 1:1.



Obr. 18 Výchozí nastavení Holoviews při využití Bokeh pro renderování
Zdroj: vlastní zpracování – Holoviews

7.3.5.9 Shrnutí

Tabulka 11 Hodnocení knihovny Holoviews

Závislosti knihovny	NumPy, pandas, Param, panel, pyviz-comms, colorcet, Markdown. Dlaší závislosti dle použité knihovny pro renderování
Podporované vstupní formáty	Závislé na grafu, nejčastěji DataFrame a slovníky
Podporované výstupní formáty	Závislé na použité knihovně, .png, .svg, .pdf, .gif, .mp4
Poskytované typy grafů	Závislé na použité knihovně. Široký výběr převážně 2D grafů
Poskytované možnosti přizpůsobení	Závislé na použité knihovně. Často limitované
Poskytované možnosti interaktivity	Závislé na použité knihovně
Jednoduchost použití	V některých případech zjednodušuje práci s daty, mohou však nastat situace, kdy je použití nižší knihovny vhodnější
Přehlednost a atraktivita výchozích nastavení	Závislé na použité knihovně. Často deformované na čtvercový formát

Tabulka 12 Kvantifikované hodnocení knihovny Holoviews

Kategorie	Bodové hodnocení	Procentuální hodnocení
Vstupní formáty	4 / 5 bodů	80 %
Výstupní formáty	3 / 3 bodů	100 %
Typy grafů	2 / 2 bodů	100 %
Možnosti přizpůsobení	1 / 2 bodů	50 %
Možnosti interaktivity	2 / 2 bodů	100 %
Celkové hodnocení	-	**86 %**

7.3.6 Pygal

7.3.6.1 Závislosti

K fungování knihovny Pygal nejsou třeba žádné závislosti. Existují však volitelné závislosti pro export grafů do formátu .png, konkrétně:

- cairosvg – knihovna pro konverzi formátu .svg na .png a .pdf
- tinycss – parser jazyka CSS pro Python
- cssselect – parser pro selektory jazyka CSS

7.3.6.2 Vstupní formáty

Jako vstupní formát dat lze využít většinu jednoduchých datových struktur, jako jsou seznamy, uspořádané n-tice, či pole knihovny NumPy. U složitějších formátů je nejprve nutné je předzpracovat, a redukovat je na zmíněné formáty.

7.3.6.3 Výstupní formáty

Výstup knihovny Pygal lze otevřít v prohlížeči**,** či exportovat ve formátu .svg. Tento formát díky vloženému JavaScript kódu nabízí omezenou míru interakce v podobě skrývání hodnot a zobrazení podrobností o datech. Pro export ve formátu .png je nutná instalace dodatečných knihoven.

7.3.6.4 Typy grafů

Výběr grafů knihovny Pygal je poměrně omezený, celkem lze vytvořit 14 typů dvourozměrných grafů, kromě základních jako například sloupcového, či koláčového, jsou v nabídce Pygal i paprskový graf, stromová mapa, ciferníky a jednoduché geografické vizualizace.

7.3.6.5 Přizpůsobení vizualizací

Možnosti přizpůsobení vizualizací vytvořených pomocí této knihovny jsou poměrně omezené a závisí na typu vytvářeného grafu. Obecně lze přizpůsobovat hlavně barevná schémata, a to pomocí objektu `Style`. Součástí knihovny jsou i předpřipravená schémata, včetně parametrických, která se generují za základě

barevného parametru (například postupným ztmavováním odstínu). Pokud jsou grafy vkládány do webové stránky, lze na ně také aplikovat kaskádové styly. Kromě grafu samotného lze pak upravovat jednotlivé hodnoty, například dodáním popisku či jeho formátováním. Popisky také mohou obsahovat odkazy.

7.3.6.6 Interaktivita

Interaktivita spočívá hlavně ve skrývání dat a interakci s popisky. Knihovna nenabízí žádnou hlubší interaktivitu nebo animace.

7.3.6.7 Jednoduchost použití

Použití knihovny Pygal je velmi jednoduché, uživateli stačí naučit se několik základních principů práce s objektem grafu. K dispozici je také přehledná dokumentace s příklady všech vizualizací.

7.3.6.8 Výchozí nastavení

Výchozí nastavení vizualizací je pro mnoho účelů dostačující, jednotlivým řadám jsou přiřazovány barvy dle schéma stylu, které lze snadno změnit při vytváření objektu grafu.

7.3.6.9 Shrnutí

Tabulka 13 Hodnocení knihovny Pygal

Závislosti knihovny	Žádné + volitelné závislosti pro export a optimalizaci
Podporované vstupní formáty	Pouze jednoduché datové struktury: seznam, n-tice, NumPy pole
Podporované výstupní formáty	Primárně .svg, možný i export do .png
Poskytované typy grafů	Pouze 14 typů dvourozměrných grafů
Poskytované možnosti přizpůsobení	Poměrně omezené, jedná se hlavně o barevná schémata
Poskytované možnosti interaktivity	Minimální – skrývání dat a popisky
Jednoduchost použití	Velmi jednoduchá
Přehlednost a atraktivita výchozích nastavení	Dostačující pro většinu případů, lze rychle upravit pomocí předpřipravených stylů

Tabulka 14 Kvantifikované hodnocení knihovny Pygal

Kategorie	Bodové hodnocení	Procentuální hodnocení
Vstupní formáty	3 / 5 bodů	60 %
Výstupní formáty	2 / 3 bodů	66,7 %
Typy grafů	1 / 2 bodů	50 %
Možnosti přizpůsobení	1 / 2 bodů	50 %
Možnosti interaktivity	1 / 2 bodů	50 %
Celkové hodnocení	-	**55,3 %**

8 Shrnutí výsledků

8.1 Celkové shrnutí knihoven

Při tvorbě libovolné vizualizace je důležité dobře zvolit používanou knihovnu. Ačkoliv u mnoha z hodnocených knihoven se ukázalo, že se jedná o poměrně všestranné nástroje, existují situace, ve kterých jsou některé vhodnější.

Knihovna Matplotlib vyniká svojí všestranností a relativní jednoduchostí. Jedná se o vhodnou knihovnu pro vývojáře, kteří nemají mnoho zkušeností s vizualizací dat. Díky své dlouhé existenci a velké popularitě je také dostupné velké množství materiálů, pro řešení případných problémů. Seaborn dále zjednodušuje některé součásti knihovny Matplotlib a rozšiřuje ji o dodatečné funkcionality, například snadné zpracování DataFrames a tvorba tabulek grafů, dělených dle zvolených kategorií, zároveň je s knihovnou Matplotlib kompatibilní a lze tak v případě potřeby kombinovat vizualizace obou knihoven. Na druhé straně mnohem složitější Bokeh dovoluje vývojáři definovat velmi komplexní vizualizace, díky možnosti pracovat na úrovni jednotlivých grafických elementů. Zároveň umožňuje vytvoření serverové aplikace, vhodné pro streamování a vizualizaci dat v reálném čase a složitější interakce s uživatelem. Holoviews se snaží usnadnit použití knihovny Bokeh při tvorbě jednodušších vizualizací a zároveň poskytuje možnost tyto vizualizace převést do knihoven Matplotlib a Plotly. V mnoha případech je však vhodnější použít tyto knihovny přímo, bez této nadstavby.

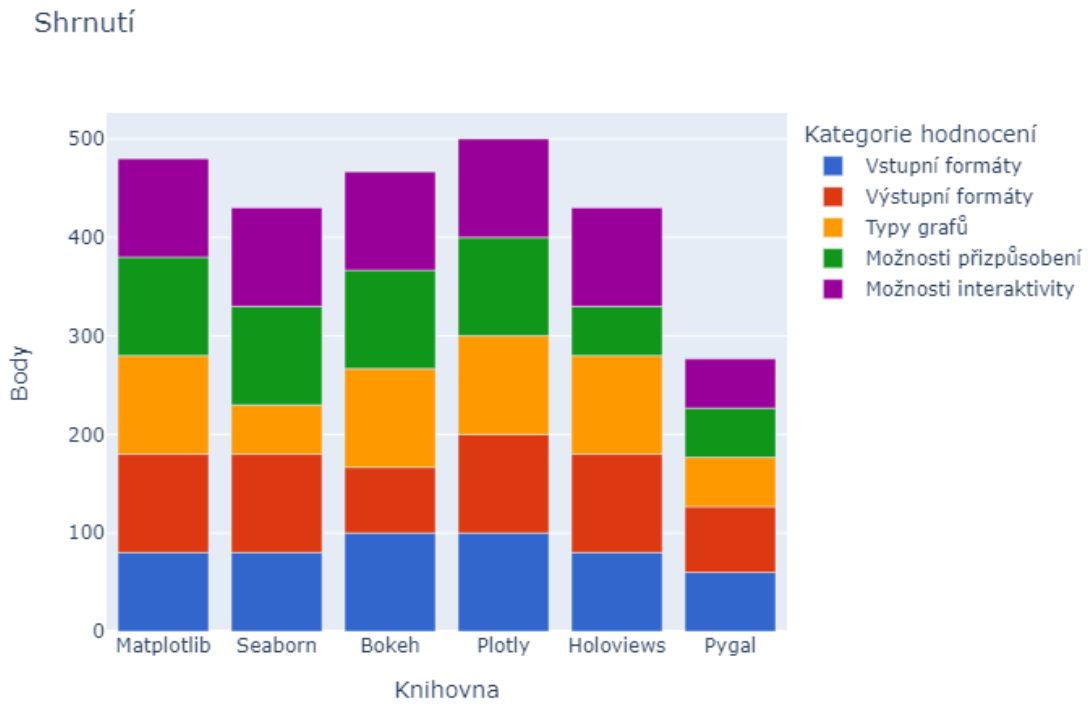
Již zmíněná knihovna Plotly se díky svému dělení na Plotly Express a Plotly Graph Objects pokrývá jak možnosti rychlé tvorby jednoduchých grafů, tak detailnější práci s jednotlivými součástmi grafu. Zároveň nabízí široké možnosti interaktivity, snadnou tvorbu trojrozměrných grafů a je dostupná i v několika dalších jazycích, jako je například JavaScript nebo R.

Poslední hodnocenou knihovnou byla Pygal, která je ve srovnání se ostatními nenabízí zdaleka tolik možností, vyniká však svojí rychlostí a jednoduchostí. Jedná se o ideální nástroj pro velmi rychlou tvorbu jednoduchých vizualizací, které mohou být následně vloženy do programů, či webových stránek, čemuž napomáhá i absence vnějších závislostí.

8.2 Shrnutí kvantifikovaného hodnocení

Tabulka 15 Celkové výsledky hodnocení knihoven

Knihovna	Celkové hodnocení
Matplotlib	96 %
Seaborn	86 %
Bokeh	93 %
Plotly	100 %
Holoviews	86 %
Pygal	55,3 %



Obr. 19 Celkový výsledek byl také vizualizován pomocí knihovny Plotly
Zdroj: vlastní zpracování – Plotly

9 Závěry a doporučení

V rámci bakalářské práce byly představeny základní teoretické principy vizualizace dat a jejich použití v jazyce Python. Vedle některých z nejpoužívanějších knihoven pro vizualizaci dat jako je Matplotlib, Seaborn, Bokeh, nebo Plotly bylo popsáno také několik méně běžných specializovaných knihoven, jako WordCloud pro vytváření vizualizací shrnujících obsah textu, GeoPlotLib zaměřený primárně na geografické vizualizace nebo MidiTime pro převod časových řad na zvuk.

V praktické části bylo navrženo řešení způsobu získávání dat k vizualizaci, s předpokladem, že vytvořený popis bude použit i pro účely výuky. Ze stejného důvodu byly vytvořeny kompletní ukázky vizualizačního řetězce, ve kterých lze najít různorodé způsoby získání a zpracování dat a jejich následnou vizualizaci. Úlohy jsou okomentovány a rozděleny do sekcí „získání“, „zpracování“ a „vizualizace“ pro snazší čitelnost kódu. V některých případech je stejná úloha vypracována několika způsoby, nebo za použití několika různých knihoven.

Užší výběr obecných vizualizačních knihoven byl pak detailněji prozkoumán a zhodnocen. Hodnoceny byly podporované vstupní a výstupní formáty, počet nabízených typů vizualizací a míra jejich přizpůsobitelnosti a interaktivity. Některé aspekty tohoto hodnocení byly převedeny na číselné hodnoty a vzájemně porovnány na základě navržené metriky.

Nejlépe byla díky své všestrannosti hodnocena vizualizační knihovna Plotly, která dosáhla maximálního možného počtu bodů. Tento výsledek ovšem neznamená, že by se jednalo o nejlepší volbu pro všechny případy užití. Naopak nejnižší hodnocení získala knihovna Pygal. K velké bodové ztrátě vedla především její relativní jednoduchost, která však může být v některých případech považována za její nejsilnější stránku. Je také nutné poznamenat, že navzdory všem snahám o objektivnost a nezaujatost, bude tento typ hodnocení vždy ovlivněn zcela subjektivními názory autora.

Nabízí se několik možných rozšíření práce, i námětů na práce navazující. Jedním z takových rozšíření by byla demonstrace a porovnání většího množství vizualizačních knihoven, či zhodnocení knihoven specializovaných na určitý obor a případné srovnání jejich výhod oproti knihovnám obecným. Dále by například mohl být vytvořen program pro interaktivní vizualizaci dat ze školního systému STAG, který by napomáhal studentům s výběrem předmětů a organizací jejich studia (například na základě jejich návaznosti, počtu kreditů, časové náročnosti předmětu apod.). Zajímavou možností aplikace teoretických znalostí vizualizace by pak byla tvorba vlastní vizualizační knihovny, například s využitím OpenGL a shaderového jazyka GLSL.

10 Seznam použité literatury

[1] PURCHASE, Helen et al. Theoretical Foundations of Information Visualization. In: *Lecture Notes In Computer Science*. 4950. 1970, s. 46–64. ISBN 978-3-540-70955-8. DOI: 10.1007/978-3-540-70956-5_3

[2] G. CHIAPPINI a R.M. BOTTINO. *Visualisation in Teaching-Learning Mathematics: The Role of the Computer* [online]. Dostupné z: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.39.3360&rep=rep1&type=pdf

[3] LORÈNE FAUVELLE. *Data visualization: definition, examples, tools, advice [guide 2021]* [online]. 2020 [cit. 03.04.2021]. Dostupné z: https://www.intotheminds.com/blog/en/data-visualization/

[4] INTOTHEMINDS. *What is a data artist? | with Nicholas Rougeux* [online]. 2020 [cit. 16.09.2021]. Dostupné z: https://www.youtube.com/watch?v=k4D9qgVb17Q

[5] JOHN HUNTER a MICHAEL DROETTBOOM. The Architecture of Open Source Applications (Volume 2): matplotlib. [cit. 16.09.2021]. Dostupné z: http://aosabook.org/en/matplotlib.html

[6] JOHN HUNTER a DARREN DALE. Overview — Matplotlib 3.4.3 documentation. 2021 [cit. 08.09.2021]. Dostupné z: https://matplotlib.org/stable/index.html

[7] MANOVICH, Lev. What is visualisation? *Visual Studies*. Routledge, 2011, roč. 26, č. 1, s. 36–49. ISSN 1472-586X. DOI: 10.1080/1472586X.2011.548488

[8] WASKOM, Michael. Seaborn: statistical data visualization. *Journal of Open Source Software*. 2021, roč. 6, č. 60, s. 3021. ISSN 2475-9066. DOI: 10.21105/joss.03021

[9] VANDERPLAS, Jake. *Python Data Science Handbook: Essential Tools for Working with Data*. 1st edition. vyd. Sebastopol, CA: O'Reilly Media, 2016. ISBN 978-1-4919-1205-8.

[10] Sponsored Projects | pandas, NumPy, Matplotlib, Jupyter, + more. In: *NumFOCUS* [online] [cit. 19.09.2021]. Dostupné z: https://numfocus.org/sponsored-projects

[11] BOKEH CONTRIBUTORS. Bokeh documentation. 2021 [cit. 08.09.2021]. Dostupné z: https://docs.bokeh.org/en/latest/index.html

[12] Plotly Open Source Graphing Libraries. [cit. 19.09.2021]. Dostupné z: https://plotly.com/api/

[13] Welcome to HoloViews! — HoloViews 1.14.5 documentation. [cit. 17.10.2021]. Dostupné z: https://holoviews.org/getting_started/index.html

[14] FLORIAN MOUNIER. Pygal — pygal 2.0.0 documentation. 2016 [cit. 08.09.2021]. Dostupné z: http://www.pygal.org/en/stable/

[15] COREY, Michael. Turn your data into sound using our new MIDITime library. In: *Reveal* [online] [cit. 17.10.2021]. Dostupné z: http://revealnews.org/blog/turn-your-data-into-sound-using-our-new-miditime-library/

[16] CUTTONE, Andrea. Geoplotlib – documentation. 5. 9. 2021 [cit. 08.09.2021]. Dostupné z: https://github.com/andrea-cuttone/geoplotlib

[17] MUELLER, Andreas. *word_cloud* [online]. 2021 [cit. 24.10.2021]. Dostupné z: https://github.com/amueller/word_cloud

[18] *The Encyclopedia of Human-Computer Interaction, 2nd Ed.* [online] [cit. 30.10.2021]. Dostupné z: https://www.interaction-design.org/literature/book/the-encyclopedia-of-human-computer-interaction-2nd-ed

[19] ASEEM KASHYAP. 8 Rules for optimal use of color in data visualization. *Medium* [online]. 28. 12. 2020 [cit. 09.09.2021]. Dostupné z: https://towardsdatascience.com/8-rules-for-optimal-use-of-color-in-data-visualization-b283ae1fc1e2

[20] MORELAND, Kenneth. *Diverging Color Maps for Scientific Visualization*. Berlin, Heidelberg: Springer, 2009. Lecture Notes in Computer Science. ISBN 978-3-642-10520-3. DOI: 10.1007/978-3-642-10520-3_9

[21] KOVESI, Peter. Good Colour Maps: How to Design Them. *arXiv:1509.03700 [cs]* [online]. 2015 [cit. 30.10.2021]. Dostupné z: http://arxiv.org/abs/1509.03700

[22] BARTRAM, Lyn, Abhisekh PATRA a Maureen STONE. Affective Color in Visualization. In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* [online]. New York, NY, USA: Association for Computing Machinery, 2017, s. 1364–1374 [cit. 09.09.2021]. ISBN 978-1-4503-4655-9. Dostupné z: https://doi.org/10.1145/3025453.3026041

[23] YI, Mike. How to Choose the Right Data Visualization [online]. 27.2.2020 [cit. 31.09.2021].
Dostupné z: https://cdn2.hubspot.net/hubfs/392937/How-To-Choose-The-Right-Data-Visualization%20(1).pdf

[24] SEVERINO RIBECCA. The Data Visualisation Catalogue. [cit. 05.04.2021].
Dostupné z: https://datavizcatalogue.com/

[25] CONOR HEALY a YAN HOLTZ. From data to Viz | Find the graphic you need. [cit. 10.04.2021]. Dostupné z: https://www.data-to-viz.com/

[26] MORELAND, Kenneth. Why We Use Bad Color Maps and What You Can Do About It. *Electronic Imaging*. 2016, roč. 2016, č. 16, s. 1–6. ISSN 2470-1173. DOI: 10.2352/ISSN.2470-1173.2016.16.HVEI-133

[27] index | TIOBE – The Software Quality Company. [cit. 14.11.2021].
Dostupné z: https://www.tiobe.com/tiobe-index/

[28] PYPL PopularitY of Programming Language index. [cit. 14.11.2021].
Dostupné z: https://pypl.github.io/PYPL.html

[29] LUTZ, Mark. *Learning Python: Powerful Object-Oriented Programming*. O'Reilly Media, Inc., 2013. ISBN 978-1-4493-5569-2.

[30] Applications for Python. In: *Python.org* [online] [cit. 14.11.2021].
Dostupné z: https://www.python.org/about/apps/

[31] THE NUMPY COMMUNITY. NumPy v1.21 Manual. [cit. 08.09.2021].
Dostupné z: https://numpy.org/doc/stable/

[32] THE SCIPY COMMUNITY. SciPy documentation — SciPy v1.8.0 Manual. [cit. 20.11.2021]. Dostupné z: https://scipy.github.io/devdocs/index.html

[33] THE PANDAS DEVELOPMENT TEAM. Getting started — pandas 1.3.4 documentation. [cit. 20.11.2021].
Dostupné z: https://pandas.pydata.org/docs/getting_started/index.html

[34] NASA PUBLIC DATA. Meteorite Landings | NASA Open Data Portal. [cit. 20.02.2022].
Dostupné z: https://data.nasa.gov/Space-Science/Meteorite-Landings/gh4g-9sfh

11 Přílohy

1) Zdrojové kódy projektu

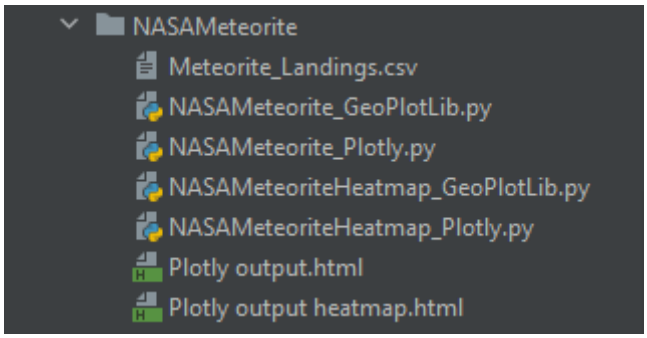
Soubory projektu jsou dostupné v podobě Git repositáře na adrese:
https://github.com/Scriptman777/PythonVisualisaton

Struktura projektu

- !Thesis – složka obsahuje elektronickou podobu této práce ve formátech .docx a .pdf
- Complete examples – složka obsahuje zdrojové kódy k úlohám kapitoly 7.2
    - Každá podsložka obsahuje jednu úlohu (ukázkové řešení, data k vizualizaci a zdrojové kódy, které jsou značeny podle použité knihovny a pro lepší čitelnost okomentovány)
- Library testing – složka obsahuje zdrojové kódy využité při testování knihoven v kapitole 7.3
    - Složka Template obsahuje prázdnou šablonu pro testy knihoven
    - Složka Summary obsahuje skript a data pro vytvoření souhrnné vizualizace z kapitoly 8.2
    - Ostatní složky obsahují samotné testy knihoven
- Loading data – složka obsahuje zdrojové kódy demonstrující načítání dat z různých zdrojů popisované v kapitole 7.1
    - Podsložky jsou děleny dle kapitol, tedy načítání dat ze souboru, API, databáze a webových stránek. Složky obsahují data k načtení a zdrojový kód, jak v klasické podobě, tak v podobě Jupyter notebook. Soubory Jupyter notebook jsou dále vyexportovány do formátu .pdf pro snazší čitelnost a možnost prohlížení výstupů bez nutnosti spouštět programy samotné.

Ukázka zpracované úlohy

Popis ukazuje strukturu úlohy z kapitoly 7.2.4. zabývající se vizualizací geografických dat o dopadu meteoritů poskytovaných agenturou NASA.



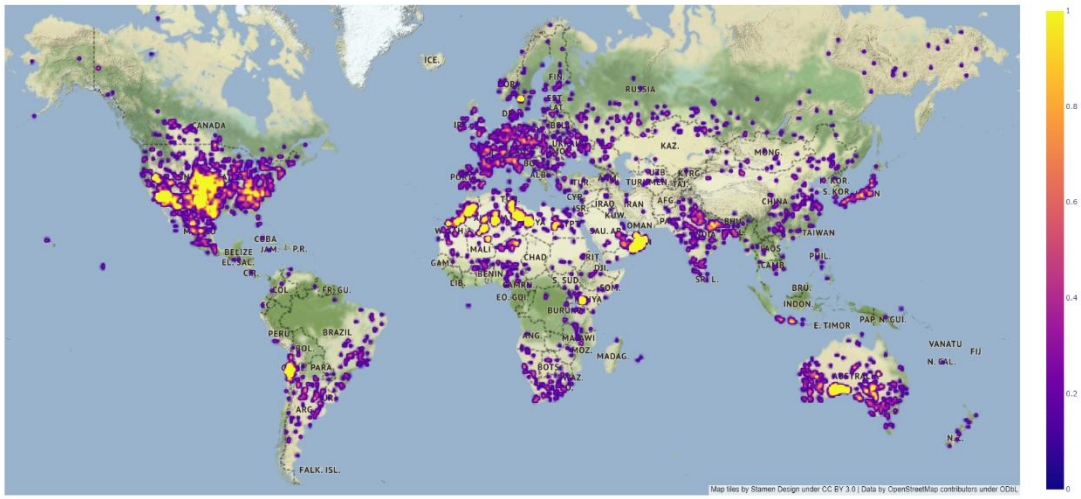
Obr. 20 Ukázka obsahu složky s úlohou
Zdroj: vlastní zpracování

Složka úlohy obsahuje zdrojová data zdrojové kódy ukázkových řešení (v tomto případě dělených dle typu vizualizace a použité knihovny). U knihoven, které nejsou schopny vizualizaci zobrazit přímo, jsou do složky ukládány výsledné soubory.

Následuje ukázka zdrojového kódu:

```
import csv
import pandas as pd
import plotly.express as px

lons = []
lats = []
names = []

# PROCESS DATA

with open("Meteorite_Landings.csv", encoding='utf-8') as landings:
    # Read CSV file
    reader = csv.DictReader(landings)
    for data in reader:
        # Get location and name
        if not data['GeoLocation'] == "(0.0, 0.0)" and
data['GeoLocation'].strip():
            lons.append(float(data['reclong']))
            lats.append(float(data['reclat']))
            names.append(data['name'])

# Create pandas DataFrame of locations
static_dict = {'name':names, 'lon':lons, 'lat':lats,}
df = pd.DataFrame(static_dict)

# VISUALIZE DATA

fig = px.density_mapbox(df, lat='lat', lon='lon', radius=5,
center=dict(lat=0, lon=180), zoom=0, mapbox_style="stamen-terrain")
fig.write_html("Plotly output heatmap.html")
```

Zdrojový kód je členěn do sekcí – získání, zpracování a vizualizace dat. Ne každá úloha obsahuje všechny tyto sekce, například tento skript se nezabývá získáním souboru s daty.



Obr. 21 Ukázka výsledné vizualizace
Zdroj: vlastní zpracování – Plotly

UNIVERZITA HRADEC KRÁLOVÉ
Fakulta informatiky a managementu
Akademický rok: 2020/2021

Studijní program: Aplikovaná informatika
Forma studia: Prezenční
Obor/kombinace: Aplikovaná informatika (ai3-p)

Podklad pro zadání BAKALÁŘSKÉ práce studenta

Jméno a příjmení: **David Továrek**
Osobní číslo: **I1900265**
Adresa: Zborovská 347, Nové Město nad Metují, 54901 Nové Město nad Metují 1, Česká republika

Téma práce: Využití jazyka Python pro vizualizaci dat
Téma práce anglicky: Data visualization using Python

Vedoucí práce: Ing. Bruno Ježek, Ph.D.
Katedra informatiky a kvantitativních metod

Zásady pro vypracování:

Cíl práce:
Cílem práce je prozkoumat možnosti vizualizace v prostředí jazyka Python. Na sadě příkladů odzkoušejte celý řetězec zpracování dat od získání až po vizualizaci.
Postup prací:

1. Prozkoumat principy a metody pro vizualizaci dat.
2. Vytvořit přehled existující vizualizačních knihoven pro jazyk Python, zahrnout možnost využití programovatelných grafických karet.
3. Navrhnout sadu ukázkových úloh demonstrujících řetězec zpracování od získávání dat až po jejich vizualizaci. Zaměřit se na různé typy datových souborů.
4. Navržené úlohy implementovat a otestovat pro dostupné datové soubory.
5. Zhodnotit dosažené výsledky.

Seznam doporučené literatury:

VANDERPLAS, Jake. *Python Data Science Handbook: Essential Tools for Working with Data*. 1st edition. vyd. Sebastopol, CA: O'Reilly Media, 2016. ISBN 978-1-4919-1205-8.
YAU, Nathan. *Visualize This: The FlowingData Guide to Design, Visualization, and Statistics*. 1st edition. vyd. Indianapolis, Ind: Wiley, 2011. ISBN 978-0-470-94488-2.
MANOVICH, Lev. What is visualisation? *Visual Studies*. Routledge, 2011, roč. 26, č. 1, s. 36–49. ISSN 1472-586X. DOI: 10.1080/1472586X.2011.548488

Podpis studenta: Datum:

Podpis vedoucího práce: Datum: