VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

VYUŽITÍ ZÍSKÁVÁNÍ ZNALOSTÍ PRO DATA Z PDF SOUBORŮ

USE OF KNOWLEDGE DISCOVERY FOR DATA FROM PDF FILES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **LIBOR DVOŘÁČEK**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. VLADIMÍR BARTÍK, Ph.D.**
SUPERVISOR

BRNO 2021

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav informačních systémů (UIFS) Akademický rok 2020/2021

Zadání bakalářské práce

Student: **Dvořáček Libor**
Program: Informační technologie
Název: **Využití získávání znalostí pro data z PDF souborů**
Use of Knowledge Discovery for Data from PDF Files
Kategorie: Data mining

Zadání:

1. Seznamte se s problematikou získávání znalostí z dat.
2. Seznamte se s programovacím jazykem Python, prostudujte dostupné knihovny pro zpracování PDF souborů a získávání znalostí (zejména shluková analýza a metoda PCA).
3. Po konzultaci s vedoucím navrhněte demonstrační aplikaci provádějící vybrané metody získávání znalostí s daty extrahovanými z PDF souborů.
4. Navrženou aplikaci implementujte a proveďte testování na vhodném vzorku dat.
5. Zhodnoťte dosažené výsledky a další možnosti pokračování tohoto projektu.

Literatura:

- Han, J., Kamber, M.: Data Mining - Concepts and Techniques, 2nd Edition. Morgan Kaufmann Publishers, 2006.
- Layton, R.: Learning Data Mining with Python. Packt Publishing, 2015.

Pro udělení zápočtu za první semestr je požadováno:

- Body 1-3.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Bartík Vladimír, Ing., Ph.D.**
Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2020
Datum odevzdání: 12. května 2021
Datum schválení: 22. října 2020

Abstrakt

Bakalářská práce se zabývá extrakcí tabulek z digitálně vytvořených pdf a následným použitím získatých dat pro datovou analýzu. Použity jsou metody redukce dimenzí a shlukové analýzy. Hlavním obsahem je rozbor dostupných nástrojů pro extrakci dat v jazyce python, popis a porovnání použitých metod strojového učení a implementace aplikace, která všechna tato témata sdružuje do jednoho funkčního celku na adrese: http://extraktor.herokuapp.com.

Abstract

This bachelor thesis deals with the extraction of tables from digitally created pdfs and the subsequent use of the obtained data for data analysis. Methods of dimension reduction and cluster analysis are used. The main content is an analysis of available tools for data extraction in the python language, a description and comparison of the used machine learning methods and implementation of an application that combines all these topics into one functional unit at: http://extraktor.herokuapp.com.

Klíčová slova

data mining, získávání znalostí, Python, PDF, PCA, Dendrogram, T-SNE, K-MEANS, UMAP, redukce dimenzí, vizualizace vícerozměrných dat, shluková analýza, Dash, Plotly, Heroku

Keywords

data mining, knowledge discovery, Python, PDF, PCA, Dendrogram, T-SNE, K-MEANS, UMAP dimensionality reduction, visualization of high-dimensional datasets, cluster analysis, Dash, Plotly, Heroku

Citace

DVOŘÁČEK, Libor. *Využití získávání znalostí pro data z PDF souborů*. Brno, 2021. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Vladimír Bartík, Ph.D.

Využití získávání znalostí pro data z PDF souborů

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Vladimíra Bartíka, Ph.D. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Libor Dvořáček
10. května 2021

Poděkování

Děkuji panu Ing. Vladimíru Bartíkovi, Ph.D za velkou ochotu při přidělování tématu této bakalářské práce a za vstřícnost, rady a čas v celém průběhu jejího psaní. Dále děkuji svým rodičům za morální podporu v průběhu studia. Poslední poděkování patří Markétce, protože... je úžasná.

Obsah

Kapitola 1

Úvod

S rostoucím počtěm počítačově generovaných informací vzniká potřeba tyto informace opět extrahovat a dále analyzovat. Datamining tak v dnešní době nabývá na stále větším významu. Obrovské množství informací se každý den ukládá ve formátu PDF a v současné době neexistuje formát, který by ho mohl nahradit. V porovnání se soubory typu DOCx, XLSx a PPTx tvoří na internetu přes 80 procent soubory právě tohoto typu, jak vyplývá z dokumentu PDF asociace. [11]

Tato práce se konkrétně zabývá extrakcí tabulek z PDF souborů generovaných plynovými chromatografy a jejich zobrazením ve dvoudimenzionálním prostoru. Klade si za cíl seznámit čtenáře s obecnou strukturou formátu PDF, nástroji pro extrakci dat z těchto souborů a analytickými metodami používanými k redukci dimenzionality a zobrazení těchto dat. Na závěr je pak popsána implementace webové aplikace, která všechna tato témata sdružuje do jednoho funkčního celku.

Tato práce je členěna celkem do sedmi kapitol. V kapitole číslo 2 se zabývám formátem PDF a je zde popsáno základní rozdělení těchto dokumentů a jejich struktura. Kapitola číslo 3 obsahuje krátký úvod do problematiky data miningu a dále popis analytických metod, které jsem použil v konečné implementaci. Jmenovitě: Principal Component Analysis, t-distrubuted Stochastic Neighbor Embedding, Dendrogram, K-Means a Uniform Manifold Approximation and Projection. Kapitola číslo 4 si klade za cíl vytvořit přehled nejpoužívanějších nástrojů pro extrakci dat z PDF za pomocí jazyka Python a porovnat je mezi sebou. Kapitola číslo 5 pak popisuje samotnou implementaci a nasazení internetové aplikace do provozu. V první části kapitoly číslo 6 se věnuji jejímu testování. Ve druhé části jsou následně použité metody mezi sebou porovnal na datasetu obsahujícím 1797 číslic v rozlišení 8x8 pixelů, který je dobře známý jako The Digit Dataset. To jsem ještě doplnil o vlastní výstupy z koncové aplikace. Poslední kapitolou číslo 7 je závěr.

Kapitola 2

Formát PDF

Tato kapitola pojednává o struktuře formátu PDF. Informace zde publikované jsou čerpány z oficiální dokumentace. [1]

Portable Document Format, zkráceně PDF, je digitální formát vyvinutý v roce 1993 firmou Adobe. V roce 2008 se stal tzv. otevřeným standardem, když vývoj převzala Mezinárodní organizace pro normalizaci. Tehdejší motivací bylo usnadnění zobrazení a výměny elektronických dokumentů bez ohledu na to v jakém prostředí byly vytvořeny. Dnes je z PDF v těchto směrech jeden z nejrozšířenějších formátů, v němž je uloženo obrovské množství informací a proto také vznikla potřeba tato data extrahovat pro další zpracování. Právě grafická integrita tohoto formátu je však bohužel vykoupena náročnějším získáváním informací z těchto souborů. Proto je při extrakci dat nutnost použít speciálně pro tento typ souboru vytvořený program. Pozdější úprava tohoto souboru však může zapříčinit nefunkčnost extrakce a nutnost modifikace programu, která bude tyto úpravy reflektovat.

2.1 Členění souboru

PDF soubor je rozdělen na čtyři základní sekce znázorněné na obrázku 2.1.

Obrázek 2.1: Sekce PDF

Hlavička

Začíná na bytu 0 a lze ji zobrazit například příkazem XXD, a to následovně:

```
$ xxd -l 16 doc.pdf
00000000: 2550 4446 2d31 2e33 0a25 c4e5 f2e5 eba7  %PDF-1.3.%......
```

Výstup se skládá z hexadecimální adresy, obsahu v haxedacimální soustavě a obsahu zakodovaném v ASCII. Některé aplikace mohou číst data ze začátku souboru pro zjištění

zda obsah souboru číst jako text nebo binární data. V případě přítomnosti binárních dat tak následuje řádek s dalším komentářem (uvozený znakem %), který obsahuje minimálně čtyři znaky s ASCII hodnotou větší než 128. Timto je zajištěno, že soubor bude vždy považován za binární.

Tělo

Obsahuje seznam objektů obsažených v pdf, například objekty reprezentující nějaký prvek na stránce, vlastní stránku, font, obrázek, případně tzv. "object stream", což je posloupnost několika za sebou uložených objektů. Objekty mohou být typu Boolean, Integer, Real, String, Name, Array, Dictionary, Stream, případně null a grafické objekty.

Tabulka odkazů

Obsahuje konkrétní odkaz na místo počátku každého objektu, který soubor obsahuje začíná klíčovým slovem **xref** následovaným jednou nebo několika podsekcemi. Každá podsekce je uvozena identifikátorem prvního objektu a počtem všech objektů v podsekci. Tim je umožněno náhodné čtení dokumentu. Příklad referenční tabulky následuje.

```
xref
0 2
0000000000 65535 f
0000010635 00000 n
20 8
0000010766 00000 n
0000010977 00000 n
0000011004 00000 n
0000011322 00000 n
0000011642 00000 n
0000011741 00000 n
0000011778 00000 n
0000014491 00000 n
```

V přikladu je referenční tabulka se dvěma podsekcemi. První objekt první podsekce má identifikátor 0 a podsekce obsahuje celkem 2 objekty. Druhá podsekce obsahuje 8 objektů a první z nich má identifikátor 20. První objekt v každé referenční tabulce má vždy identifikátor 0 a je to rezervovaný objekt, který nikam neodkazuje. První sloupec udává offset od začátku souboru, na kterém je objekt umístěn. (Objekt číslo 20 se nachází na offfsetu 10766). Druhý sloupec udává verzi objektu. Třetím sloupcem je FLAG, který udává je-li objekt použítý či nikoli. U objektu 0 je verze standardně nastavena na hodnotu 65535 a je označen jako nepoužitý. Celý záznam jednoho objektu má vždy délku 20 bytů včetně znaků konce řádku. Více podsekcí je výsledkem **inkrementálních aktualizací** (viz. sekce 2.4). Ty spočívají v přidání dalšího těla, tabulky odkazů a traileru na konec původního souboru. S příchodem PDF 1.5 byl představen také xref stream pro podporu souborů s velikostí nad 10GB. Soubor tak může obsahovat oba typy záznamů pro zajištění zpětné kompatibility.

Trailer

Jedná se o slovník. Vždy obsaženými klíči jsou **/Root** a **/Size**. Root obsahuje odkaz na kořenový objekt reprezentující celý dokument. Size udává počet položek v tabulce odkazů.

Volitelné klíče mohou být následující:

- **/Prev** v případě modifikovaného souboru udává na jakém offsetu se nachází předešlá tabulka odkazů
- **/Info** je odkaz na slovník obsahující metadata o souboru
- **/Encrypt** obsahuje informaci, že soubor je zašifrován
- **/ID** obsahuje dvě unikátní ID dokumentu. Ty nejsou zašifrované, a proto je možné je přečíst a zjistit jestli jde o správný dokument bez toho abychom dokument museli dešifrovat.

```
Trailer
    <<
        /ID[<EBE7B292709917E5AA6C9C54424C1A79>
        <0F9DDA910379A6A02A9E1F992A9CAB2D> ]
        /Root 14 0 R
        /Size 28
        /Prev 10056
        /Info 20 0 R
    >>
    startxref
    14544
    %%EOF
```

Kořenový objekt je objekt číslo 14. **Startxref 14544** říká, že referenční tabulka začíná na offsetu 14544. **%%EOF** značí konec souboru

2.2 Objekty

Tato sekce obsahuje popis typů objektů, které se mohou v PDF souboru vyskytovat.

Integer
Decimální celé číslo, kterému může předcházet znaménko.

Real
Reprezentuje celé číslo se znaménkem na začátku a desetinou tečkou na začátku, na konci nebo uprostřed.

String
Může reprezentovat 0 nebo více za sebou následovaných znaků. V případě textových řetězců jsou pro oddělení použity kulaté závorky. Špičaté závorky pak v případě zápisu řetězce v hexadecimální formě.

```
(FIT je nejlepsi!)
<464954206a65206e656a6c6570161ed21>
```

Jméno

Objekt definovaný normou jako atomický symbol, neboli sekvence unikátních znaků kromě znaku null. Unikátní znamená, že objekty se stejnou sekvencí znaků odkazují na tentýž objekt. Atomicita pak značí absenci vnitřní struktury. Jméno je uvozeno lomítkem (/). Může obsahovat jakékoli znaky, nicméně bílé znaky a znaky s ASCII hodnotou mimo interval <33, 126> je nutné zapsat v dvouciferné hexadecimální notaci. Příkladem budiž jméno "FIT VUT",které nutno zapsat jako /FIT#20VUT Maximální délka jména není normou definována, nicméně může být limitována konkrétním programem pracujícím s daným souborem, případně jeho prostředím.

Pole

Tyto objekty se zapisují do hranatých závorek. Mohou být heterogenní, to znamená že mohou obsahovat prvky různých typů.

```
[ FIT/VUT 1 True ]
```

Slovník

Slovník je složený z páru klíč-hodnota. Zapisujeme to dvojitých špičatých závorek a platí zde pravidlo unikátního klíče. Klíčem je jméno, hodnota může být jakýkoli typ objektu, včetně dalšího slovníku. Stejně jako u pole doléhá maximální velikost slovníku implementačním limitům. Příklad pole obsahující metadata:

```
<< /Producer (macOS Version 11.0.1 \(Build 20B29\) Quartz PDFContext,
    AppendMode 1.1)
        /CreationDate (D:20201226163129Z00'00')
        /ModDate (D:20201228084833Z00'00')
        /Title (test)
        /Creator (Pages)
>>
```

Klíče jsou uloženy jako objekty typu jméno. Hodnoty pak jako řetězce.

Stream

Neomezeně dlouhá sekvence bytů. Skládá se z identifikátoru, slovníku a vlastních dat oddělených na začátku klíčovým slovem stream a na konci klíčovým slovem endstream. Příklad:

```
18 0 obj
        << /Length 222 /Filter /FlateDecode >>
        stream
            x]n {bKq_NrX[HsRl|0O=03qc0Ht4$d}4EZw8=8C^#y2l)}A cT{.?\JBt?
                \&-(RFnF -tFQJu*Zx6}U)o|&pA
        endstream
endobj
```

V tomto konkrétním případě obsahuje slovník stream objektu informaci o délce streamu a typu filtru. Filter udává jakým způsobem má být stream čtečkou dekódován.

Textové objekty

Slouží k uložení textu a náleží do množiny grafických objektů. Textový objekt může obsahovat tři typy operátorů:

- **text state** - definují vlastnosti související se vzhledem textu, například mezery mezi slovy nebo jednotlivými znaky, použitý font a jeho velikost.
- **text positioning** - specifikují pozici textu na stránce
- **text showing** - zajišťují vlastní zobrazení textu na stránce

Příklad textového objektu:

```
BT
        /F4 42 Tf
        20 40 Td
        0 0 1 rg
        0 Tr
        (FIT) Tj
ET
```

- **BT** značí začátek textového objektu a přechod do stavu pro renderování textu.
- **Tf** specifikuje použité písmo a jeho velikost.
- **Td** pak udává od jakých souřadnic se začne vykreslovat první znak. Standardně nastavenou černou barvu výstupního textu lze změnit například operátorem rg jemuž předchází specifikace mohutnosti tří základních barev RGB modelu v rozsahu <0,1>.
- **Tr** nastavuje způsob renderování znaků. Tím lze definovat výplň, obtažení nebo ořez textu.
- **Tj** vezme svůj vstupní operand v podobě řetězce a vykreslí ho na základě výše definovaných parametrů.
- **ET** označuje konec textového objektu.

Fonty

PDF definuje 5 typů fontů. Mezi nejběžnější patří fonty Typu 1. Jde o 14 vestavěných fontů při jejichž použití není potřeba jejich definici ukládat do PDF. Z ostatních typů lze ještě zmínit písmenný formát TrueType a jeho pozdější rozšíření OpenType. Původně vyvinuty firmou Apple jako alternativa k fontům Typu 1 firmy Adobe. Hlavní odlišností těchto dvou formátů je způsob popisu jednotlivých znaků. U standardu PostScript Type 1 jde o kubické beziérovy křivky oproti kvadratickým B-splinům použitých v případě TrueType. Z toho vyplývá, že méně kontrolních bodů v případě TrueType zajišťuje sice rychlejší vykreslování, nicméně z části redukuje volnost v popisu jednotlivých křivek. Příklad objektu se slovníkem obsahující definici TrueType fontu:

```
6 0 obj
  << /Type /Font
    /Subtype /TrueType
    /BaseFont /AAAAAB+HelveticaNeue
    /FontDescriptor 15 0 R
    /Encoding /MacRomanEncoding
    /FirstChar 32
    /LastChar 146
    /Widths [ 278
```

```
    0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 556 0 0 0 0 0 0 0 0 0 0 0 0
    0 648 0 0 704 0 0 0 0 0 0 0 0 0 0 760 0 0 685 0 574 0 611 0 611
    0 0 0 0 0 0 0 0 537 0 537 593 537 296 0 0 0 0 0 0 0 0 574 593 0
    0 500 315 0 500 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
    0 0 0 222 ]
  >>
endobj
```

- **/Type** specifikuje typ objektu.
- **/Subtype** udává typ písma.
- **/BaseFont** je postscriptové jméno písma.
- **/FontDescriptor** odkazuje na ostatní metriky písma, například tloušťku a kurzívu.
- **/Encoding** označuje kódování znaků neboli vytváří vztah mezi kódy jednotlivých znaků a matematickým popisem znaků.
- **/FirstChar/LastChar** udává ASCII hodnotu znaku prvního/posledního prvku v poli Widths.
- **/Widths** obsahuje pole šířek jednotlivých znaků použitých v dokumentu.

Path objekty

Path objekty jako textové objekty náleží do množiny grafických objektů. Mohou obsahovat tři typy operátorů:

- **path construction** - používají se k vlastní definici jednotlivých grafických prvků
- **clipping path** - umožňuje použít hranice výsledného objektu pro ořez následujícího
- **path-painting** - používá se jako poslední příkaz a zajišťuje vlastí vykreslení

Příklad: vykreslení červeného čtverce

```
stream
    1 0 0 rg
    200 300 400 400 re
    f
endstream
```

- **rg** značí použití černé barvy
- **re** přídá čtverec k současné cestě na souřadnicích 200, 300 s šířkou i výškou 400
- **f (fill)** uzavře (pokud se tak již nestalo) a vyplní cestu barvou

Nepřímé objekty

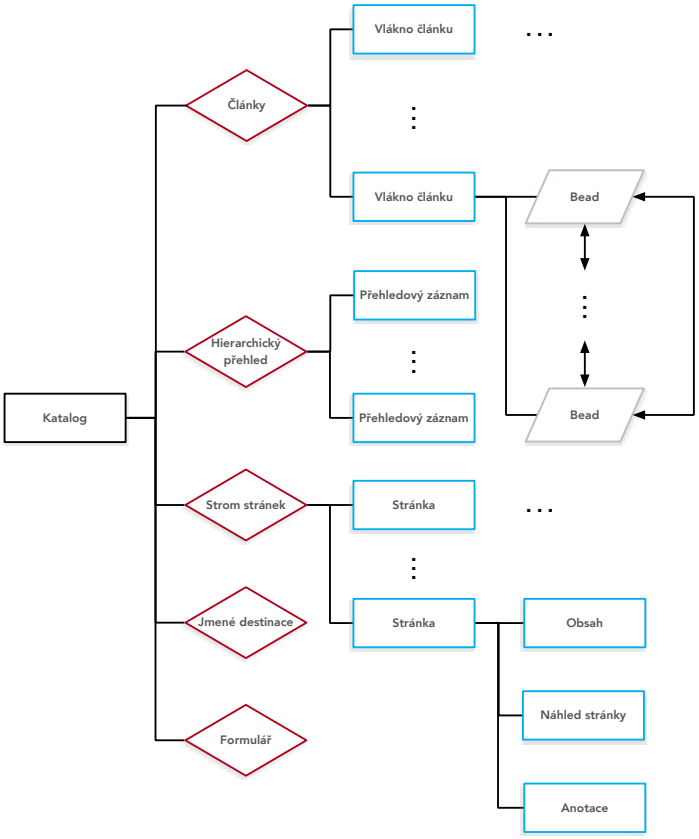
Nepřímé objekty obsahují unikátní identifikátor pomocí kterého se na něj ostatní objekty mohou odkazovat. Ten se skládá z čísla objektu a verze objektu. Při vytvoření objektu je verze standardně nastavena na hodnotu 0. Příklad:

```
5 0 obj
       [ /ICCBased 9 0 R ]
endobj
```

Definuje nepřímý objekt mající číslo 5 a verzi 0, obsahem je pole se jménem **/ICCBased** a odkazem na jiný nepřímý objekt (9 0 R). Odkaz se skládá z identifikátoru odkazovaného objektu a klíčového písmene R.

2.3 Struktura dokumentu

Struktura PDF dokumentu je uspořádána do stromové hierarchie, která je zjednodušeně zachycena na obrázku 2.2.



Obrázek 2.2: Struktura dokumentu

Katalog

Tvoří kořenový uzel stromové struktury dokumentu. Odkazuje na něj položka **Root** v traileru dokumentu. Jedná se o slovník, který obsahuje odkazy na objektu popisující obsah dokumentu.

Příklad katalogu:

```
14 0 obj
<<  /Type /Catalog
       /Pages 2 0 R
       /MarkInfo
       << /Marked true
       >>
       /StructTreeRoot 11 0 R
>>
endobj
```

- **Pages** je odkaz na kořen stromu stránek
- **MarkInfo** udává informace o značkách
- **StruckTreeRoot** je odkaz na kořenový uzel popisující logickou strukturu dokumentu

Strom stránek

Obsahuje dva typy uzlů. Page tree nodes a leafs (page objects). Organizací stránek do stromové struktury je zajištěno rychlejší zobrazení požadované stránky i v mnohasetstránkovém dokumentu.

Příklad page tree node:

```
2 0 obj
<< /Type /Pages
       /MediaBox [0 0 842 595]
       /Count 1
       /Kids [ 1 0 R ]
>>
endobj
```

- **MediaBox** - rozměry stránek
- **Count** - počet stránek
- **Kids** - odkazy na listové uzly reprezentující jednotlivé stránky

Příklad page objektu:

```
1 0 obj
<< /Type /Page
       /Parent 2 0 R
       /Resources 4 0 R
       /Contents 3 0 R
       /MediaBox [0 0 842 595]
>>
endobj
```

- **Parent** - odkaz na rodičovský uzel
- **Resources** - odkaz na pojmenované zdroje
- **Content** - vlastní obsah stránky

Články

Používají se v dokumentech, kdy logicky za sebou následující obsah není uložen v dokumentu sekvenčně. Například pokud text pokračuje na další stránce. Jednotlivé části textu jsou rozděleny na tzv. **Beads**. Články z hlediska datové struktury obsahují pole slovníků. V případě článků i jejich částí se jedná o datovou strukturu slovník. Kořen podstromu **Články** je pak reprezentován jako pole.

Jmenné destinace

Používají se například při odkazu do jiného dokumentu. Použitím jména destinace namísto čísla stránky je zabráněno invalidaci odkazu v případě modifikace dokumentu odkazovaného.

Interaktivní formulář

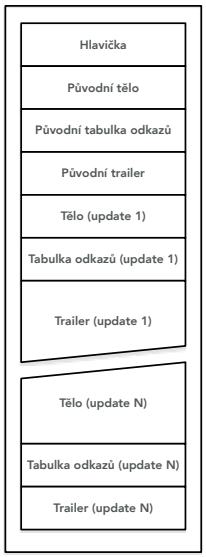
Používá se při potřebě získaní dat od uživatele.

Hierarchický přehled

Umožňuje uživateli lepší navigaci v dokumentu zobrazením interaktivního hierarchického obsahu.

2.4 Inkrementální aktualizace

Přidáním nových prvků základní struktury na konec souboru je možné provádět inkrementální aktualizace. Přidané tělo bude obsahovat objekty nově vložené do dokumentu. Přidaná referenční tabulka pak odkazy pouze na nové, změněné, nahrazené nebo vymazané objekty. Přidaný trailer je pak pouze kopií předešlého traileru s přidaným nebo modifikovaným parametrem **/Prev**, který odkazuje na předešlou referenční tabulku. Každý trailer včetně všech předchozích je zakončen znakem konce souboru.Tímto způsobem modifikace souboru je umožněno PDF snadněji editovat a následně rychleji ukládat i velké dokumenty v řádu desítek gigabytů.



Obrázek 2.3: Sekce PDF s inkrementálními aktualizacemi

Kapitola 3

Data Mining

Tato kapitola se zabývá stručným úvodem do problematiky data miningu a popisem metod shlukové analýzy a redukce dimenzionality, které jsem nastudoval a použil ve výsledné implementaci. Konkrétně reprezentací aglomerativního hierarchického shlukování Dendrogramem 3.2 a metodou částečného shlukování K-Means 3.3. Následně jsou popsány metody redukce dimenzionality v pořadí dle data publikace. Analýzu hlavních komponent 3.4 následuje metoda t-SNE 3.5 a celou kapitolu uzavíra, v současnosti nejmodernější, metoda UMAP 3.6.

3.1 Úvod do data miningu

Informace v této sekci byly čerpány z knihy Introduction to Data Mining [14]

Data miningem obecně rozumíme automatické získávání užitečných znalostí z dat. Proces získávání znalostí se dělí do tří hlavních kroků. Vstupní data je potřeba před samotným data miningem upravit. Této části se říká preprocessing a zahrnuje například spojení dat z různých zdrojů, čištění dat, odstranění duplicitních záznamů, normalizaci nebo redukci dimenzionality. Následuje vlastní data mining a po něm postprocessing dat, kdy dochází filtrování, vizualizaci a interpretaci dat například jejich integrací do rozhodovacích systémů. Úlohy dataminingu se dělí na dvě základní:

V případě prediktivních úloh je cílem předpovědět hodnotu určitého atributu, která je nazývána target, na základě znalosti hodnot atributů jiných. Naproti tomu cílem deskriptivních úloh je najít v datech nějaký skrytý význam. Například shluky, korelace nebo anomálie.

Z prediktivních úloh vychází prediktivní modelování jehož náplní je tvorba modelu pro předpověď hodnoty hledané proměnné. Toto modelování se dělí na dva základní přístupy. Klasifikaci a regresi. První se používá v případě diskrétních hodnot. Druhá naopak v případě hodnot spojitých. Používá se například v oblasti prodeje pro předpověď zboží, které si zákazník pravděpodobně koupí na základě zboží zakoupeného v minulosti.

Dále se používá asociační analýza, která slouží pro hledání skrytých vzorců v datech. Cílem této analýzy je najít nejzajímavější vzorce s ohledem na efektivitu. Vzorce jsou reprezentovány asociačními pravidly. To může obsahovat například položky nákupu a zapisuje se jako: repelent -> víno [7]. Takové pravidlo říká, že mezi položkami existuje korelace. Neboli, pokud zákazník koupí repelent je velká pravděpodobnost, že koupí i víno. Asociační analýza se dále používá například pro hledání vztahů mezi jednotlivými elementy klimatického systému naší planety, v bioinformatice pro předběžné zpracování sítí proteinových interakcí za účelem predikce funkce proteinů.

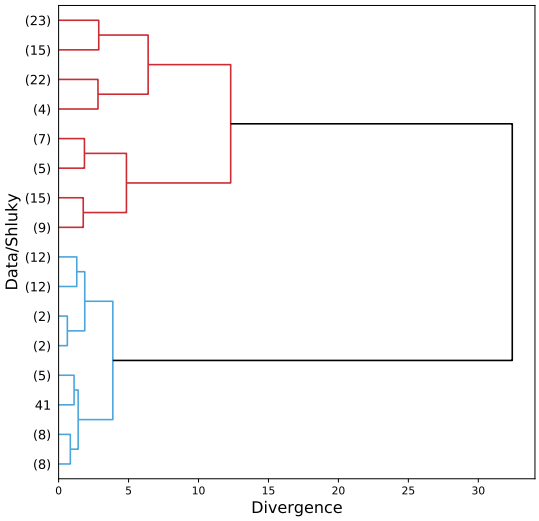
Shluková analýza se zabývá hledáním podobnosti v datech. Používá se například pro klasifikaci objektů do skupin na základě jejich vlastností. Konkrétní příklady můžeme nalézt v biologii kde se používá pro analýzu genetických informací jako hledání genových skupin, které mají podobné funkce [4]. Shlukování se dělí na částečné a hierarchické. V případě částečného shlukování nemohou shluky obsahovat shluky další. V případě hierarchického podshluky obsahují. Takto vnořené shluky tvoří stromovou strukturu, kterou je možné reprezentovat Dendrogramem blíže popsaným v podkapitole 3.2. Částečné shlukování je v této práci reprezentováno metodou K-means, která se snaží najít předem definovaná počet shluků reprezentovaných svými centroidy a je blíže popsaná v podkapitole 3.3

Detekce anomálií jinak nazývaná také detekce deviací hledá data, která se výrazně liší od všech ostatních. To nemusí nutně znamenat, že těchto dat není mnoho. Pokud například nějaká událost nastane v jednom z tisíce případů, kterých je celkem miliarda, jde stále o milion anamálií. Používá se například pro detekci podvodů v bankovnictví. Na základě typického chování majitele kreditní karty lze v případě odcizení detekovat změny v nákupních zvycích zapříčiněné chováním pachatele. Proto je data mining důležitým oborem, který bude v budoucnu stále nabývat na významu.

3.2 Dendrogram

Informace v této podkapitole byly čerpány z knihy Introduction to Data Mining. [14]

Dendrogram se používá pro grafické znázornění míry podobnosti mezi zkoumanými daty při hierarchickém shlukování. Hierarchické shlukování se dělí na dva přístupy. Pro výpočet podobnosti může být použito **aglomerativní** hierarchické shlukování což je takzvaný přístup zdola nahoru.



Obrázek 3.1: Dendrogram

V tomto případě se data nejprve rozdělí do singleton shluků a následně dochází k jejich porovnávání. V každém kroku jsou spojeny dva sobě nejpodobnější shluky na základě určené metriky. Algoritmus končí při dosažení jednoho shluku obsahující všechna zkoumaná

data nebo splněním nějaké podmínky. Výsledek aglomerativního hierarchického shlukování zachycený dendrogramem reprezentuje strom zachycený na obrázku 3.1. Vertikální osa zde značí shluky vstupních dat. Horizontální osa znázorňuje pořadí, v jakém byly nové shluky tvořeny a míru jejich podobnosti.

Další možností je **divizní** hierarchické shlukování neboli přístup shora dolů. Zde je nejprve vytvořen jeden shluk obsahující všechny prvky a v následných iteracích dochází k jeho dělení na shluky menší. Konec dělení nastává v okamžiku, kdy výsledkem iterace budou pouze singleton shluky nebo si budou data dostatečně podobná a další dělení tak pozbude dalšího významu.

Používané metriky

Euklidovská vzdálenost je založena na aplikaci Pythagorovy věty na kartézské souřadnice dvou zjišťovaných bodů. Vzorec pro výpočet vzdálenosti v n-dimenzionálním prostoru bodu A se souřadnicemi $(a_1, a_2, \dots a_n)$ a bodu B se souřadnicemi $(b_1, b_2, \dots b_n)$ by vypadal takto:

$$vzdalenost(A, B) = (a_1 - b_1)^2 + (a_2 - b_2)^2 \dots (a_n - b_n)^2 \tag{3.1}$$

Manhattanská metrika (alternativní názvy: New Yorská metrika, Taxicab Geometry) je odvozena od uspořádání městské zástavby na ostrově Manhattan s čtvercovým půdorysem domovních bloků a z toho vyplyvající sítí pravoúhlých ulic. V zásadě jde o to, že z bodu A do bodu B se lze dostat pouze po rovných čarách, přičemž každé navazující čáry musejí svírat pravý úhel. Vzdálenost dvojice bodů A,B v rovině je rovna součtu absolutních hodnot rozdílu jejich souřadnicových bodů.

$$vzdalenost(A, B) = |a1 - b1| + |a2 - b2| \tag{3.2}$$

Metody hierarchického shlukování

Tato sekce popisuje některé používané přístupy k určení vzdálenosti dvojice shluků. Informace zde uvedené vycházejí z oficiální dokumentace knihovny SciPy [18] Vyjádření divergence shluků může být dáno vzdáleností mezi nejbližšími body porovnávaných shluků. V takovém případě se jedná o metodu **single-linkage** (Nearest Point Algorithm, Nearest Neighbor Clustering Algorithm). Opakem je **complete-linkage** (Farthest Neighbor Clustering Algorithm), kdy jsou použity body nejvzdálenější. Tyto přístupy moho být však zkreslující v případě přítomnosti odlehlých bodů. Tomu lze zamezit použitím metody porovnávající vzdálenosti geometrických středů - Centroidů. Po sloučení je vypočítán **centroid** ze všech bodů nově vzniklého shluku, případně je používán pro tento výpočet **medián** původních středů. Dále lze k výpočtu použít průměrnou vzdálenost mezi všemi body dvou skluků a jiné.

3.3 K-MEANS

Informace v této podkapitole vycházejí z knihy Introduction to Data Mining. [14]
Jedná se o metodu strojového učení bez učitele (unsupervised machine learning). Spolu s aglomerativním hierarchickým shlukováním patří mezi základní metody shlukové analýzy. Oproti metodě K-medoids, při které je pro hledání shluků využiváno jejich nejreprezentativnějšího bodu, jsou zde použity **centroidy**, které jsou na začátku náhodně rozmístěny do prostoru dat a následnou iterací přemístěny do správných pozic vůči vstupním datům. Alternativu představuje například Kohonenova samo-organizační mapa, která pracuje naopak s maticí propojených bodů, na kterou následně iterativně optimalizuje vstupní data.

Vstupní data a parametry

Mějme dataset X ve vícerozměrném prostoru R^d. A počáteční parametr k odpovídající počtu konečných shluků. Pokud je předem známo do kolika skupin jsou vstupní data rozdělena, bude počtu skupin odpovídat i počet požadovaných shluků. Pokud však taková informace dopředu není známa, lze pro stanovení počtu shluků použít například **Elbow Method**. Spočívá v nalezení minimální cílové funkce pro počet shluků definovaný jako uzavřený celočíselný interval. Počet shluků je nepřímo úměrný optimální hodnotě cílové funkce 3.5. Inflexní bod výsledné funkce (Scree Plot) pak odpovídá hodnotě optimálního počtu shluků. Neboli je příhodné zvolit dostatečně nízkou hodnotu při jejímž použití nedochází k prudkému nárustu v součtu kvadratických chyb. Další metodou pro určení optimálního počtu shluků je vypočítání **koeficientu siluety** (Silhouette method) podle vzorce:

$$K_s = \frac{(b-a)}{max(a,b)} \tag{3.3}$$

kde a je rovno průměru všech vzdáleností ve shluku a b je rovno průměru ke shluku nejbližšímu. Nejvyšší hodnota v grafu znázorňujícím průměr všech těchto koeficientů pro danou hodnotu k odpovídá optimálnímu počtu shluků [16].

Inicializace

V prvním kroku je možné místa pro centroidy zvolit náhodně. Následkem toho ovšem nemusí algoritmus vždy konvergovat ke správnému řešení a to zapříčiňuje nutnost více běhů s následným vyhodnocením toho nejlepšího na základě nejnižší hodnoty cílové funkce. Alternativou je použití algoritmu **k-means++**[3]. V tomto případě je náhodně umístěn pouze centroid první. Následně dojde k výpočtu jeho vzdálenosti od všech datových bodů $x \in X$ a následující centroid je zvolen na základě největší vzdálenosti od již ustanovených centroidů. Tento krok odpovídá vzorci

$$\frac{dist(x)^2}{\sum_{x \in X} dist(x)^2} \tag{3.4}$$

který je opakován, dokud není rozmístěno všech k centroidů. Výraz $dist(x)$ zde odpovídá vzdálenosti datového bodu x od nejbližšího centroidu.

Přiřazení

Po prvotní inicializaci je každý datový bod přiřazený nejbližšímu z centroidů. Pro určení vzálenosti se používá standardně **Euklidovská vzdálenost** (viz 3.2). Cílová funkce pro jednodimenzionální dataset, kterou je potřeba minimalizovat, bude obecně odpovídat součtu vzdáleností všech datových bodů od všech centroidů:

$$SEE = \sum_{i=1}^{K} \sum_{x \in C_i} dist(C_i, x)^2 \tag{3.5}$$

kde $dist$ je Euklidovská vzdálenost mezi dvěma objekty v Euklidovském prostoru. Efektivita Euklidovské vzdálenosti však s přibývajícím počtem dimenzí klesá. Proto je v takovém případě vhodné před samotným algoritmem použít některou z metod redukce dimenzí. Další obvyklé používané varianty pak zahrnují Manhattanskou metriku, kvadratickou Euklidovskou vzdálenost aj.

Posun

Na závěr dojde k výpočtu nového váženého průměru všech datových bodů a posunutí všech centroidů do geometrickýchch středů příslušných shluků. Poté se vyhodnotí podmínka pro ukončení algoritmu a buď se opět opakuje přiřazení nebo dojde k ukončení algoritmu. Touto podmínkou může být předem stanovený počet iterací nebo případ kdy ke změně příslušnosti datových bodů k shluku dochází velmi málo, vůbec nebo nastává oscilace datových bodů mezi shluky.

3.4 Principal Component Analysis

Tato podkapitola obsahuje popis metody redukce dimenzionality jménem Analýza hlavních komponent (Alternativní názvy: Principal Component Analysis, Karhunen-Loèveho transformace, Hotellingova transformace). V následujícím textu je při referenci používána zkratka PCA z anglického názvu. Informace v této kapitole byly čerpány z knihy Introduction to Datamining. [14]

PCA je jedna ze základních metod extrakce proměnných a je založená na maticové faktorizaci. Má široké spektrum použití. Používá se k redukci dimenzí vstupních dat před jejich analýzou, pro vizualizaci vícerozměrných dat nebo například při předzpracování vstupních dat pro algoritmy strojového učení. Tím je proces výrazně urychlen, jelikož se není potřeba zabývat méně důležitými proměnnými, které by proces pouze zpomalovaly. Redukce je docíleno vytvořením dimenzí, jež jsou lineární kombinací vstupních dimenzí. "První dimenze je vybrána s ohledem na to, aby zachytila největší možnou variabilitu dat. Druhá dimenze je ortogonální vzhledem k první a zachycuje co nejvíce zbývající variablity dat a tak dále."[14]

Standardizace

Před vlastní analýzou je potřeba provést standardizaci vstupních proměnných. Nutnost tohoto kroku je zapříčiněna různým rozsahem těchto proměnných a to by analýzu mohlo výrazně zkreslit. Cílem PCA je totiž najít takové dimenze, ve kterých mají data největší rozptyl. Příkladem mohou být vzorky dat obsahující dimenzi s intervalovou proměnnou (výška člověka) a dimenzi s binární proměnnou (pohlaví). První zmíněná by v tomto případě dominovala nad druhou, protože její rozptyl bude zcela jistě větší.

Standardizaci lze provést podle následujícího vzorce:

$$Si = (Xi - mean(X))/\sigma(X) \tag{3.6}$$

Si je standardizovaná hodnota, Xi je původní hodnota, $mean(X)$ je průměr hodnot X a σ je směrodatná odchylka hodnot X

Kovarianční matice

Kovariance udává statistickou míru lineární závislosti dvou veličin, neboli "jak moc se dvě veličiny od sebe liší"[14]. Lze ji vyjádřit jako druhou mocninu směrodatné odchylky. Kovarianční matice pak sumarizuje kovarianci mezi každými dvěma veličinami vstupních dat. Pro původně trojrozměrná data tak bude mít matice rozměry 3x3.

Kovariance stejné veličiny Cov(A,A) se nazývá variance (rozptyl). Platí komutativní zákon. Cov(A,B) = Cov(B,A) což znamená, že se bude vždy jednat o symetrickou matici. Kovariance je definována následovně:

$$Cov(A, B) = E[AB] - E[A]E[B]$$

E zde značí střední hodnotu.

"Cílem PCA je najít netriviální lineární kombinaci našich původních proměnných tak, aby kovarianční matice byla diagonální. Když je toto hotové, výsledné proměnné jsou nekorelované, tj. nezávislé."[5]

Vlastní vektory a vlastní čísla

Z kovarianční matice je následně nutné spočítat pro každou dimenzi tzv. vlastní vektor a vlastní číslo, které udává míru variance pro danou dimenzi. Výpočet vyjadřují vzorce:

$$|Cov - \lambda I| = 0 \tag{3.7}$$

kde Cov je kovarianční matice λ jsou vlastní čísla a I je jednotková matice. Spočítáním determinantu dostaneme polynom, jehož kořeny představují hledaná vlastní čísla. Následný výpočet vlastních vektorů vypadá následovně:

$$(A - \lambda_j I)e_j = 0 \tag{3.8}$$

Tímto postupem je pro každou vlastní hodnotu λ_j zjištěn její vlastní vektor e_j.

Hlavní komponenty

Všechny vypočtené vlastní vektory se seřadí podle hodnoty jejich vlastních čísel od nejvyšší po nejmenší. Vlastní vektor s největším vlastním číslem tak představuje směr, ve kterém se data od sebe liší nejvíce. Vektor s druhým nejvyšším vlastním číslem reprezentuje ortogonální vektor vzhledem k prvnímu a zachycuje směr ve kterém mají data druhou nějvětší varianci a tak dále. Tyto vektory jsou nazývány hlavními komponentami. Výběrem výstupní množiny vektorů, s nejvyšší vypovídající hodnotou o původních, vznikne tzv. feature vector. Posledním krokem je úprava původních dat podle os hlavních komponent. Té docílíme vynásobením transponované matice původních normalizovaných dat s transponovanou maticí Feature vektoru:

$$FinalniData = FeatureVector^T * NormalizovanaData^T$$

3.5 t-distibuted Stochastic Neighbor Embedding

Tato podkapitola obsahuje popis metody t-SNE (z anglického t-distributed stochastic neighbor embedding) Informace zde obsažené jsou získány z článku autorů. [12].

Oproti metodě PCA se jedná se o metodu nelineární a relativně novou. Poprvé byla prezentována v roce 2008 duem Laurens van der Maaten a Geoffrey Hinton. Na rozdíl od metody redukce dimenzí PCA, kterou lze použít i pro vizualizaci je t-SNE pro tento úkol přímo navržena. Dalším rozdílem oproti PCA je použití pravděpodobnosti namísto vzdálenosti pro vyjádření podobnosti vstupních dat. Od původní metody SNE prezentované v roce 2002 se pak liší především použitím Studentova t-rozdělení pro vyjádření podobnosti.

Podobnost vícerozměrných dat

V prvním kroku je vypočtena podobnost mezi každými dvěma body vícerozměrných dat $X = x_1, x_2, \ldots, x_n$. Euklidovská vzdálenost je převedena na podmíněnou pravděpodobnost

podle vzorce:

$$p_{j|i} = \frac{exp(-||x_i - x_j||^2/2\sigma_i^2)}{\sum_{k \neq i} exp(-||x_i - x_k||^2/2\sigma_i^2)} \tag{3.9}$$

Jde o vyjádření s jakou pravděpodobností si datový bod x_i zvolí za svého souseda bod x_j. Vyjádřeno v poměru k hustotě jejich pravděpodobnosti při normálním rozložení se středem v bodě x_i.[[cit]] V případě blízkých datových bodů bude podmíněná pravděpodobnost vysoká, naopak pro velice vzdálené body bude extrémě malá.

Podobnost méněrozměrných dat

Množina bodů méněrozměrných dat $Y = y_1, y_2, \ldots, y_n$ je výsledkem náhodného promítnutí množiny vícerozměrných dat do méně dimenzí. Pro účely této práce jsou pro srozumitelnost uvažovány dimenze pouze dvě. Podmíněná pravděpodobnost každých dvou bodů méněrozměrných dat je dána vzorcem:

$$q_{j|i} = \frac{exp(-||y_i - y_j||^2)}{\sum_{k \neq i} exp(-||y_i - y_k||^2)} \tag{3.10}$$

V původní metodě SNE [10] je v tomto bodě použito pro výpočet podobnosti normální rozdělení. To však zapříčiňuje tzv. "crowding problem", kdy při zobrazení vícerozměrných dat do méně rozměrů dochází k jejich zhuštění v důsledku přirozeného nedostatku místa v menším počtu dimenzí. Proto je autory použito již zmíněné Studentovo t-rozdělení s jedním stupněm volnosti. Dělenec se v případě velkých vzdáleností blíží Zákonu převrácených čtverců (viz definice 3.5.1) a tím je zajištěno, že algoritmus téměř není náchylný na změny v měřítku v případě vyhodnocování vzdálených bodů. Další vlastností je stejná interakce v případě dvou vzdálených shluků, tak dvou bodů. Studentovo rozdělení si sebou nese také výhodu absence exponentu a z toho plynoucí rychlejší výpočet.

Definice 3.5.1 (Zákon převrácených čtverců) *Fyzikální zákon, který říká, že pro každý bodový zdroj, který šíří svůj vliv rovnoměrně do všech stran platí, že se zvyšující se vzdáleností od tohoto zdroje klesá intenzita jeho vlivu.*

$$I \propto \frac{1}{d^2} \tag{3.11}$$

kde d je vzdálenost od zdroje a I je intenzita jeho vlivu [9]

Kullback-Leiblerova divergence

Udává divergenci dvou vstupních rozdělení pravděpodobnosti. Cílovou funkci tak definuje vzorec:

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} log \frac{p_{j|i}}{q_{j|i}} \tag{3.12}$$

Prostým dosazením pravděpodobnosti vícerozměrných dat $p_{i|j}$ podle vzorce 3.9 však vzniká problém s odlehlými body, pro které jsou všechny vzdálenosti od ostatních bodů velké a výsledná pravděpodobnost tak příliš malá. Následkem je pak velmi malý vliv na cílovou funkci a tím pádem je výsledná pozice bodu v méně dimenzích hůře definována. Proto v tomto případě zavedeme nové symetrické rozdělení pravděpodobnosti definované jako:

$$p_{i|j} = \frac{p_{j|i} + p_{i|j}}{2n} \tag{3.13}$$

Následkem toho bude mít každy datový bod na cílovou funkci významný vliv.

Cílem t-SNE je minimalizovat divergenci za použití metody Klesání podle gradientu (Gradient descent), kterou lze pro KL divergenci vyjádřit jako:

$$\frac{\delta C}{\delta y_i} = 2\sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j) \tag{3.14}$$

Pro každou iteraci pak bude vzorec definován jako:

$$\gamma^{(t)} = \gamma^{(t-1)} + \eta\frac{\delta C}{\delta \gamma} + \alpha(t)(\gamma^{(t-1)} - \gamma^{(t-2)}) \tag{3.15}$$

kde k výpočtu gradientu je přidán ještě moment. To má za následek rychlejší optimalizaci a zároveň zabraňuje algoritmu v nalezení neoptimálního lokálního minima.

Perplexita

V případě t-SNE ovlivňuje perplexita výslednou hodnotu rozptylu Gaussova rozdělení pravděpodobnosti při počítání podobnosti datových bodů vícerozměrných dat. Je obecně definována vzorcem:

$$Perp(P_i) = 2^{H(P_i)} \tag{3.16}$$

kde H(Pi) značí Shannonovu entropii Pi v bitech:

$$H(P_i) = -\sum_j p_{j|i} \log_2 p_{j|i} \tag{3.17}$$

Z hlediska různé hustoty dat však nelze předem ideální hodnotu stanovit. V oblastech s velkou hustotou dat je vhodnější nižší hodnota perplexity a následného rozptylu. U oblastí s malou hustotou je tomu naopak. Dá se tedy s určitostí předpokládat, že pro konkrétní data bude vhodné s hodnotami experimentovat. Obecně je však podle doporučení autorů vhodné používat hodnoty v intervalu <5,50>.

Míra učení (learning rate, epsilon)

Stejně jako u perplexity ideální hodnotu nelze stanovit. Příliš malá hodnota epsilon způsobí velkou hustotu výslednych dat. Naopak hodnota příliš velká zapříčiní pravý opak. Lokální shluky dat budou algoritmem velmi brzy rozděleny a podoba výsledných dat tak bude velmi řídká. [17]

3.6 Uniform Manifold Approximation and Projection

Tato podkapitola poskytuje stručný popis metody redukce dimenzionality UMAP neboli Uniform Manifold Approximation and Projection. Informace pro tuto kapitolu byly čerpány z článku autora Lelanda McInnese z roku 2018. [13]

Stejně jako metoda t-SNE je UMAP metoda nelineární a řadí se do skupiny metod redukce dimenzí používající grafy nejbližšího souseda (z anglického: nearest neighbor graph). Metoda se dělí na dvě fáze. Konstrukci grafu z vícerozměrných dat a následně optimalizaci grafu s méněrozměrnými daty.

Konstrukce grafu z vícerozměrných dat

Pro reprezentaci vícerozměrných dat je použit váhový graf k-nejbližších sousedů, kde vrcholy reprezentují datové body a hrany s váhami udávající pravděpodobnost propojení konkrétních dvou bodů. Mějme množinu bodů vstupního datasetu $X = x_1, x_2, \ldots, x_n$ Nejprve je potřeba vypočítat pro každý bod set jeho nejbližších sousedů za použití nearest neighbor descent algorimu. Základem je předpoklad, že "soused mého souseda bude pravděpodobně i můj soused". [15] Výsledkem je množina $x_{i_1}, x_{i_2}, \ldots, x_{i_n}$ pro každé x, kde n je vstupní hyperparametr udávající počet nejbližších sousedů. Pro každé x_i je spočítáná vzdálenost od nejbližšího souseda. Ta je tak dána vzorcem:

$$\rho_i = min\{dist(x_i, x_{i_j}) | 1 \leq j \leq k, dist(x_i, x_{i_j}) > 0\} \tag{3.18}$$

Při následném nastavení směrodatné odchylky rozdělení všech ostatních bodů ve shluku i musí platit rovnost:

$$\sum_{j=i}^{k} exp\left(\frac{-max(0, dist(x_i, x_{i_j}) - \rho_i)}{\sigma_i}\right) = \log_2(k) \tag{3.19}$$

Nastavením ρ_i je zajištěno, že každý bod bude spojen s nejméně jedním dalším bodem. "Prakticky se výrazně zlepší znázornění vícerozměrných dat oproti metodě t-SNE, která v těchto případech začíná trpět prokletím dimenzionality." [13] "Prokletí dimenzionality je vyjádřením všech jevů, které se objevují u vicerozměrných dat a které mají nejčastěji neblahé důsledky na chování a výkon algoritmů strojového učení."[19]

Následně je vypočten graf k-nejbližších sousedů definovaný jako trojice $G = (V, E, w)$, kde V jsou vrcholy odpovídající bodům vstupního datasetu, E je množina orientovaných hran $\{(x_i, x_{i_j}) | 1 \leq j \leq k, 1 \leq i \leq N\}$ a w je váhová funkce definovaná vzorcem:

$$w((x_i, x_{i_j})) = exp\left(\frac{-max(0, dist(x_i, x_{i_j}) - \rho_i)}{\sigma_i}\right) \tag{3.20}$$

Výsledkem je indukovaný graf pro každé x_i. Posledním krokem této části je spojení lokálních grafů do jednoho celku. Tím je vytvořen z topologického prostoru tzv. **simplicial complex**.

Konstrukce grafu méněrozměrných dat

Tento krok popisuje vytvoření ekvivalentního méněrozměrného grafu. Zde UMAP používá vzv. **force directed graph layout**. Ten reprezentuje datové body jako uzly které se navzájem odpuzují a jednotlivé hrany jako přitažlivé síly mezi nimi [13] Přitažlivou sílu vrcholů i a j vyjadřuje vzorec:

$$\frac{-2ab||y_i - y_j||_2^{2(b-1)}}{1 + ||y_i - y_j||_2^2} w((x_i, x_j))(y_i - y_j) \tag{3.21}$$

kde y_i a y_j jsou souřadnice. A a b jsou hyperparametry. Odpudivá silá je vyjádřena vzorcem:

$$\frac{2b}{(\epsilon + ||y_i - y_j||_2^2)(1 + a||y_i - y_j||_2^{2b})}(1 - w((x_i, x_j)))(y_i - y_j) \tag{3.22}$$

kde ϵ je malé číslo zabraňující dělení nulou. Pro porovnání obou rozdělení pravděpodobnosti je zde využita jako cílová funkce Křížová entropie namísto KL-Divergence, která se používá pro tento účel v případě metody t-SNE.

Hyperparametry

Počet komponent udává počet výsledných dimenzí. **Metrika** zajišťuje jakým způsobem bude počítána vzdálenost mezi každými dvěma body v prostoru. **Počet nejbližších sousedů** je parametr použitý při konstrukci vícerozměrného grafu v první části metody. Nízká hodnota zapříčiní rozpoznání drobnějších detailů vstupních dat. Vysoká hodnota způsobí zaměření na celkovou strukturu dat. **Minimální vzdálenost** udává jak blízko u sebe budou znázorněny body v ménědimenzionálním grafu. Nízká hodnota zapříčiní vyšší hustotu dat s ohledem na větší věrohodnost jejich reprezentace.

Kapitola 4

Nástroje pro extrakci

V rámci důkladného otestování v současné době dostupných nástrojů pro extrakci dat z pdf v jazyce python jsem použil následující typy testovacích souborů:

- **Výstupní zprávy** generované v laboratorních zařízeních, konkrétně plynových chromatografech. Tvoří je záhlaví, zápatí, hlavička, graf a tabulka s daty zabírající jednu či více po sobě jdoucích stran.
- **Elektronická kniha** obsahující kromě několika grafických prvků výhradně text, bez tabulek, rozsah cca. 700 stran.
- **Daňový formulář** W8-BEN s extenzivním množstvím textu v několika různých formátech organizovaných do logických bloků oddělených grafickými prvky, rozsah 1 strana.
- Heslem chráněný **výpis z banky** tvořený obrázkem a dvěma tabulkami obsahující hlavičku a samotná data, rozsah jednotky stran.

Poslední tři položky byly použity nad rámec zadání.

PyPDF2

Jde o velice jednoduchou knihovnu, která umožňuje extrakci, spojování a psaní pdf souborů. Extrakce dat se však neobešla bez komplikací. Tento nástroj často ignoruje některý text na základě jeho velikosti. Občas dochází k vynechání prvního a posledního řádku. Podle mých zkušeností se hodí pouze k extrakci prostého textu bez grafických prvků, popřípadě velice jednoduchých formulářů, nikoli však pro extrakci tabulek.

pdfPlumber

Nástroj pdfPlumber je postaven na komunitně spravovaném forku knihovny **pdfminer.six**, která vychází z dnes již neudržované knihovny pdfminer. Nejlepší výsledky v extrakci tabulek a excesivní možnosti nastaveni byly příčinou použití této knihovny v závěrečné implementaci. Jako jediný obsahuje systém grafického ladění viz obrázek 4.1 jehož zhoršená kvalita je dána výstupem knihovny **ImageMagick**.

tabula-py

Oficiální dokumentace charakterizuje tuto knihovnu jako: "Jednoduchý Python wrapper vytvořený z tabula-java, který umí načítat tabulky z PDF. Můžete číst tabulky z PDF a převádět je do datového rámce pandas. tabula-py také umožňuje extrahovat soubor PDF

přímo do souboru CSV / TSV / JSON."[2]. Pracuje ve třech volitelných modech lattice, stream a guess. Offiální dokumentace uvádí lattice vhodný pro klasické tabulky oddělené vodícími čarami. Stream jako vhodnější v případě absence vodících čar. Pokud není explicitně extrakční metoda zadána je standardně nastavena guess, která si ve všech testech většinou poradila. Dále je možné nastavit extrakci z výřezů a ty pak sloučit do jednoho. Ve výchozím nastavení je extrahována pouze první tabulka z první strany dokumentu. Toto nastavení lze modifikovat pomocí parametu pages. V případě, že dokument obsahuje více tabulek, lze je extrahovat najednou pomocí parametru multiple tables. Výstupem je v tomto případě list dataframů. Bohatě formátované tabulky však nebyl schopen extrahovat. Ta již ovšem překročila rámec formy běžně zpracovávaných PDF.

pdftotext + Natural Language Toolkit

Kombinace těchto dvou nástrojů se hodí pouze pro analýzu textu, nikoli extrakci dat z tabulek. Knihovna **pdftotext** extrahuje data, která je možno procházet po stránkách. Pomocí **NLTK** je možné tato data rozdělit na tokeny a následně analyzovat.

134	6.3	2.8	5.1	1.5	Iris-virginica
135	6.1	2.6	5.6	1.4	Iris-virginica
136	7.7	3.0	6.1	2.3	Iris-virginica
137	6.3	3.4	5.6	2.4	Iris-virginica
138	6.4	3.1	5.5	1.8	Iris-virginica
139	6.0	3.0	4.8	1.8	Iris-virginica
140	6.9	3.1	5.4	2.1	Iris-virginica
141	6.7	3.1	5.6	2.4	Iris-virginica
142	6.9	3.1	5.1	2.3	Iris-virginica
143	5.8	2.7	5.1	1.9	Iris-virginica
144	6.8	3.2	5.9	2.3	Iris-virginica
145	6.7	3.3	5.7	2.5	Iris-virginica
146	6.7	3.0	5.2	2.3	Iris-virginica
147	6.3	2.5	5.0	1.9	Iris-virginica
148	6.5	3.0	5.2	2.0	Iris-virginica
149	6.2	3.4	5.4	2.3	Iris-virginica
150	5.9	3.0	5.1	1.8	Iris-virginica

Obrázek 4.1: pdfPlumber: Grafické ladění

Kapitola 5

Implementace

Následujicí kapitola pojednává o implementaci koncové webové aplikace sdružující všechna výše popsaná témata. Informace o knihovnách Dash a Plotly byly čerpány z jejich officiální dokumentace [6]. Informace o platformě Heroku pak z její oficiální stránky [8].

5.1 Struktura aplikace

Výslednou implementaci jsem se rozhodl pojmout jako již zmíněnou webovou aplikaci. A to hlavně z důvodů dnes všudypřítomné dostupnosti napříč platformami bez nutnosti instalace. K tomu jsem využil již zmíněné knihovny Plotly a Dash jazyka Python. Framework Dash je přímo navržen pro vytváření webových analytických aplikací. Do jednoho celku sdružuje webový framework Flask, grafickou knihovnu Plotly.js a knihovnu pro vytváření interaktivních uživatelských rozhraní React.js. Aplikace je členěna na několik samostatných stránek, kde každá stránka obsahuje implementaci jedné analytické metody. Samotná struktura projektu na úrovni složek odpovídá strukturám rozložení webových aplikací Django, či Flask a znázorňuje ji obrázek. 5.1.

```
extraktor/
  |— app.py
  |— index.py
  |— parser.py
  |— apps
       |— __init__.py
       |— pca.py
       |— dendrogram.py
       |— tsne.py
       |— umap.py
       |— graph_settings.py
```

Obrázek 5.1: Struktura aplikace

Soubor `app.py` slouží pouze pro vytvoření serveru po inicializaci instance třídy Dash a definici meta tagů pro responsivní design s ohledem na mobilní zařízení. Poté je aplikace importována do souboru `index.py`, který slouží jako její hlavní stránka. Následuje balíček `apps`, který obsahuje popis ostatních stránek. Soubor `init.py` může obecně obsahovat inicializační kód. V tomto případě je však prázdný a slouží pouze jako označení balíčku. V souboru `parser.py` jsou implementovány funkce zodpovídající za samotnou extrakci

dat z vložených souborů. V počátcích návrhu jsem uvažoval s co největší univerzálnosti aplikace. Tyto úvahy se však ukázaly jako liché, hlavně vzhledem k náročnosti výsledné implementace.

5.2 Dash komponenty

Vzhled aplikace neboli layout se skládá z vizuálních komponent popsaných dvěma třídami organizovaných do stromové struktury. Třída `dash_html_components` obsahuje ekvivalenty všech html elementů popsaných v jazyce Python. Pro znázornění načítání jednotlivých grafů jsem například použil element `dcc.Loading`. Je však možné se html úplně vyhnout a použít k sestavení webové aplikace pouze knihovnu `dash_core_components`, která obsahuje základní elementy pro vstup dat (upload) a jejich výběr (slider, range slider, dropdown). Tato implementace však využíva kombinaci obou.

Komponenty jsou mezi sebou propojeny tzv. callbacky. Jedná se o Python dekorátor s argumenty `Input`, `Output` a `State`. Jak názvy napovídají argumenty Input a Output slouží k definici vstupních a výstupních komponent. Argument State se liší pouze tím, že komponenty zde definované nezapříčiňují spuštění callbacku při jejich změně. Následně definovaná funkce (převzatá též z oficiální dokumentace) se provede vždy, když dojde ke změně hodnoty v některé ze vstupních komponent.

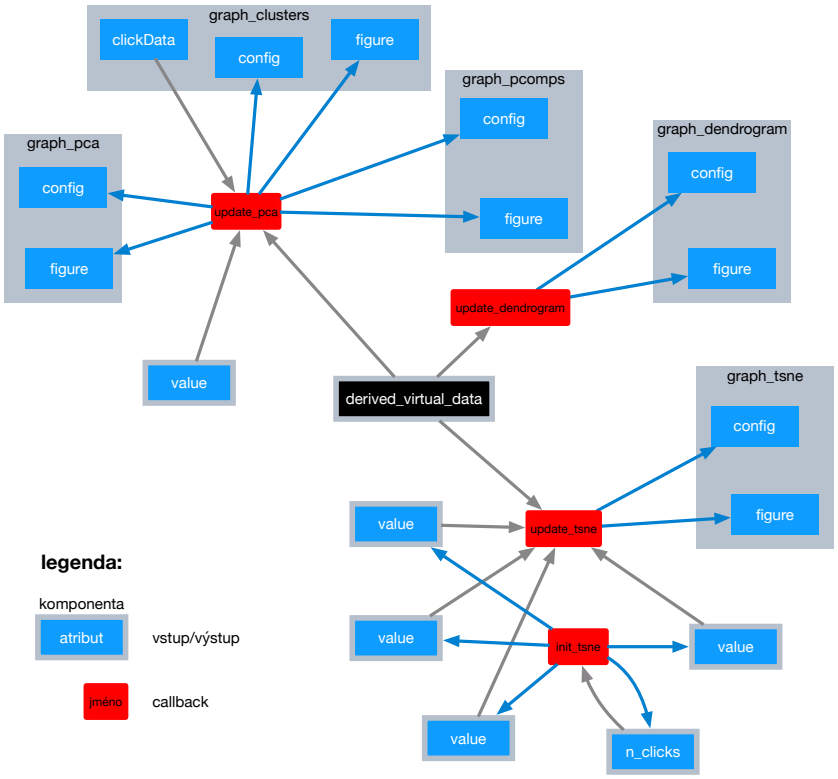
```
@app.callback(
    Output(component_id=my-output, component_property=children),
    Input(component_id=my-input, component_property=value)
)
def update_output_div(input_value):
    return Output: {}.format(input_value)
```

Na obrázku 5.2 je znázorněn strom callbacků pro generování grafů PCA, Dendrogramu a t-SNE. Vstupem callbacků generujících grafy jsou vždy data z tabulky (na obrázku znázorněné atributem `derived_virtual_data`) plus případný výstup z elementu dále modifikující výslednou podobu grafu. Callback funkce je však zavolána až při návštěvě stránky s konkrétním grafem, nedochází tak ke generování všech grafů v okamžiku nahrání dat do tabulky, která je reprezentována třídou `dash_table.DataTable`.

5.3 Vstupní data

Každý soubor je po importu rozdělen na tokeny knihovnou `pdftotext`, které jsou následně prohledány na shodu s klíčovými slovy, která slouží pro identifikaci typu konkrétního dokumentu. Následně je pro každý typ zavolána funkce na míru, která extrahuje data z dokumentu do struktury `pandas.DataFrame`.

Pro samotnou extrakci jsem použil knihovnu `pdfPlumber`, která v porovnání s ostatnímy knihovnami určenými ke stejnému účelu obstála v případě vstupních dokumentů nejlépe. Samotná extrakce však může dosahovat délky jednotek sekund, což má neblahý vliv na uživatelský prožitek. Proto jsem alespoň pro rychlejší opětovnou extrakci stejných dat implementoval cache paměť za pomocí knihovny `pickle`. Před každou extrakcí se tak algoritmus dotáže, jestli již byla data extrahována. V případě kladné odpovědi načte pro každý soubor předzpracovaná data do datové tabulky z pickle archivu. V případě záporné odpovědi extrahuji vybrané sloupce do datové struktury pandas.DataFrame. Provedu předzpracování a data uložím do cache. V obou případech dojde následně ke konkatenaci všech dat do



Obrázek 5.2: Strom s callbacky

tabulky jedné a k výběru sloupce pro extrakci, který byl předán jako argument funkce. Poté jsou už data převedena do struktury `dashtable.DataTable`. Jedná se o komponentu, která umožňuje široké možnosti podmínečného formátování. Tabulka zůstává umístěna za všech okolností nad footerem stránky. Na základě výběru z hlavního menu se mění pouze zobrazovaný graf. Takto má uživatel vstupní data vždy na dosah. Ze škály nastavení pro tabulku jsem kromě formátu sloupců použil dělení na stránky, konkrétně po 20 řádcích, což mi přišlo jako optimální velikost. Tím je také zajištěna pevná výška za všech okolností. Další výhodou je rychlejší a uživatelsky příjemnější práce s velkými daty. Dále má uživatel možnost filtrovat a editovat data, což se okamžitě projeví na výsledné podobě grafu.

V současné implementaci je možné pro vstup použít soubory typu pdf, nebo použít vzorová data pro případ demonstrace. Tato data byla poskytnuta zadavatelem. Z nich jsem vybral skupinu tří datasetů lišících se především velikostí, abych demonstroval použití aplikace v závislosti právě na velikosti vstupních dat. Následně je možné použít jednotlivé analytické metody.

5.4 Analýza a grafy

Každé metodě je věnována samostatná stránka a samostatný soubor pro snadnou rozšiřitelnost. Odkazy na jednotlivé stránky jsou realizovány elementy `dcc.Link` v navigačním panelu umístěném za všech okolností na vrcholu stránky. Každý soubor popisující stránku tvoří dvě základní části. Popis vzhledu a definice funkcí. Multi-stránkové rozložení aplikace je vyřešeno pomocí callbacku s funkcí `goto_page` jehož spuštění je vyvoláno stiskem komponenty `dcc.Link` z navigačního panelu při výběru konkrétního grafu. Následným vět-

vením podle vstupní hodnoty `pathname` je vybrán layout zvoleného grafu a ten je předán do skryté komponenty `html.Div`, která se nachází těsně pod navigačním panelem a byla dosud skryta.

Více možnými vstupy je dána potřeba zjistit, která komponenta byla updatována. Toto lze zařídit pomocí příkazu:

```
dash.callback_context.triggered[0][prop_id].split(.)[0]
```

který vrací jméno poslední updatované komponenty. Následným větvením je pak zajištěn vstup z více zdrojů. Po zobrazení dat v tabulce má uživatel možnost extrahovat alternativní sloupce pomocí elementu `dcc.Dropdown`. V souvislosti s touto funkcí vznikla potřeba předání dat z jednoho callbacku do dalšího. Konkrétně šlo o jména zpracovávaných souborů. To jsem vyřešil elementem `html.Div`, který má po celou dobu vlastnost display nastavenou na hodnotu none a slouží pouze pro uložení těchto dat. Existuje více možností jak toto předání dat zajistit ovšem vzhledem k malému objemu dat jsem zvolil tu nejjednodušší. V samotném callbacku je potom tento element zastoupen právě v argumentu State, který nemá na spuštění callbacku vliv.

Metodu PCA jsem kromě vlastního grafu obohatil o dva další pomocné grafy provázané s hlavním. Jednak jde o graf znázorňující první čtyři hlavní komponenty a jejich varianci (`graph_pcomps`). Hlavní graf je inicializován se dvěma hlavními komponentami, avšak uživatel má možnost dodatečně zvolit zobrazení dalších pomocí komponenty dcc.Slider, který se nachází pod grafem. Konkrétně dvou, tří nebo čtyř prvních hlavních komponent. Druhý graf pak obohacuje původní PCA metodou K-Means (`graph_clusters`). Jde o tzv. scree plot znázorňující pro jednotlivé počty shluků součet druhých mocnin vzáleností datových bodů od nejbližšího centroidu (inertia). Uživatel tak může vybrat požadovaný počet shluků do kterých chce výstup metody PCA dodatečně rozdělit. Aktualizace grafu s PCA je realizována přes callback s názvem `update_pca`, který aktualizuje PCA graf vždy při změně dat v tabulce, hodnoty slideru reprezentující počet komponent nebo označením počtu shluků přímo v grafu K-Means. V době psaní této bakalářské práce dochází vždy k aktualizaci všech tří grafů pokáždé, když potřebuje být aktualizován pouze graf hlavní. Toto je bohužel zapříčiněno současnými nedostatky ze strany knihovny dash. Samotnou ukázku rozložení prvků na stránce pro metodu PCA zachycuje obrázek 5.3.

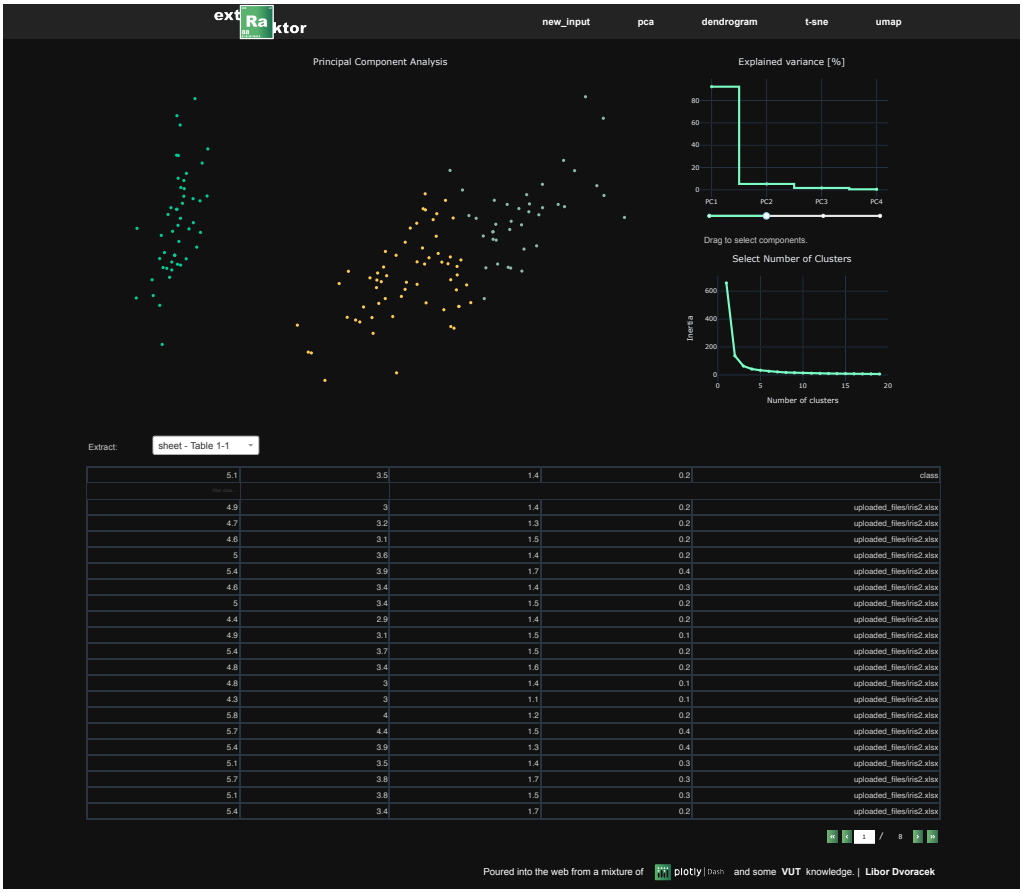
Graf znázorňující Dendrogram jako jediný nemá žádné pomocné grafy, ani jiné vstupy upravující jeho podobu. Vyplývá to ze skutečnosti, že kromě orientace grafu nejsou žádné další parametry k dispozici.

Nad rámec původního zadání jsem implementoval metotu t-SNE. Zde jsem nepoužil žádné pomocné grafy, jako v případě metody PCA, ale zaměřil jsem se více na kvalitu samotného zobrazení, které pokrývá přes 90 procent šířky celé stránky. Namísto dalších grafů jsem zajistil interaktivitu grafu implementací boční lišty s volitelným nastavením hyperparametrů. Při zobrazení stránky se prvotní graf inicializuje s hodnotami doporučenými autory původního článku o metodě t-SNE. Uživatel však může tyto parametry následně modifikovat pro lepší pochopení zobrazovaných dat a v případě nezdaru obnovit původní hodnoty tlačítkem Reset.

Poslední implementovanou metodou je UMAP. Zde jsem zvolil stejný přístup jako u předešlé metody. Změna je patrná pouze v použitých hyperparametrech. Stejně jako u předešlé metody jsem se při inicializaci hodnot držel doporučení autora.

Pro všechny grafy kromě PCA a Dendrogramu jsem použil třídu `plotly.express`. Graf pro metodu PCA je jako jediný generován pomocí třídy tvořené několika struktu-

rami `plotly.graph_objects`. To je nástupce třídy `plotly.graph_objs`. Tato třída nabízí, na rozdíl, od třídy plotly.express širší možnosti nastavení.

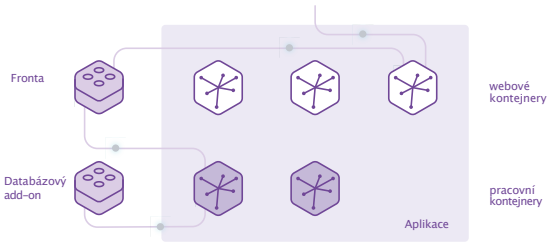


Obrázek 5.3: GUI

Pro generování Dendrogramu je pak použita třída `plotly.figure_factory`. Z dostupných konfiguračních vlastností grafu jsem využil pouze funkci zoom, která je nastavena defaultně u všech hlavních grafů. Pomocné grafy pak nemají vlastnosti žádné, protože by se v jejich případě míjely účinkem. Tyto dva způsoby konfigurace, stejně jako nastavení barevné palety grafů, se nachází v samostatném souboru `graph_settings.py`.

5.5 Nasazení aplikace na server

Pro vlastní nasazení aplikace jsem použíl cloudovou `Platform as a Service Heroku` založenou na kontejnerech a Python Web Server Gateway Interface Green Unicorn HTTP Server. Pro samotnou správu kódu používám Heroku CLI na rozdíl od propojení přímo s repozitářem služby GitHub. Po nahrání aplikace na server obvyklými git příkazy bylo nutné vytvořit buildpack, který slouží k transformaci kódu na tzv. slug. Tím je zajištěn běh aplikace v Heroku kontejnerech nazývaných Dynos 5.4. Aplikace dostane požadavek. Ten je předán některému z webových kontejnerů, který zařadí požadavek do fronty a uživateli je zaslána zpráva o úspěšném doručení. Pracovní kontejner následně výjme požadavek z fronty a provede ho. Případně ještě výsledek uloží do databáze, čímž může být výsledek uživateli doručen jako součást jiného požadavku.



Obrázek 5.4: Heroku Dynos

Kromě standardního buildpacku `heroku/python` bylo nutné použít experimentální buildback `heroku-comunity/apt`. Některé knihovny totiž nebylo možné nainstalovat bez specifikace jejich repozitářů v souboru Aptfile. Pro prvotní testy jsem použil základní kontejner s 512MB paměti RAM. Aplikaci je však možno škálovat jak horizontálně přidáním více kontejnerů, tak vertikálně použitím kontejnerů výkonějších. Při použití základní konfigurace kontejneru však zobrazení grafu některých metod zabralo neúměrné množství času a velice často docházelo k timeoutu ze strany serveru. Proto jsem se rozhodl použit konfiguraci vyšší, konkrétně verzi 2X s 1GB RAM. To zapříčinilo výrazné zkrácení dob načítání v řádu několika desítek procent.

Kapitola 6

Testování a experimenty

6.1 Testování

Tato sekce popisuje testování lokální i online verze aplikace.

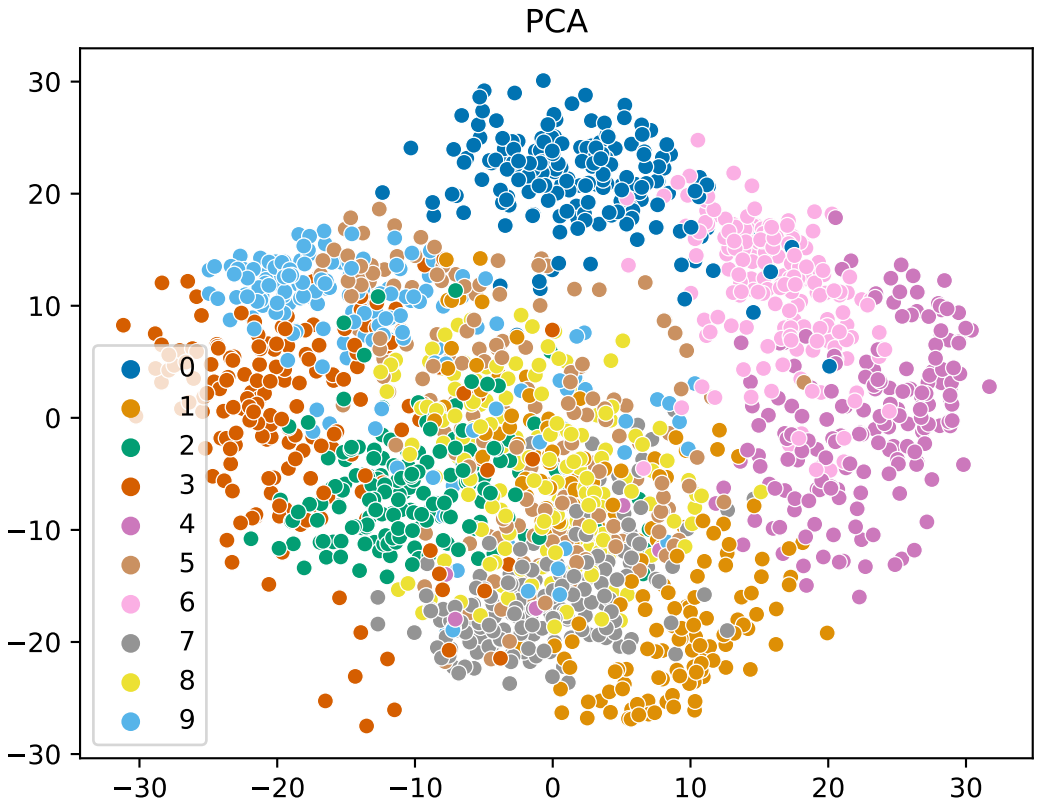
Testování jsem rozdělil do několika kategorií. Zaprvé jsem testoval funkcionalitu aplikace testem všech interních tlačítek a odkazů. Samotné input elementy testovat potřeba nebylo, jelikož při definici jsou jako vstupy povolena pouze čísla ve vhodném formátu. Uživatel tak není schopen zadat jakoukoliv neplatnou hodnotu.

Vstupy aplikace jsem otestoval skupinou souborů různých typů mimo vlastní pdf, pro které je aplikace vytvořena. Chybová hlášení spojená s nepoužitelnými soubory, jakož i všechna ostatní jsou uživateli skryta a vypisována pouze do konzole. Dále jsem testoval grafické uživatelské rozhraní aplikace a vykreslení grafů. To odhalilo nejzávažnější chybu. Graf metody t-SNE nelze v době psaní tohoto textu vykreslit v požadované době 30 sekund a následně dochází k timeoutu ze strany serveru.

Testování kompatibility jsem provedl na kombinacích operačních systémů Windows 10 a MacOS a internetových prohllížečů Safari, Google Chrome a Mozilla Firefox. Žádný z těchto testů však nepřinesl objev výraznější odchylky od požadavaných funkčních či grafických vlastností aplikace.

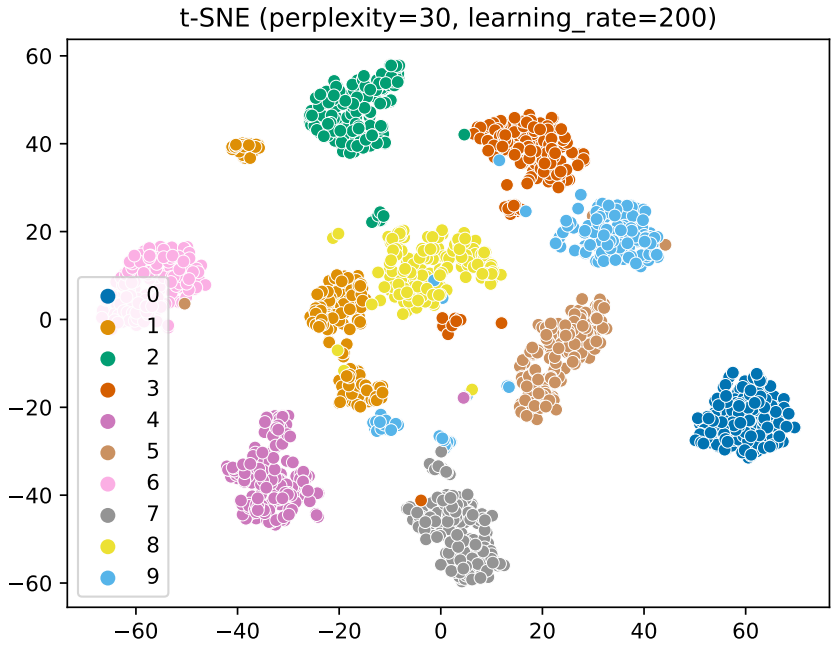
6.2 Experimenty

Tato sekce obsahuje porovnání metod pro redukci dimenzionality popsaných v kapitole 3. Jako vstupní dataset jsem použil importovaný **The Digits Dataset** z knihovny scikit-learn. Jedná se o 1797 obrázků číslic v rozlišení 8x8 pixelů. Na obrázku 6.1 je pro zobrazení použit metoda PCA.

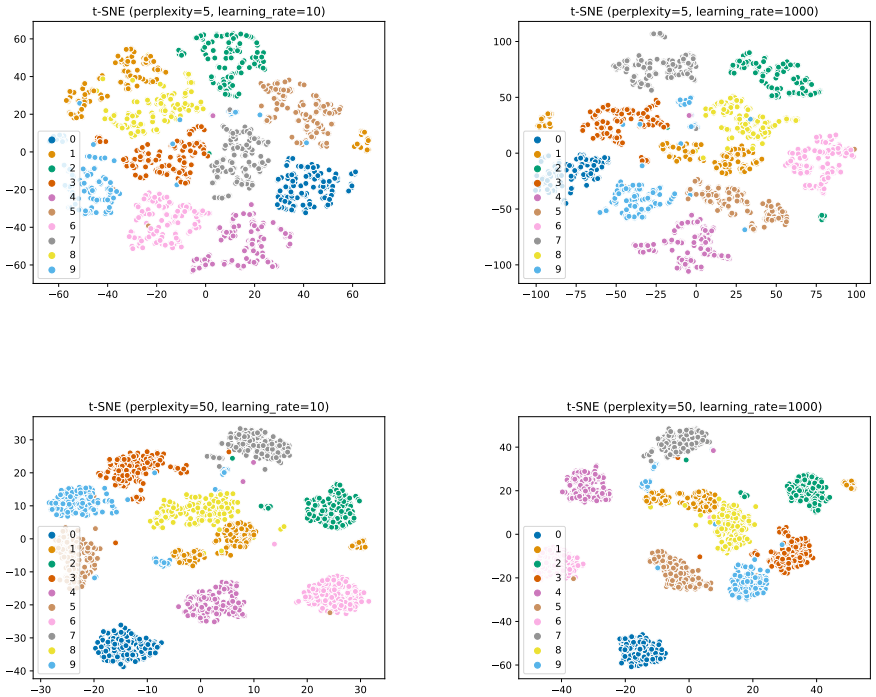


Obrázek 6.1: PCA

Zde je vidět nedostatečná klasifikace číslic 3, 8 a 9. Oproti tomu metoda t-SNE si v základním nastavení s perplexitou 30 a mírou učení 200 poradila mnohem lépe, jak je patrné z obrázku 6.2



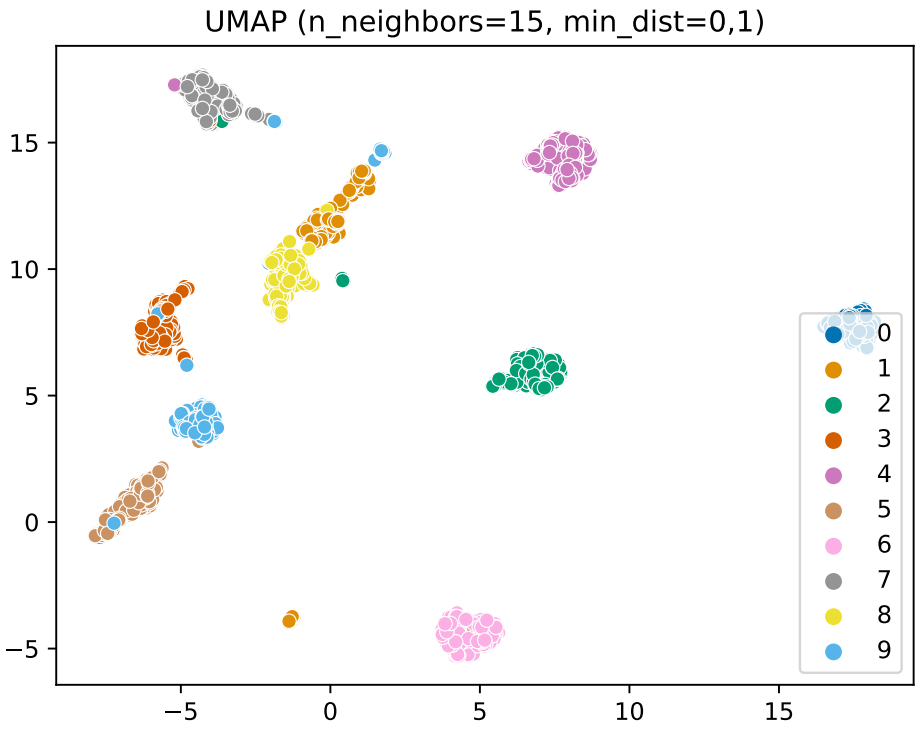
Obrázek 6.2: t-SNE



Obrázek 6.3: t-SNE hraniční hodnoty hyperparametrů

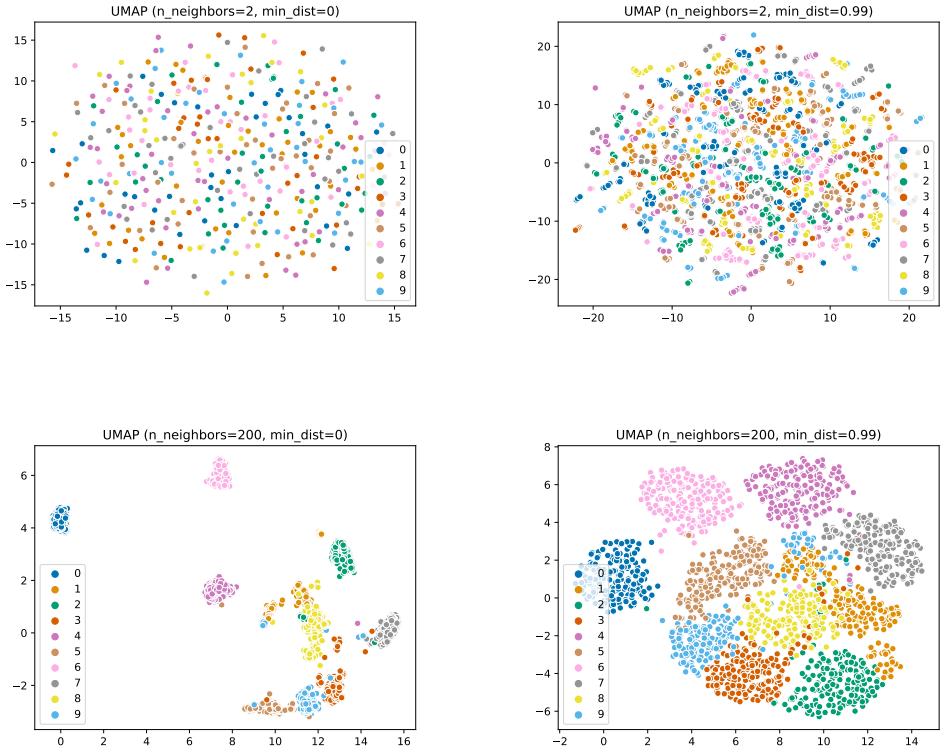
Přesto však v tomto případě došlo k příliš velkému zhuštění dat, a proto bude potřeba pro věrohodnější zachycení zobrazovaných dat nutné nastavit parametr perplexity směrem dolů. Grafy s nastavením všech hraničních hodnot hyperparametrů zachycuje obrázek 6.3. Zde se ukazuje, že nízká hodnota perplexity je stěžejním parametrem pro vizualizaci vstupního datasetu.

Na obrázku 6.4 je zachyceno zobrazení metodou UMAP. Zde došlo opět v velkému zhuštění dat a ztrátě informace. Standardně nastavená hodnota minimální vzdálenosti je tudíž až příliš nízká.



Obrázek 6.4: UMAP

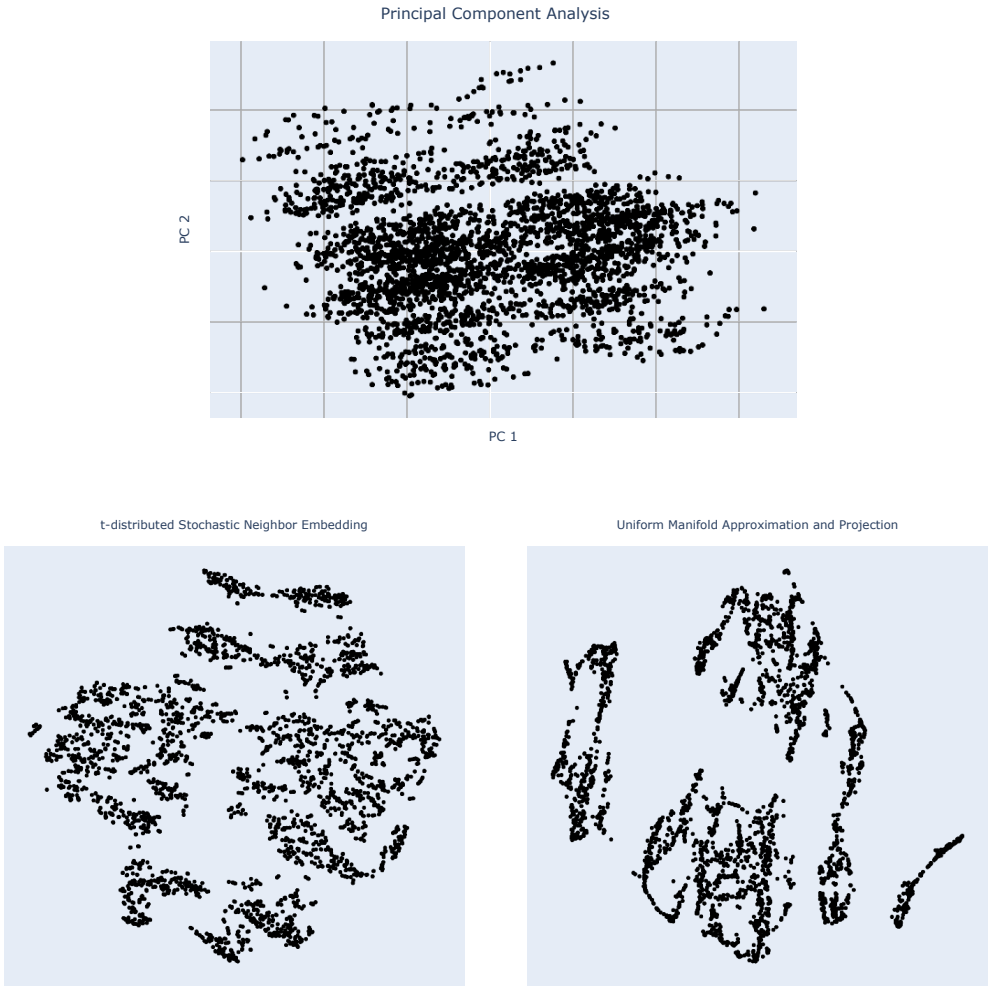
Obrázek 6.5 znázorňuje zobrazení dat pro hraniční hodnoty hyperparametru počet sousedů a minimální vzdálenosti. Je zjevné, že metoda UMAP je na nastavení hyperparametrů náchylná mnohem více, než je tomu i metody t-SNE. V případě příliš nízkého počtu sousedů jsou výsledná zobrazení daleko od požadovaného výsledku. Naproti tomu při zvětšení parametru minimální vzálenosti bodů lze dosáhnout, v případě tohoto demonstračního datasetu, velice příznivých výsledků.



Obrázek 6.5: UMAP - hraniční hodnoty hyperparametrů

Z výsledků lze vyvodit, že obě novější metody jsou oproti metodě PCA ve vizualizaci dat na vyšší úrovni. Velkou nevýhodou metody UMAP oproti t-SNE je délka času potřebného pro výpočet.

Pro další ilustraci zde uvádím také výstupy metod redukce dimenzionality přímo z výsledné aplikace vypočítané a zobrazené pro největší testovací dataset dodaný zadavatelem. Ve všech případech jsem použil výchozí nastavení parametrů. Jedná se o 3032 vzorků chemické sloučeniny s názvem Melamin. Tyto obrázky bohužel postrádají požadovanou úroveň kvality, která je zapříčiněna nemožností exportovat vektorovou grafiku z knihovny plotly. I přesto je zde jasně patrné mnohem vyšší úroveň separace dat u metod t-SNE a UMAP na rozdíl od metody PCA viz obrázek 6.6



Obrázek 6.6: Výstupy aplikace extRaktor

Kapitola 7

Závěr

Původním cílem této bakalářské práce bylo seznámení s knihovnami jazyka Python pro extrakci dat ze souborů PDF. Tento bod jsem splnil studiem a praktickými testy několika nejpoužívanějších knihoven. Závěry z toho plynoucí jsem popsal v kapitole 4. Bod zadání týkající se seznámení s analytickými metodami používanými k redukci dimenzionality jsem splnil převážně četbou odborných prací jejich autorů. Tyto metody jsem pak popsal v kapitole 3. Následně jsem metody otestoval na sadě demonstračních dat a porovnal v kapitole 6. Při rozmýšlení nad podobou výsledné aplikace jsem se seznámil s knihovnami plotly a dash, které jsou navrženy právě pro vytváření analytických webových aplikací a z mých zkušeností při práci s nimi není implementace v této bakalářské práci určitě posledním projektem, který v nich vytvořím. Výsledná implementace z posledního bodu zadání je popsána v kapitole 5. V případě webové aplikace se mi bohužel ke dni odevzdání této práce nepodařilo implementovat spolehlivě fungující metodu t-SNE, která po nasazení online trpí velmi častými timeouty, přestože lokálně funguje bez nejmenších problémů. Přesto byl však původní záměr práce splněn, protože metoda t-SNE byla implementována nad rámec zadání. Výsledná aplikace je dostupná na adrese: `http://extraktor.herokuapp.com`.

Jelikož jsem v návrhu aplikace počítal s budoucím rozšířením, nabízí se logicky obohatit výslednou aplikaci o další analytické metody a další druhy laboratorních zpráv, jejichž data bude možno extrahovat a analyzovat. Dalším škálováním zdrojů ve službě Heroku lze docílit ještě většího výkonu a více zredukovat čas potřebný pro extrakci dat a vykreslení grafů. Posledním možným vylepšením může být perzistentní provedení grafů, aby nedocházelo k opětovnému načítání grafu v rámci jednoho sezení, případně připojení dat z dalších zdrojů. Například databází.

Literatura

[1] Adobe. *Document management - Portable document format* [online]. 2008 [cit. 2021-03-11]. Dostupné z: https://www.adobe.com/content/dam/acom/en/devnet/acrobat/pdfs/PDF32000_2008.pdf.

[2] Ariga, A. *Tabula-py* [online]. 2021 [cit. 2021-04-07]. Dostupné z: https://github.com/chezou/tabula-py.

[3] Arthur, D. *K-Means++: The Advantages of Careful Seeding* [online]. [cit. 2021-04-17]. Dostupné z: http://ilpubs.stanford.edu:8090/778/1/2006-13.pdf.

[4] Atluri, G. *Association Analysis Techniques for Bioinformatics Problems* [online]. Minnesota: [b.n.], 2009 [cit. 2021-05-02]. Dostupné z: https://www-users.cs.umn.edu/~kumar001/dmbio/kumar_bicob.pdf.

[5] CalTech. *Covariance and Principal Component Analysis* [online]. [cit. 2021-04-17]. Dostupné z: http://pmaweb.caltech.edu/~physlab/lab_21_current/Ph21_5_Covariance_PCA.pdf.

[6] Dash. *Dash Python User Guide* [online]. 2021 [cit. 2021-04-07]. Dostupné z: https://dash.plotly.com.

[7] Duhigg, C. *Síla zvyku*. 1. vyd. BizBooks, 2013. ISBN 978-80-265-0055-1.

[8] Heroku. *Heroku Development Center* [online]. 2021 [cit. 2021-04-07]. Dostupné z: https://devcenter.heroku.com.

[9] Herr, N. *Inverse Square Law* [online]. [cit. 2021-03-28]. Dostupné z: http://www.csun.edu/~vceed002/books/sourcebook/chapters/15-geometric-principles/inverse-square.html.

[10] Hinton, G. a Roweis, S. *Stochastic Neighbor Embedding* [online]. [cit. 2021-03-28]. Dostupné z: https://cs.nyu.edu/~roweis/papers/sne_final.pdf.

[11] Johnson, D. *PDF statistics - the universe of electronic documents* [online]. 2018 [cit. 2021-04-24]. Dostupné z: https://www.pdfa.org/wp-content/uploads/2018/06/1330_Johnson.pdf.

[12] Maaten, L. van der. *Visualizing Data using t-SNE* [online]. 2008 [cit. 2021-03-28]. Dostupné z: https://lvdmaaten.github.io/publications/papers/JMLR_2008.pdf.

[13] McInnes, L. *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction* [online]. 2018 [cit. 2021-03-28]. Dostupné z: https://arxiv.org/pdf/1802.03426.pdf.

[14] Pang Ming Tan, V. K. *Introduction to Data Mining*. 2. vyd. Pearson, 2018. ISBN 0133128903.

[15] Radovanović, M. *NN-Descent on High-Dimensional Data* [online]. [cit. 2021-04-07]. Dostupné z: https://perun.pmf.uns.ac.rs/radovanovic/publications/2018-wims-nndes.pdf.

[16] Scikit. *Silhouette score* [online]. [cit. 2021-04-07]. Dostupné z: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html.

[17] scikit. *T-distributed Stochastic Neighbor Embedding* [online]. 2021 [cit. 2021-03-28]. Dostupné z: https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html.

[18] Scipy. *Hierarchical clustering* [online]. [cit. 2021-03-18]. Dostupné z: https://docs.scipy.org/doc/scipy/reference/generated/scipy.cluster.hierarchy.linkage.html.

[19] Verleysen, M. a François, D. *The Curse of Dimensionality in Data Mining and Time Series Prediction* [online]. [cit. 2021-04-07]. Dostupné z: https://www.researchgate.net/profile/Damien-Francois-2/publication/221582081_The_Curse_of_Dimensionality_in_Data_Mining_and_Time_Series_Prediction/links/004635333fa7fbbc71000000/The-Curse-of-Dimensionality-in-Data-Mining-and-Time-Series-Prediction.pdf.