Vysoká škola ekonomická v Praze

Fakulta informatiky a statistiky

Vývoj prediktivního modelu pro doporučování obsahu zákazníkům IPTV společnosti

DIPLOMOVÁ PRÁCE

Studijní program: Aplikovaná informatika

Studijní obor: Informační systémy a technologie

Autor: Bc. Tomáš Polák

Vedoucí diplomové práce: Ing. et Ing. Soňa Karkošková, PhD.

Praha, Květen 2020

Prohlášení

Prohlašuji, že jsem diplomovou práci „Vývoj prediktivního modelu pro doporučování obsahu zákazníkům IPTV společnosti“ vypracoval samostatně za použití v práci uvedených pramenů a literatury.

V Praze dne 4. května 2020 .....

Bc. Tomáš Polák

Poděkování

Rád bych poděkoval Ing. et Ing. Soně Karkoškové za vedení mé diplomové práce, a dále za odborné rady a doporučení, které mi během zpracování mé diplomové práce poskytla. Dále bych také rád poděkoval Ing. Mgr. Janovi Romportlovi Ph.D., Ing. Petrovi Stanislavovi a Ing. Tomášovi Kalendovi, za možnost zpracovávat projekt mé diplomové práce v prostředí společnosti IPTV poskytovatele.

Abstrakt

Služby *“IPTV”* a *“Video on Demand”* jsou v současné době velmi populární, přičemž uživatelé zmíněných služeb oceňují především velký výběr obsahu a snadnou dostupnost služeb. Schopnost prezentovat uživateli relevantní, a pro klienta zajímavý obsah, poskytuje dodavateli zmíněných služeb značnou konkurenční výhodu na současném trhu.

Cílem této diplomové práce je vyvinout prediktivní model, jenž bude vybrané skupině uživatelů IPTV služeb predikovat zajímavý obsah. Zpracování praktické části této diplomové práce bude probíhat ve spolupráci s jedním z největších IPTV poskytovatelů na Českém trhu.

Výstupem diplomové práce je statistický prediktivní model, jenž bude implementován u zmíněné společnosti, a jehož cílem bude vybrané skupině uživatelů predikovat zajímavý obsah.

Klíčová slova

Strojové učení, Prediktivní model, Doc2vec, Rekomendační systémy, IPTV.

Abstract

"IPTV" and *"Video on Demand"* services are very popular in the modern age. The users of these services appreciate a large selection of content and easy accessibility of these services. The ability to present individualized and relevant content to the users, gives the IPTV providers a large competitive advantage on the current market.

The main goal of this diploma thesis is to develop a predictive model, which will suggest an interesting content for a selected group of the IPTV users. Development of the practical part of this thesis will be conducted in collaboration with one of the largest IPTV providers on the Czech market.

The output of this diploma thesis will be a statistical predictive model, which will be implemented at the mentioned service provider and its function will be prediction of interesting content for the targeted user group.

Keywords

Machine learning, Predictive model, Doc2vec, Recommender systems, IPTV.

Obsah

Seznam obrázků

Úvod

Neustálý technologický rozvoj přináší velké změny do různých odvětví, přičemž mezi ně patří i trh s audiovizuálním obsahem, který v posledních letech prochází řadou změn. Velký podíl na tom má zejména digitalizace obsahu, jenž přinesla zejména nové zdroje audiovizuálního obsahu a dále i navýšení kvality přenášeného audiovizuálního obsahu (přinejmenším z technologického pohledu). Klasické televizní vysílání je v současné době nahrazováno za pomoci *„Internet Protocol Television"* (dále jen IPTV) a *„Video on Demand"* (dále pouze VoD) službami.

Služby IPTV poskytují stejně tak jako klasické televizní vysílání přenos mluveného slova a obrazu do koncového zobrazovacího zařízení u zákazníka. Hlavním rozdílem mezi klasickým televizním vysíláním a IPTV službami spočívá ve způsobu přenosu audiovizuálního obsahu. V případě běžného televizního vysílání je obraz a zvuk přenášen za pomoci rádiových vln, satelitního vysílání nebo kabelového připojení. Oproti tomu je obsah IPTV přenášen za pomoci internetového připojení. Mezi další rozdíly a výhody IPTV služeb patří například rozšířené funkce zobrazení informací o přehrávaném obsahu, sledování obsahu s časovým posunem vůči živému vysílání a další. Na českém trhu existuje řada IPTV poskytovatelů, například T-Mobile TV, O2TV a další.

Oproti tomu služby VoD poskytují uživateli distribuci audiovizuálního obsahu, jenž je, stejně tak jako v případě IPTV, přenášen za pomoci internetového připojení. Rozdíl mezi službami IPTV a VoD spočívá zejména v přenášeném obsahu. V případě služeb IPTV se jedná o přenos audiovizuálního obsahu tak jako u klasického televizního vysílání, tedy existují různé televizní kanály a programy, které přenášejí obsah, na nějž se specializují (sport, dokumenty atd.). Oproti tomu v případě služeb VoD si uživatel vybírá obsah na základě svých preferencí, jenž mu bude v reálném čase přenášen do koncového zařízení, a neřídí se televizním programem jednotlivých televizních kanálů. Ve většině případů jsou pomocí služeb VoD přenášeny zejména filmy a seriály. Mezi zahraniční poskytovatele VoD služeb lze zařadit například *Netflix* a *HBO GO*, přičemž mezi českými poskytovateli lze uvést například *O2 Videotéka*.

Stejně tak jako se postupem času mění technologie, mění se i preference cílové skupiny uživatelů zmíněných služeb. Uživatelům již nestačí přístup k základnímu obsahu klasických televizních programů. Právě naopak, vyžadují velké množství rozlišného obsahu, který bude pro daného uživatele relevantní na základě jeho osobních preferencí, a bude okamžitě dostupný ke shlédnutí na jejich koncovém zobrazovacím zařízení. Průzkum provedený v USA, ukazuje že VoD služby se těší velké popularitě zejména mezi mladšími generacemi uživatelů (1). Oproti tomu klasické televizní vysílání (v tomto ohledu sem lze zařadit i přenos televizního vysílání pomocí služeb IPTV) má své příznivce zejména u starších generací uživatelů.

U obou zmíněných služeb hraje velkou roli obsah, který svým uživatelům poskytují. Proto se poskytovatelé těchto služeb soustředí na to, aby nabídli uživatelům co největší výběr různorodého obsahu. V mnoha případech ovšem dochází k tomu, že koncový uživatel je

zahlcen stovkami jemu prezentovaných možností ve formě filmů, a seriálů a má problém si vybrat (2).

Z tohoto důvodu jsou pro poskytovatele zmíněných služeb velmi přínosné takzvané *„Recommender systems“* (do českého jazyka lze přeložit jako „Rekomendační systémy“, případně „Doporučující systémy“). Rekomendační systémy se zaměřují na predikci obsahu, který bude pro uživatele zajímavý, na základě dat, jenž mají o uživateli k dispozici z minulosti. Součástí rekomendačního systému je i takzvaný *„Recommendation Algorithm“* (do češtiny lze přeložit jako „Rekomendační algoritmus“), který lze chápat jako prediktivní model, jenž obsahuje sadu matematických instrukcí, na jejichž základě jsou vytvářeny predikce pro daný rekomendační systém.

S méně či více sofistikovanými rekomendačními systémy se většina lidí setkává dennodenně, aniž by si toho sami byly vědomi. Mezi běžná využití rekomendačních systémů patří například:

- Rekomendační systémy v *e-commerce*
    - Doporučování položek v e-obchodech (např. *Amazon*, *Alza* atd.)
    - Doporučování položek při pronájmu realit (např. *Booking.com*, *Airbnb* atd.)
- Reklamní rekomendační systémy
    - Výběr vhodného reklamního obsahu pro daného zákazníka
- Hudební rekomendační systémy
    - Návrh hudby, která by se uživateli služby mohla líbit na základě jeho preferencí (např. *Spotify*, *Apple Music* atd.)
- Rekomendační systémy pro doporučování video obsahu
    - Návrh filmů a seriálů, které by se uživateli mohli líbit na základě jeho preferencí (např. *Netflix*, *HBO GO* atd.)

Na rekomendační systémy pro doporučování video obsahu je zaměřena i tato diplomová práce. Primárním cílem této diplomové je vytvořit prediktivní model, který bude pro vybranou skupinu uživatelů IPTV služeb predikovat zajímavý obsah. Při zpracování praktické části této diplomové práce, byla navázána spolupráce s jedním z největších IPTV poskytovatelů na Českém trhu. V případě že model bude dosahovat uspokojivých výsledků, bude vyvíjený model následně implementován do provozu u dané společnosti.

Cíle práce

Hlavním cílem této diplomové práce je vytvořit prediktivní model, jenž vybrané skupině uživatelů IPTV služeb (skupina specifikována ve čtvrté subkapitole této kapitoly) predikuje zajímavý obsah, na základě jejich předešlých preferencí. Tento hlavní cíl lze rozdělit do tří menších částí. První částí bude analýza a transformace dat z původních datových zdrojů, přičemž data (v anonymizované formě) poskytne již zmíněný poskytovatel IPTV služeb. Druhou částí bude převod dat z relační databázové formy do numerické reprezentace pomocí vektorů, jenž bude následně využita při trénování vlastního modelu. Třetí částí bude vytvoření vlastního modelu, na základě dat získaných ve druhé části.

Sekundárním cílem této diplomové práce je teoretická charakteristika vybraných algoritmů, postupů a technologií, na úrovni, jenž bude mít čtenář této práce zapotřebí, k pochopení vlastního textu této diplomové práce.

Způsob dosažení cíle práce

Cíle této diplomové práce lze rozdělit do dvou rovin: teoretické a praktické. K dosažení stanovených cílů budou aplikovány vědecké metody při psaní práce. Teoretické cíle této diplomové práce, mezi něž patří charakteristika vybraných algoritmů, postupů a technologií, budou dosaženy rešerší dostupných zdrojů o dané problematice. Mezi tyto zdroje patří například odborné publikace a články, dostupné na specializovaných webových portálech. Bližší informace o zdrojích využitých při zpracování této diplomové práce lze nalézt v kapitole č. 1.

Mezi cíle praktické části této práce paří analýza a transformace dat, převod dat z relační databázové formy do numerické reprezentace a následně vytvoření prediktivního modelu. V první části budou zpracována data z dostupných datových zdrojů do požadované podoby, následně budou data pomocí vybraného algoritmu převedena do numerické reprezentace a na závěr bude vytvořen prediktivní model, jehož cílem bude vybrané skupině uživatelů predikovat zajímavý obsah.

Struktura práce

Jak již bylo zmíněno v druhé subkapitole, tuto diplomovou práci lze rozdělit do teoretické a praktické roviny. Teoretická rovina seznamuje čtenáře s vybranými koncepty, jejichž pochopení bude zásadní pro porozumění vlastního textu práce. Praktická rovina popisuje postup analýzy a transformace zdrojových dat, převod dat do numerické formy a vytváření prediktivního modelu.

Stejně tak, jako je tomu i u jiných diplomových prací, slouží úvodní kapitola k seznámení čtenáře s obsahem práce, přičemž jsou v ní stanoveny cíle, struktura, předpoklady, omezení a výstupy této diplomové práce. Následující čtyři kapitoly této práce lze zařadit do teoretické roviny. První kapitola poskytuje rešerši dostupných zdrojů, zabývajících se danou problematikou. Druhá kapitola je zaměřena na základní popis vybraných technologií, algoritmů a konceptů datového dolování a strojového učení, jenž souvisí s tématem této práce. Třetí kapitola poskytuje základní charakteristiku metodiky CRISP-DM, dle níž byla zpracována praktická část této diplomové práce. Čtvrtá kapitola je zaměřena na popis vývojového prostředí, ve kterém bude zpracovávána praktická část této práce.

Do praktické roviny lze zařadit pátou až osmou kapitolu. Pátá kapitola se zabývá analýzou dostupných datových zdrojů a jejich následnou transformací do požadované podoby. Šestá kapitola je zaměřena na převod dat z relační databázové formy do numerické (vektorové) formy. Sedmá kapitola je zaměřena na vytvoření prediktivního modelu, jehož cílem je vybraným zákazníkům IPTV poskytovatele predikovat zajímavý obsah. Osmá kapitola se soustředí na evaluaci vytvořeného prediktivního modelu.

Poslední kapitolou vlastního textu práce je závěrečná kapitola, poskytující shrnutí této diplomové práce. Kromě zmíněných kapitol obsahuje diplomová práce řadu příloh, jenž obsahují zdrojový kód vybraných součástí praktické části diplomové práce.

Předpoklady a omezení práce

Hlavním předpokladem pro korektní zpracování této diplomové práce je dostatek dat o uživatelských relacích při využívání služeb IPTV v požadované kvalitě. Data by měla obsahovat co nejméně prázdných polí a měla by být dostatečně přesná, aby poskytla přiměřenou vypovídací hodnotu pro finální model (Samotné požadavky na kvalitu dat a datového vzorku jsou upřesněny v kapitole 5.2.1). Jak již bylo zmíněno v předchozích kapitolách, data použitá pro praktickou část této diplomové práce poskytne vybraná IPTV společnost.

Jedním z omezení této diplomové práce bude její zaměření na konkrétní skupinu uživatelů. Aby bylo dosaženo dostatečné kvality dat, jenž byla zmíněna v přechozím paragrafu, bude se tato diplomová práce soustředit pouze na uživatele IPTV služeb využívající konkrétní technologické zařízení pro příjem IPTV (konkrétně se jedná o set top box EKT). IPTV poskytovatel poskytuje i možnosti příjmu IPTV pomocí dalších zařízení (*Xbox* konzole, Webový prohlížeč, jiné druhy set top boxů atd.), ovšemže data o využívání těchto zařízení nejsou u IPTV poskytovatele ukládána v dostatečné kvalitě. Detailnější popis tohoto omezení je uveden v kapitolách 2.1.3 a 5.2.1.

Společnost, jenž poskytuje data pro praktickou část této diplomové práce působí mimo jiné i jako telekomunikační operátor. Z důvodů mnohých regulací a omezení (např. ve formě GDPR), definujících, jak může daná společnost nakládat s citlivými daty klientů, nebude možné zveřejnit data, která byla využita při zpracování této diplomové práce. Ze stejných důvodů nebude tato diplomová práce obsahovat detaily implementace vytvářeného prediktivního modelu a konkrétní informace o jeho využití v provozních systémech.

Výstupy práce a očekávané přínosy

Hlavní výstup této diplomové práce vyplývá z primárního cíle, přičemž se jedná o statistický prediktivní model. Tento model bude vybraným uživatelům IPTV služeb predikovat zajímavý obsah, na základě jejich preferencí z minulosti. Samotný model bude následně implementován marketingovým a technologickým oddělením společnosti do rekomendačního systému, a jeho výsledky budou zobrazeny vybrané skupině koncových uživatelů.

Poskytovatel IPTV služeb od implementace rekomendačního systému (jehož součástí bude model vytvořený v rámci této práce) očekává větší spokojenost uživatelů, rozšířené možnosti prodeje prémiového obsahu a snížení odchodovosti uživatelů od služeb IPTV daného poskytovatele.

1 Rešerše zdrojů

Knižní publikace

Knižní publikace *„Mastering Python for data science: explore the world of data science through Python and learn how to make sense of data“* (3). Tuto knižní publikaci jsem využil zejména při psaní teoretické části této diplomové práce. Kniha poskytuje mimo jiné i základní popis celého procesu zpracování dat a základní metody a nástroje z oblastí *Data Science* (dále jen DS) a *Machine Learning* (dále jen ML, do českého jazyka lze přeložit jako strojové učení). Kniha dále poskytuje i seznámení se základními balíčky využívanými při DS a ML, v programovacím jazyku *Python* (např. *NumPy*).

Knižní publikace *„Python Data Science Handbook”* (4). Tato kniha poskytuje pokročilé informace o oblastech DS a ML, přičemž je zaměřena na programovací jazyk Python a balíčky funkcí souvisejících se zpracováním, transformací a vizualizací dat při využití známých konceptů DS a ML. Mezi zmíněné Python balíčky patří například *NumPy*, *Pandas*, *Matplotlib* a další. Vybrané koncepty a funkce z knihy byly využity při psaní praktické části této diplomové práce.

Knižní publikace *„Python for data analysis: data wrangling with Pandas, NumPy, and IPython“* (5). Stejně tak jako předchozí zmíněná knižní publikace, je tato kniha zaměřena na oblast DS a ML při využití programovacího jazyka Python. Rozdíl mezi oběma knihami spočívá v tom, že tato kniha do detailu popisuje oblast datových struktur, časových řad a jiné oblasti konceptů DS a ML. Knihu jsem využil při psaní teoretické části této diplomové práce.

Knižní publikace „*Recommender Systems An Introduction*“ (6), je zahraniční publikace zabývající se tématem rekomendačních systémů. Publikace se zaobírá teoretickou koncepcí rekomendačních systémů, přičemž uvádí i příklady implementací rekomendačních systémů. Tuto knihu jsem využil při psaní kapitoly 2.4 o rekomendačních systémech.

Knižní publikace „*Dobývání znalostí z databází*“ (7). Tato knižní publikace je zaměřena, jak již její název napovídá, na získávání informací z datových zdrojů a jejich interpretaci tak, aby byly získány relevantní znalosti. Tuto knihu jsem využil při psaní kapitoly č. 3 o metodice CRISP-DM. Kniha dále popisuje různé algoritmy strojového učení a další informace o přípravě a zpracování dat v různých nástrojích k tomu určených. Ovšemže informace k těmto tématům jsem primárně čerpal z aktuálnějších zdrojů, jelikož kniha samotná pochází z roku 2003 a oblast datové analýzy a strojového učení se neustále vyvíjí.

Odborné články

Odborný článek *„Distributed Representation-based Recommender Systems in E-commerce“* (8). Článek je zaměřený na vybrané druhy přístupů k rekomendačním systémům v rámci e-commerce, přičemž porovnává přístupy *item-to-item* a *user-to-user* a jejich využití v rámci *Word2vec* a *Doc2vec* modelů. Na závěr je vytvořený experimentální model na základě *user-to-item* rekomendačního systému. Článek je pro tuto diplomovou práci přínosný v alternativním pohledu na implementaci rekomendačních systémů.

Odborný článek *„Distributed Representations of Words and Phrases and their Compositionality“* (9). Známý článek v rámci kterého byl původně představen koncept numerické reprezentace slov pomocí vektorů zvaný Word2vec. Článek byl využit zejména v kapitole 2.3.2 při psaní charakteristiky technologie Word2vec.

Odborný článek *„Distributed Representations of Sentences and Documents“* (10). Článek poskytuje rozšíření původního článku (9) který prezentoval koncept Word2vec, rozšiřuje původní koncept na zpracování celých dokumentů zvaný Doc2vec. Základním rozdílem mezi danými koncepty je, že Word2vec se soustředí na numerickou reprezentaci slov a celých vět, přičemž Doc2vec je zaměřeno na numerickou reprezentaci celých odstavců a dokumentů. Článek byl využit při psaní teoretické části této diplomové práce v kapitole 2.3.3.

Technologické webové servery a dokumentace nástrojů

Oblast datové analytiky a strojového učení se neustále mění. Mění se trendy, technologie i know-how z této problematiky. Knižní publikace (zejména ty překládané do českého jazyka), neposkytují vždy nejaktuálnější informace o těchto tématech. Z tohoto důvodu bylo zapotřebí při zpracování teoretické i praktické části této diplomové práce využít i podklady a informace z různých odborných i méně odborných technologických webových serverů, jenž velmi často poskytují aktuálnější informace než knižní publikace.

Primárním webovým serverem, jenž byl využit při psaní této diplomové práce je server „*Towards Data Science*“ (11). Články na tomto serveru komplexně pokrývají oblast datové analytiky a strojového učení z mnoha různých pohledů. Velmi často jsou na tomto serveru rozebírána aktuální témata a implementace konkrétních algoritmů či řešení vybraných úloh.

Dalším podkladem, využitým zejména při psaní praktické části této diplomové práce, jsou webové servery, jenž obsahují dokumentaci jednotlivých využitých technologií. Mimo jiné se jedná o následující technologie a servery, obsahující jejich dokumentaci:

- Apache Spark (ver. 2.4.5) – (12)
- Gensim (13)
- Microsoft Azure (14)
- PySpark (15)
- Python (16)

2 Charakteristika technologií

Tato kapitola je zaměřena na teoretický popis a charakteristiku technologií a konceptů, jenž byly využity v následujících kapitolách této diplomové práce, které popisují praktickou část této diplomové práce. Mezi hlavní zdroje využité při psaní této kapitoly patří knižní publikace a odborné články uvedené v první kapitole této práce.

Řada témat uvedených v následující kapitole byla již v minulosti detailně popsána a definována v různých knižních publikacích a odborných článcích (např. *IPTV*, *Data Mining*, *Python*, *Spark* a další). Cílem této diplomové práce není rekapitulace již zmíněných technologií a konceptů, tato kapitola pouze poskytuje jejich zjednodušený popis, který bude čtenář této práce potřebovat pro pochopení obsahu následujících kapitol.

2.1 IPTV a VoD

2.1.1 Charakteristika IPTV a VoD

Služba IPTV neboli *„Internet Protocol Television"* (do českého jazyka lze volně přeložit jako „televize přes internetový protokol"), funguje jako alternativa ke klasickému televiznímu vysílání šířeném přes tradiční kabelové připojení, satelitové připojení nebo zemské digitální vysílání. Pojem IPTV byl v průběhu času definován rozlišně, zejména díky postupným změnám technologií pro přenos IPTV.

Oficiální definici IPTV od Mezinárodní telekomunikační unie (v anglickém originále *„International Telecommunication Union"*) lze do českého jazyka přeložit následovně (17):

„IPTV lze definovat jako multimediální služby, například televize / video / audio / text / grafika / data, které jsou přenášeny pomocí sítí založených na internetovém protokolu, řízených tak, aby poskytli požadovanou úroveň kvality služeb, zážitku, bezpečnosti, interaktivnosti a spolehlivosti."

Detailnější definici od Aliance pro telekomunikační průmyslová řešení (v anglickém originále *„Alliance for Telecommunications Industry Solutions"* lze do českého jazyk přeložit následovně (17):

„IPTV je definováno jako bezpečný a spolehlivý způsob přenosu mediálních a dalších služeb koncovým uživatelům. Mezi tyto služby patří například přenos živého televizního vysílání, videa na vyžádání (VoD) a interaktivního televizního vysílání (iTV). Tyto služby jsou přenášeny přes zabezpečenou síť využívající síťové pakety IP protokolu k přenosu audia, videa a kontrolu signálu. V kontrastu s přenosem videa přes nezabezpečené veřejné internetové sítě, zabezpečení a výkon IPTV služeb je úzce řízený tak, aby byl zákazníkovi poskytnut co možná nejlepší prožitek služby, přičemž ve výsledku vzniká zajímavé obchodní prostředí pro poskytovatele obsahu, inzerenty a uživatele služeb."

Pro potřeby této diplomové práce stačí IPTV vymezit jako službu, jenž poskytuje televizní vysílání a další formy audiovizuálního obsahu pomocí TCP/IP protokolu, namísto tradičních forem televizního vysílání. Ve většině případů je IPTV služba distribuována pomocí dodavatele služby a skládá se z živého televizního vysílání a video obsahu na vyžádání (VoD) (18).

Praktická část této diplomová práce je zpracována ve spolupráci s jedním z největších IPTV poskytovatelů v České republice, přičemž zmíněná společnost rovněž poskytuje svým klientům přenos živého televizního vysílání a zpoplatněné promítání komerčních filmů a seriálů z takzvané videotéky, jenž funguje na principu *Video on Demand* (VoD).

Definici služby *„Video on Demand“ (*VoD) od webového serveru *Techopedia.com* lze do českého jazyka přeložit následovně (19):

„Video on Demand (Vod) je systém, který umožnuje zhlédnutí uživatelem vybraného video obsahu na koncovém zařízení (televize nebo osobní počítač). Video na vyžádání je jednou z dynamických funkcí poskytovaných pomocí IPTV. VoD poskytuje uživateli menu s výběrem dostupného video obsahu, ze kterého si uživatel může vybrat. Video data jsou přenášena přes Real-Time Streaming protokol.“

VoD umožnuje uživatelům okamžitý přístup k video obsahu na jejich koncových zařízeních, přičemž je poskytován rozmanitý obsah (např. sportovní záznamy, dokumentární obsah, filmy a seriály). Hlavním rozdíly mezi klasickým televizním vysíláním (včetně pohledu přenosu živého vysílání pomocí IPTV) a VoD jsou uvedeny níže.

- Televizní vysílání – Využívá *broadcast* technologie neboli distribuci audiovizuálního obsahu širokému spektru obecenstva.
- VoD – Využívá *unicast* přenos dat, přičemž obsah je distribuován pouze konkrétnímu uživateli.

Alternativním příkladem využití služeb VoD patří i videokonferenční hovory. Služby VoD se těší velké popularitě, nicméně je nutno podotknout, že jsou i v současné době v některých částech světa omezovány šířkou pásma internetových sítí (19). Samotné služby VoD určené pro přenos komerčního audiovizuálního obsahu lze rozdělit do tří kategorií (20):

1. *Subscription video on demand* (SVOD) – Uživatel tohoto druhu služby může volně zobrazit veškerý dostupný obsah služby bez omezení za paušální měsíční poplatek. Mezi poskytovatele této služby patří např. *Netflix*, *HBO GO* a jiné.
2. *Transactional video on demand* (TVOD) – V tomto případě platí uživatel služby poplatek na základě zobrazení daného obsahu, čili takzvaně *„pay-per-view*“. Dle druhu služby si uživatel může zaplatit pronájem vybraného obsahu na určitou dobu, nebo si daný obsah koupí a poté je mu volně přístupný bez časového omezení. Jako příklad zmíněných služeb na českém trhu lze uvést např. *O2videotéku*.
3. *Advertising-based video on demand* (AVOD) – Tento druh služeb je pro uživatele výjimečný v tom, že za poskytované služby nemusí platit žádný poplatek. Poskytovatel služby nabízí uživatelům audiovizuální obsah za shlédnutí reklamních sdělení. Zde lze uvést jako příklad video portál *YouTube*.

Dříve zmíněný poskytovatel IPTV služeb, nabízí svým klientům služby VoD pomocí modelu *Transactional video on Demand* (TVOD). Uživatel služby tedy platí za pronájem/koupi daného audiovizuálního obsahu.

2.1.2 Technologie IPTV a VoD

Cílem této subkapitoly není poskytnou čtenáři detailní informace o tom, jak fungují služby IPTV a VoD z technologického hlediska. Jedná se pouze o základní přehled internetových protokolů využívaných pro vlastní přenos dat a vlastního zapojení zmíněných služeb u koncového zákazníka.

Služby IPTV využívají ve většině případů IP multicast, přičemž datové pakety jsou z jednoho zdroje přeposílány na více koncových stanic. Nejčastěji využívané internetové protokoly pro IPTV služby jsou zmíněny níže:

- *Internet Group Management Protocol* (IGMP) – IGMP je internetový protokol, jenž umožnuje koncovému zařízení v internetové síti, přihlásit se do skupiny zařízení v multicast skupině. Tento protokol je využíván zejména pro přenos živého televizního vysílání koncovým uživatelům (21).
- *Real-Time Streaming Protocol* (RTSP) – RTSP je protokol zaměřený na přenos dat mezi koncovým zařízením a serverem poskytujícím data v reálném čase, přičemž je využíván zejména při poskytování VoD obsahu (22).

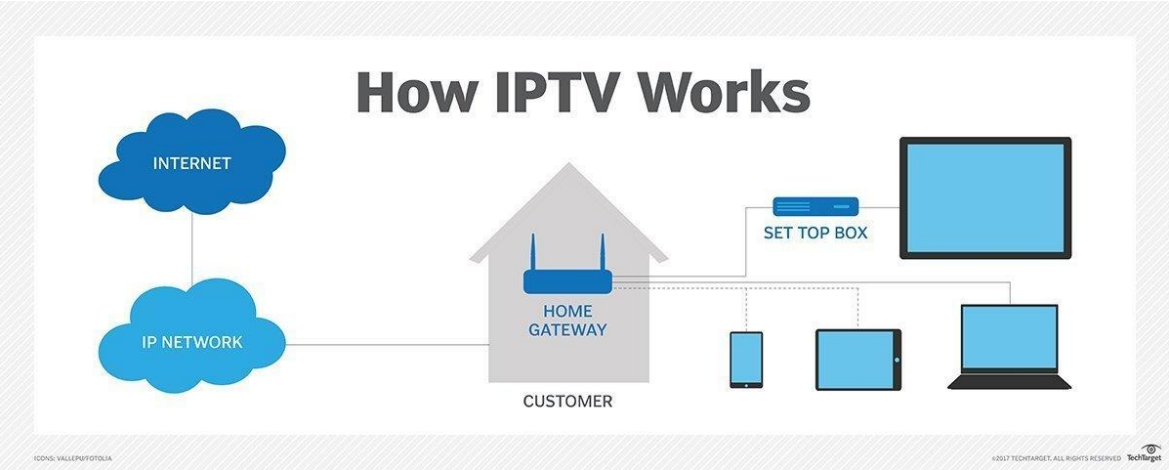
Mezi další využívané protokoly patří *Real-Time Messaging Protocol* (RTMP) a *Hyper Text Transfer Protocol* (HTTP) (18).

Samotné zapojení služeb IPTV a připojení ke koncovým zařízením není složité, ale i přesto jej u většiny poskytovatelů IPTV služeb provádí technik vyškolený od IPTV poskytovatele. Uživatel zmíněných služeb musí mít internetové připojení (ve většině případů od stejného poskytovatele jako od služeb IPTV a VoD), aby mu dané služby korektně fungovaly. Toto připojení je zprostředkováno pomocí internetového směrovače v obydlí uživatele. Následující fáze je ovlivněna tím, jestli uživatel využívá služby IPTV a VoD na osobním počítači, nebo pro zobrazení obsahu využívá alternativní zařízení (ve většině případů set top box, ale ke službám IPTV a VoD lze přistupovat i například z různých herních konzolí – *Microsoft Xbox*, *Sony PlayStation* a jiných zařízení).

V případě, kdy uživatel využívá služby IPTV na osobním počítači, stačí aby se pouze přihlásil přes webový portál poskytovatele zmíněných služeb a nemusí podnikat žádné další kroky. (Pozn. v takovém případě není zapotřebí aby byl uživatel připojen do internetové sítě od stejného poskytovatele služeb jako u služeb IPTV. Přístup ke službám IPTV bude fungovat i od jiného internetového poskytovatele).

Pokud uživatel využívá alternativní zařízení pro zobrazování obsahu služeb IPTV na své televizi (ve většině případů se bude jednat o set top box), musí zařízení nejprve propojit s internetovým směrovačem a televizním přijímačem. Zařízení pak bude mimo jiné sloužit pro dekódování datových paketů a zobrazení na televizní obrazovce uživatele.

Příklad schématu zapojení obou scénářů lze nalézt na obrázku č. 1.



Obrázek 1 Schéma zapojení IPTV služby, zdroj: (18)

Typy IPTV služeb

Jak již bylo zmíněno v předchozích subkapitolách, služba IPTV se skládá ze dvou hlavních částí – IPTV přenosu televizního vysílání a VoD přenosu audiovizuálního obsahu na vyžádání uživatele. Samotný IPTV přenos televizního vysílání lze rozdělit do dvou částí:

- *Live TV* – Živé televizní vysílání – V případě živého televizního vysílání se jedná o příjem televizního vysílání, vysílaného v reálném čase, tak jak je tomu v případě klasických metod příjmu televizního vysílání.
- *Time-Shifted TV* – Televizní vysílání s časovým posunem – Jednou z předností IPTV služeb je i možnost koncového uživatele služby sledovat zpětně s vybraným časovým posunem. V případě, kdy uživatel sleduje televizní vysílání s časovým posunem, je uživateli zobrazován obsah z takzvaného archivu, přičemž princip sledování je podobný jako u služby VoD. Většina poskytovatelů tímto způsobem umožnuje shlédnout až 14 dní starý obsah.

Z pohledu praktické části této diplomové práce, není důležité jaký typ sledování IPTV uživatel využívá. Data o sledování jsou pro jednotlivé druhy IPTV vysílání ukládaná do stejných tabule, s jiným parametrem typu přenosu.

2.1.3 Charakteristika EPG

Pojem EPG (v anglickém originále „*Electronic Programming Guide*“) je dle webového serveru *Techtarget.com* definován následně (23):

Elektrický programový průvodce (EPG) je aplikace využívána v set-top boxech a moderními televizemi pro zobrazení seznamu aktuálního a plánovaného obsahu, který je nebo bude dostupný na jednotlivých televizních kanálech, včetně krátkého shrnutí a popisků každého pořadu. EPG je ekvivalentem tištěného televizního programového průvodce.

EPG tedy poskytuje informace o jednotlivých pořadech jenž jsou na jednotlivých televizních kanálech vysílány, jak v současné chvíli, tak plánované do budoucnosti. O tom na jak dlouhé

období do budoucnosti jsou koncovému uživateli prezentovány informace rozhoduje poskytovatel TV či IPTV dle svých technologických možností. V případě IPTV poskytovatele, s jehož spolupráci byla zpracovávána praktická část této diplomové práce, jsou koncovému uživateli dostupná data na deset dní dopředu.

Kromě zobrazení aktuálního a plánovaného televizního programu poskytuje EPG i další rozšířené možnosti správy a kontroly televizního zařízení. Ve většině případu je EPG koncovému uživateli zobrazeno ve formě kontextového menu, kde si uživatel vybírá své preference (23). Dále může v tomto menu využít další funkce, např. dětský zámek, vyhledávání televizních programů na základě tématu/obsahu, či objednání privátního VoD obsahu.

Z pohledu praktické části této diplomové práce jsou EPG data obzvláště důležité, jelikož na jejich základě bude možné klasifikovat obsah, na nějž se uživatelé dívali a přiřadit mu doplňující parametry (např. žánr/téma pořadu, hodnocení ze serveru zabývající se filmy/seriály – webový portál IMDB atd.). Bez zmíněných doplňujících parametrů by rekomendační model neměl potřebný základ pro vytváření predikcí.

2.1.4 Technologická omezení

Jak již bylo zmíněno v úvodní kapitole, pro praktickou část této diplomové práce bylo nutné nastavit určitá omezení, jejich detailní popis lze nalézt v 5.2.1. V tuto chvíli je zapotřebí poznamenat, že poskytovatel služeb IPTV svým klientům nabízí různé možnosti přístupu ke svým službám (např. různé druhy set top boxů, herních konzolí, webových prohlížečů atd.). Mezi jednotlivými druhy přístupů existují rozdíly v datech, zaznamenávajících uživatelskou aktivitu. Například data o připojení ke službám IPTV pomocí webových prohlížečů jsou mnohem méně kvalitní než data získávaná ze set top boxů.

Z tohoto důvodu je datový vzorek omezen pouze na data pocházející z vybraného druhu set top boxu (konkrétně set top boxu zvaného EKT). Kvalita těchto dat dosahuje požadované úrovně pro další zpracování v praktické části této diplomové práce.

2.2 Data Mining

Cílem této subkapitoly není popis celého komplexního odvětví *Data Mining*, nicméně poskytnout základní seznámení čtenáři této diplomové práce o oblasti dolování dat a vymezit vybrané pojmy (např. algoritmus, prediktivní model a jiné), nutné pro pochopení obsahu následujících kapitol.

2.2.1 Charakteristika

Definici pojmu *„Data Mining“ neboli* proces „dolování dat“ již v minulosti definovalo mnoho různých subjektů. Komplexní definici poskytuje například webová encyklopedie *Britannica,* a lze ji do českého jazyka přeložit následovně (24):

„Dolování dat, také nazýváno získávání znalostí z databází, je v počítačových vědách proces nacházení zajímavých a užitečných vzorců a vztahů ve velkých objemech dat. Tato disciplína kombinuje znalosti z oblasti statistiky a umělé inteligence (jako jsou například neuronové sítě a strojové učení) s funkcemi ze správy databází za účelem analýzy velké digitální sbírky dat, zvaných data set. Dolování dat má rozsáhlá využití v business sféře (pojišťovnictví, bankovnictví, druhotném prodeji), vědeckém výzkumu (astronomie, medicína) a státní bezpečnosti (detekce kriminálních a teroristických aktivit).“

Zjednodušenou verzi výše uvedené definice poskytuje webový server *Techtarget.com*, přičemž do českého jazyka ji lze přeložit následovně (25):

„Proces dolování dat spočívá v třídění velkého množství dat za účelem identifikace vzorců a navázání vztahů vedoucí k řešení problémů skrze datovou analytiku. Nástroje dolování dat umožnují podnikům predikovat budoucí trendy.“

Dolování dat je v současné době velmi populární odvětví datové analytiky, na jehož základech jsou postaveny některé principy strojového učení a umělé inteligence. Je využíváno v mnoha různých výzkumných oblastech, přičemž mezi ně patří i matematika, kybernetika, genetika a marketing (25). Mnoho podniků interně využívá metody dolování dat k predikci chování zákazníků a zvýšení efektivnosti, což jim na současném kompetitivním trhu poskytuje konkurenční výhodu.

2.2.2 Proces dolování dat dle metodiky CRISP-DM

Vlastní kroky procesu dolování dat lze definovat různorodě, v závislosti na typu projektu, používaných nástrojích pro dolování dat, použité metodice a dalších parametrech. Pro potřeby této diplomové práce ovšem využijeme proces implementace dolování dat pomocí metodiky *Cross-Industry Standard Process for Data Mining* (dále jen CRISP-DM), jejíž vybrané části budou rovněž využity v praktické části této diplomové práce. Více informací naleznete v kapitole 3.2.

Metodika CRISP-DM dělí proces implementace dolování dat do šesti hlavních fází, které jsou popsány v tabulce č. 1.

Tabulka 1 Fáze metodiky CRISP-DM, zdroj: (26)

Název fáze	Popis fáze
Business Understanding	Cílem této fáze je porozumění businessu, tedy porozumění obchodním a klientovým cílům, kterých má proces dolování dat dosáhnout. Na základě obchodních cílů a dostupných prostředků jsou stanoveny cíle procesu dolování dat.

Data Understanding	Cílem fáze porozumění datům je shromáždění dat ze zdrojových systémů, na nichž je prováděn takzvaný *Sanity check*, jehož cílem je rozhodnout, zdali je možné s dostupnými daty dosáhnout požadovaného cíle z fáze *Business Understanding*.
Data Preparation	V této fázi dochází k takzvané přípravě dat, tudíž data získaná ve fázi pochopení dat jsou transformována a připravena k produkcionalizaci (je na nich možné natrénovat model). Probíhá zde mimo jiné i čištění dat, agregace dat, třídění dat, anonymizace dat a další.)
Modelling	Poté co jsou data v předchozí fázi připravena k produkcionalizaci, je na jejich základě vytvořen matematický / statistický model, který definuje vzorce. Poté je model otestován na kontrolním vzorku, aby byla potvrzena pravost a korektnost zjištěných výstupů.
Evaluation	V rámci fáze vyhodnocení dochází k porovnání vzorů identifikovaných v procesu dolování dat vůči cílům stanovených ve fázi porozumění businessu. Je rozhodnuto o případné implementaci modelu do produkčního prostředí.
Deployment	Pokud byl model v předchozí fázi uznán za vhodný je implementován do produkčního prostředí, a jeho výstupy jsou implementovány do každodenních vědeckých či podnikových operací.

Obrázek č. 2 zobrazuje optimální průchod procesem implementace dolování dat.

Obrázek 2 Optimální proces dolování dat dle CRISP-DM, zdroj: (26)

2.2.3 Algoritmy a Modely

Nejprve je nutné definovat rozdíl mezi algoritmy a modely dolování dat. Definici pojmu „algoritmus“ dle webového serveru *Techtarget.com* lze do českého jazyka přeložit následovně (27):

„Algoritmus je procedura nebo vzorec který řeší vybraný problém, na základě provádění sledu specifikovaných akcí. Počítačový program lze považovat za pokročilý algoritmus. V matematice a počítačových vědách je algoritmus ve většině případů krátká sada instrukcí, která řeší opakujíc se problém.“

Zjednodušeně řečeno algoritmus je sada pravidel, jejichž zpracováním řešíme vybraný problém. Algoritmy jsou v oblasti informačních technologií a počítačových věd využívány k mnohým účelům. Patří mezi i ně i například algoritmy pro vyhledavače obsahu, šifrovací algoritmy a další, přičemž pro potřeby této diplomové práce jsou podstatné algoritmy pro strojové učení a dolování dat.

Pojem „model" (v kontextu oblasti dolování dat a strojového učení) byl různými subjekty definován různě, přičemž definici společnosti *Microsoft* lze do českého jazyka přeložit následovně (28):

„Model pro dolování dat je vytvořen aplikací algoritmu na data, přičemž se jedná o více než algoritmus nebo kontejner s metadaty: Je to sada dat, statistik a vzorů, které mohou být aplikovány na nová data pro generování předpovědí a vyvozování závěrů o vztazích v datech. "

V oblasti dolování dat a strojového učení slouží algoritmus k natrénování modelu na datech a vytvoření vlastního modelu. Model lze tedy považovat za vzorec výpočtu formovaný na základě výsledku algoritmu, který dostane vybraná data na vstupu a produkuje vybrané hodnoty na výstupu (29).

Dělení algoritmů pro dolování dat

Jedním ze základních dělení algoritmů strojového učení je dělení dle způsobu učení. Níže jsou uvedeny dvě základní kategorie:

- *Supervised learning* – Algoritmy pro učení s učitelem jsou využívány s daty, která jsou dobře označená a kde jsou definovány vstupy i jejich příslušné výstupy. Na základě studia trénovacích dat je vygenerována funkce, která predikuje zařazení nových instancí. Cílem algoritmu tedy je naučit se z poskytnutých vstupů jak zreprodukovat výstupy na nových datech (3, str. 108).
- *Unsupervised learning* – Algoritmy pro učení bez učitele jsou využívány s daty, která nejsou přesně definována a není definována cílová proměnná. Algoritmus hledá v datech nové vzorce a skryté struktury v neoznačených datech (3, str. 108 - 109).

Mezi rozšířené kategorie strojového učení dále patří:

- *Semi-supervised learning* – Algoritmy kombinace strojového učení s učitelem i bez učitele jsou využívány s daty, kdy trénovací vzorek dat obsahuje jak označená tak neoznačená data. Tyto algoritmy jsou využívány zejména v případech, kdy extrakce relevantních příkladů z dat je složitý proces a ve většině případů je musí označovat odborník na danou problematiku, což je časově náročné (30).
- *Reinforcement learning* – Algoritmy zpětnovazebného učení (dále také učení posilováním) se od ostatních druhů algoritmů strojového učení liší zejména v přístupu a práci s daty. Data jsou na vstupu vložena jako stimulus k modelu z prostředí, na něž musí model strojového učení reagovat. Zpětná vazba není získávána ve formě procesu učení jako je tomu u algoritmů učení s učitelem,

nicméně samo prostředí poskytuje tresty a odměny (3, str. 110). Tyto algoritmy jsou využívány zejména v robotice a v rozsahu této diplomové práce nejsou potřebné.

Techniky dolování dat

Základní techniky dolování dat jsou uvedeny v tabulce č. 2.

Tabulka 2 Základní techniky dolování dat, zdroj: (26)

Anglický název	Popis techniky dolování dat
Classification	Technika klasifikace, jak již sám název napovídá, identifikuje, do které kategorie spadá nová instance pozorování. Samotné rozdělení do kategorií probíhá na základě pozorování z trénovací množiny dat, kde již známé příslušnost do kategorie. Tato technika patří mezi algoritmy učení s učitelem.
Regression	Regresní analýza slouží v datovém modelování k identifikaci a analýze vztahů mezi proměnnými, přičemž je identifikována pravděpodobnost, že při výskytu specifické proměnné, v okolí ostatních proměnných. Tato technika patří mezi algoritmy učení s učitelem.
Clustering	Shlukování, rovněž poskytuje náznak principu techniky v názvu. Tato technika je založena na hledání instancí, jež jsou si vzájemně podobné, přičemž samotný proces poskytuje porozumění rozdílů podobností v datech. Tato technika patří mezi algoritmy učení bez učitele.
Association rules	Technika asociační pravidla na základě stanovených pravidel pomáhá najít souvislost mezi dvěma a více instancemi, přičemž jsou vyhledávány skryté vzory v datovém vzorku. Tato technika patří mezi algoritmy učení bez učitele.
Outlier Analysis / Detection	Tato technika je zaměřena na vyhledávání specifických záznamů v datovém vzorku, jenž neodpovídají očekávanému vzorci chování, přičemž spadá do kategorie algoritmů kombinace strojového učení s učitelem i bez učitele“
Sequential patterns	Tato technika datového dolování k nalezení a identifikaci podobných vzorů, či trendů, za určité období. Tato technika je využita algoritmy učení bez učitele.
Prediction	Technika Predikce využívá kombinaci technik dolování dat mezi něž patří například i trendy, sekvenční vzorce, shlukování, klasifikace a další. Na základě výskytu události v datech z minulosti jsou predikovány budoucí události.

Poznámka.: Kromě technik pro datové modelování uvedených v tabulce č. 2 existují i další, pokročilejší techniky dolování dat. Pro potřeby této diplomové práce ovšem postačí toto základní dělení, přičemž další informace o pokročilých technikách datového dolování je možné získat ve specializovaných knižních publikacích.

Evaluace modelu

Jednou z klíčových fází při vytváření modelu při dolování dat (případně strojovém učení) je jeho evaluace. Je zapotřebí ověřit, že model skutečně správně funguje a můžeme důvěřovat jeho predikcím v případě, kdy model dostane nová data. Za tímto účelem existuje řada metrik, jenž slouží k evaluaci výkonnosti modelu. Samotná evaluace modelu probíhá nad testovacím vzorkem dat, který nebyl modelu při trénování dostupný. Tím je zajištěno, že samotná evaluace není ovlivněna předchozí zkušeností modelu. Metody pro evaluaci výkonu modelu lze rozdělit do dvou kategorií (31):

- *Holdout* – Principem této evaluace modelu je otestování modelu s jiným daty než se kterými byl natrénován, což poskytuje objektivní odhad výkonu učení modelu. Předností tohoto druhu evaluace je rychlost, jednoduchost a flexibilita. Na druhou stranu tato metoda evaluace může mít vysokou variabilitu výsledků, v případě, kdy existují rozdíly v testovacím a trénovacím datovém vzorku. Datový vzorek je rozdělen do tří částí:
    1. *Trénovací data* – Vzorek dat určený pro budování prediktivních modelů.
    2. *Validační data* – Vzorek dat určený pro evaluaci modelu vytvořeného v trénovací fázi, přičemž tento datový vzorek poskytuje testovací prostředí pro optimalizaci vybraných parametrů a výběr nejlepší alternativy. (Validační data nejsou zapotřebí u všech modelů datového dolování).
    3. Testovací data – Vzorek dat určený pro evaluaci predikční funkce výkonu modelu na datech, jenž nebyly modelu dostupné při trénování modelu.
- *Cross-validation* – Principem této evaluační metody je rozdělení původního datového vzorku na trénovací datový vzorek (tak jako je tomu v případě metody „*Holdout*“) a nezávislý datový vzorek určený k evaluační analýze. Nejčastěji používanou metodou je „*k-fold cross validation*“:
    1. Originální datový vzorek je rozdělen do **k** rovnoměrných částí (počet k definuje uživatel, ve většině případů 5–10).
    2. Proces běží iterativně dle počtu k, kdy v každém průběhu slouží právě jedna část datového vzorku jako testovací/validační data a zbytek datového vzorku jako trénovací data.
    3. Výsledná průměrná chyba je získána průměrem všech chyb proběhlých iterací, a vypovídá o efektivnosti modelu.

Metriky evaluace modelu

Abychom byli schopni efektivně vyjádřit výkonnost modelu, je zapotřebí kvantifikovat vybrané metriky výkonnosti. Nastavení korektních metrik při evaluaci výkonnosti modelu je tedy velmi důležité. Konkrétní metriky se mezi sebou mohou lišit dle techniky dolování dat (například úlohu Klasifikace dat evaluujeme dle jiných metrik než úlohu Regrese),

existují ovšem i některé univerzální metriky, které je možné využít u více druhů technik dolování dat (např. metrika „*precision-recall*“) (31).

Informace o konkrétních metrikách a evaluaci modelu, jenž je výstupem této diplomové práce, naleznete v kapitole 8.

2.3 Zpracování přirozeného jazyka

Zpracování přirozeného jazyka (v anglickém originále *„Natural language processing“* zkráceně NLP), je tradiční disciplína počítačových věd (zejména strojového učení), jenž se zabývá komunikací mezi uživatelem a počítačovým programem. Definici NLP dle společnosti *SAS* lze do českého jazyka přeložit následovně (32):

„Zpracování přirozeného jazyka (NLP) je odvětvím oblasti umělé inteligence, které pomáhá počítači porozumět, interpretovat a manipulovat s lidským jazykem. NLP je založené na mnoha disciplínách, mezi něž patří i počítačové vědy a výpočetní lingvistika tak, aby umožnilo počítači porozumět lidské komunikaci.“

Zpracování přirozeného jazyka je jednou z nejnáročnějších disciplín počítačových věd a strojového učení. Tradice jsou instrukce, jež počítač interpretuje a vytváří z nich počítačové programy, předávané ve formě programovacího jazyka, jenž počítačům poskytuje přesné, jednoznačné a vysoce strukturované instrukce (33). Lidská řeč je ovšem velmi často opakem (může obsahovat dvojsmysly, lze ji vykládat různě dle proměnných v prostředí, slangu atd.)

Cílem této diplomové práce ovšem není se do detailu zabývat oblastí zpracování přirozeného jazyka. Další informace o oblasti zpracování přirozeného jazyka lze nalézt v mnoha knižních publikacích, jenž se danou oblastí zabývají, či v různých odborných článcích. Pro potřeby této diplomové práce je důležitá jedna z podoblastí zpracování přirozeného jazyka, a to oblast tematického modelování.

2.3.1 Tematické modelování

„Tematické modelování“ (v anglickém originále takzvaně *„topic modeling“*), slouží k nalezení a shlukování vybraných slov (takzvaných témat, v anglickém originále „*topics*“) v rámci dat s velkým objemem textu.

Velmi zjednodušenou definici tematického modelování poskytuje webový server *MonkeyLearn* (34, pozn.: překlad autora):

„Tematické modelování je technika strojového učení bez učitele, která je schopna skenovat sadu dokumentů, vybrat slova a věty obsahující opakující se vzory, a automaticky „shlukovat“ skupiny slov a podobné výrazy, které nejlépe charakterizují sadu dokumentů“.

Přesnější definici poskytuje oddělení počítačových vět z *University of Massachusetts Amherst*, které definuje tematické modelování následovně (35, pozn.: překlad autora):

„Tematické modely poskytují jednoduchý způsob analýzy velkého množství neoznačeného textu. Takzvané „téma“ se skládá ze shluku slov, jenž se často vykytují společně. Za využití kontextuálních stop mohou tematické modely spojovat slova s podobnými významy a rozlišovat slova s vícero významy.“

Tematické modelování je tedy technika strojového učení bez učitele, kde dochází k automatické analýze textu a následnému vytvoření shluků slov pro vybranou sadu dokumentů. Princip tematického modelování spočívá v počítání slov a seskupování slov, kde se opakuje vzor z čehož jsou následně odvozována témata v nestrukturovaných datech.

Tato technika je tedy založena na detekci opakujících se vzorů, mezi něž patří například frekvence slov a vzdálenost mezi slovy, přičemž podobný obsah (slova a věty), jenž se vyskytuje často, je shlukován do celků. Na základě těchto informací je možné rychle odvodit významový obsah každé sady textů. Výhodou této techniky je že samotný model není zapotřebí „trénovat“ (34).

Existuje mnoho algoritmů, které lze využít při tematickém modelování. Pro výběr konkrétního algoritmu na prostředí, kde bude algoritmus aplikován (např. některé algoritmy dosahují lepších výsledků při využití s kratšími texty, některé mají naopak lepší výsledky při použití s delšími texty), je zapotřebí vzít v potaz i formát samotných dat. Mezi zmiňované algoritmy patří například:

- *Latent Semantic Analysis* (LSA)
- *Latent Dietrich Allocation* (LDA)
- *Word2vec / Doc2vec*

Informace o algoritmech *Latent Semantic Analysis* (LSA) a *Latent Dietrich Allocation* (LDA) lze nalézt v mnoha knižních publikacích a odborných článcích, jenž se zabývají oblastí tematického modelování. Pro potřeby této diplomové práce jsou relevantní zejména algoritmy Word2vec a jeho rozšíření Doc2vec, jenž budou využity v praktické části této diplomové práce. Charakteristiky technik Word2vec a Doc2vec lze nalézt v kapitole 2.3.2 (respektive 2.3.3), přičemž popis praktického řešení a jeho implementace je uvedena v kapitole 6.

2.3.2 Word2vec

Algoritmus „Word2vec“ byl prvně popsán v odborném článku „„*Distributed Representations of Words and Phrases and their Compositionality*“ (9), přičemž hlavním principem tohoto algoritmu je generování reprezentačních numerických vektorů z textu (slov). Dle webového serveru *Pathmind* lze algoritmus Word2vec definovat následovně (36, pozn.: překlad autora):

„Word2vec je dvouvrstvá neuronová sít, která zpracovává text pomocí „vektorizace“ slov. Vstupem je textový korpus a výstupem je sada vektorů: Tyto vektorové funkce reprezentují obsah vlastního textu.“

Princip tohoto algoritmu spočívá v seskupení vektorů podobných slov ve vektorovém prostoru, přičemž samotná podobnost vektorů je počítána matematicky. Word2vec vytváří

vektory, které jsou distribuovanou numerickou reprezentací slovních rysů (příkladem může být například kontext individuálních slov).

V případě, kdy má algoritmus Word2vec dostatek informací o datech, použití a kontextu slov, je schopen velmi přesně odhadnout význam vlastního slova, na základě jeho předchozích výskytů. Zmíněné odhady mohou být využity k vytvoření asociace vybraného slova s dalšími slovy, nebo ke shlukování dokumentů a jejich klasifikace dle jejich tématu. Výstupem tohoto algoritmu je takzvaný „slovník“, kde je ke každému slovu z původního textu připojeno jeho numerické vyjádření vektorem.

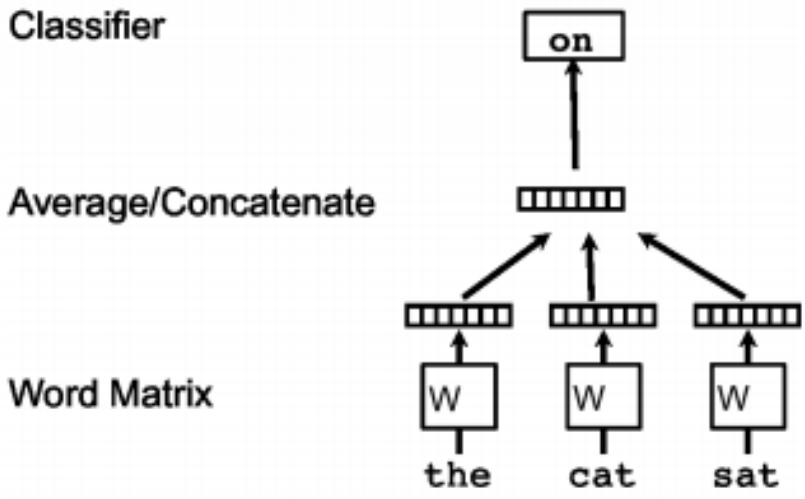
Mezi přednosti algoritmu Word2vec patří pokročilé možnosti využití nejen nad textovými daty. Slova textu lze v tomto kontextu považovat za diskrétní stavy, přičemž stejně tak lze vyjádřit i jiná data (např. genové sekvence, kódy, hudební play listy a další verbální nebo symbolické série, kde lze nalézt opakující se vzory) (36).

Model Word2vec využívá pro vytváření numerické reprezentace dat dva druhy modelů (37):

- *Continuous bag of words (CBOW)*
- *Skip-Gram*

Continuous Bag of Words (CBOW)

Tento model je založena na predikci cílového slova z kontextu okolních slov. Po dokončení trénování modelu je každé slovo reprezentováno pomocí vektorové funkce. Předností této metody je vysoká rychlost a lepší prezentaci slov, jež se v textu vyskytují častěji (38). Obrázek č. 3 graficky znázorňuje algoritmus CBOW.



Obrázek 3 Schéma modelu CBOW, zdroj: (38)

Algoritmus na základě okolních slov „*the*“, „*cat*“, „*sat*“ predikuje cílové slovo „on“.

Skip-Gram

Tento model je opakem modelu CBOW. Na základě cílového slova jsou predikovány všechna okolní slova, přičemž tato metoda dosahuje lepších výsledků při menších objemech dat a lépe reprezentuje vzácnější slova. Na druhou stranu je tento model pomalejší než v případě modelu CBOW (38).

2.3.3 Doc2vec

Algoritmus „Doc2vec" je rozšířením původního algoritmu Word2vec. Toto rozšíření bylo poprvé představeno v odborném článku „*Distributed Representations of Sentences and Documents*" (10). Primárním cílem algoritmu Doc2vec, je numerické vyjádření dokumentu, nezávisle na jeho velikosti. Problém, který zde vyvstává je, že dokumenty nemusí být nutně složeny z logické struktury, tak jako je tomu u slov. Hlavním rozdílem mezi Word2vec a Doc2vec je rozšíření modelu Doc2vec o takzvané „*Paragraph ID*", jenž slouží k identifikaci konkrétního dokumentu/části textu.

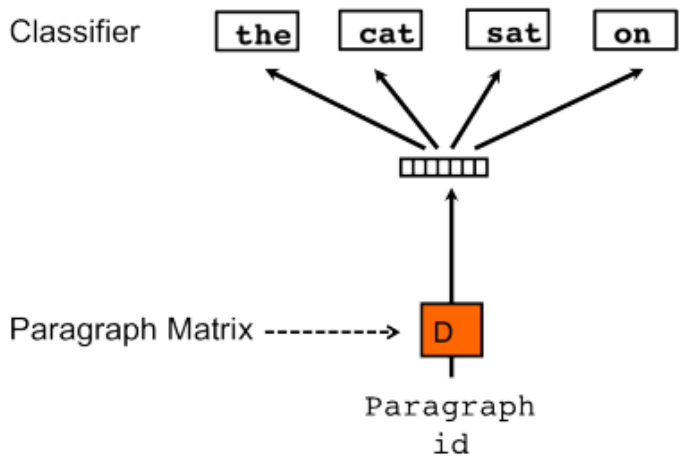
Algoritmus Doc2vec využívá pro vytváření numerické reprezentace dat dva druhy modelů (39):

- *Distributed memory version of paragraph vector (PV-DM)*
- *Distributed bag of words version of paragraph vector (PV-DBOW)*

Distributed bag of words version of paragraph vector (PV-DBOW)

Doc2vec rozšířením modelu Skip-Gram je model *Distributed bag of words version of paragraph vector*. Namísto predikce dalšího slova, jako je tomu v původním verzi modelu, je zde využíván vektor odstavce ke klasifikaci slov v dokumentu. Zmíněný vektor je nazýván „*Paragraph ID*" a slouží jako identifikátor dokumentu. Předností tohoto modelu je jeho rychlost, jelikož využívá méně paměti, protože není zapotřebí ukládat vektory slov (38, 39).

Obrázek č. 4 znázorňuje rozšíření původní model CBOW z modelu Word2vec.



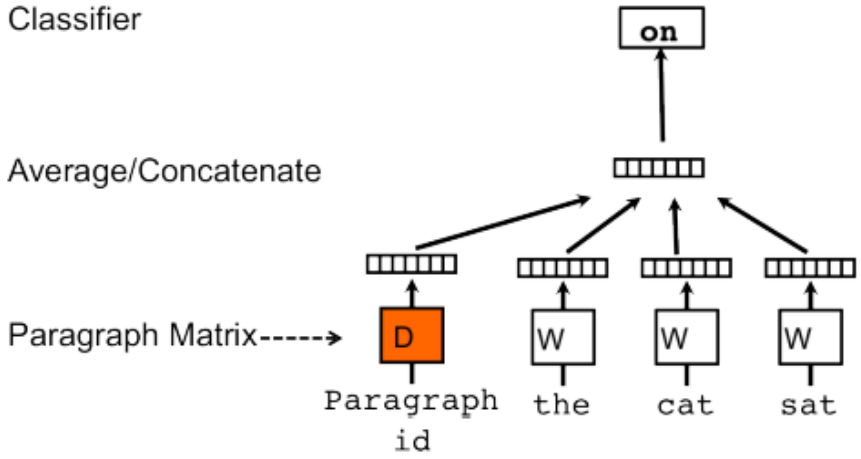
Obrázek 4 Schéma modelu PV-DBOW, zdroj: (10, str. 4)

Distributed memory version of paragraph vector (PV-DM)

Doc2vec rozšířením původního modelu CBOW je model „*Distributed memory version of paragraph vector*“, přičemž původní model je rozšířen o takzvané „*Paragraph ID*“. Tento vektor (D) slouží jako unikátní identifikátor dokumentu/části textu, přičemž vyjadřuje numerickou reprezentaci dokumentu. Samotný model slouží jako paměť, která si pamatuje, co v textu v daném kontextu chybí, nebo vyjadřuje téma dokumentu/části textu (38, 39). Grafické znázornění modelu je dostupné na obrázku č.5.

Průběh modelu:

1. Vektory slov i vektory odstavců jsou inicializovány náhodně.
2. Vektory odstavců jsou vždy přiřazeny pouze jednomu dokumentu. Vektory slov jsou sdíleny napříč všemi dokumenty.
3. Vektory slov i odstavců jsou následně zprůměrovány nebo zřetězovány a předány do stochastického gradientu a gradient je následně získán zpětným šířením.



Obrázek 5 Schéma modelu PV-DM, zdroj: (10, str. 3)

2.4 Rekomendační systémy

Rekomendační systémy jsou v současné době velmi populární v různých odvětvích. Mnoho internetových společností využívá rekomendační systémy k poskytování individualizované nabídky svým uživatelům, což jim často přináší značnou konkurenční výhodu. Méně či více sofistikované rekomendačními systémy jsou již na internetu součástí různých systému s nimiž se většina lidí setkává dennodenně. V některých případech si toho sami uživatelé ani nejsou vědomi. Běžně se s rekomendačními systémy setkáváme například v těchto odvětvích:

- Rekomendační systémy v *e-commerce*
  - Doporučování položek v e-obchodech (např. *Amazon*, *Alza* atd.)
  - Doporučování položek při pronájmu realit (např. *Booking.com*, *Airbnb* atd.)
- Reklamní rekomendační systémy
  - Výběr vhodného reklamního obsahu pro daného zákazníka

- Hudební rekomendační systémy
    - Návrh hudby, která by se uživateli služby mohla líbit na základě jeho preferencí (např. *Spotify*, *Apple Music* atd.)
- Rekomendační systémy pro doporučování video obsahu
    - Návrh filmů a seriálů, které by se uživateli mohli líbit na základě jeho preferencí (např. *Netflix*, *HBO GO* atd.)

Webový server *Techtarget.com* definuje rekomendační systém následovně (40, pozn.: překlad autora):

„Rekomendační systém, také známý jako „recommendation engine“, je počítačový software, který analyzuje dostupná data k návrhu doporučení položky, jež by mohla zajímat uživatele webové stránky, mimo jiné například kniha, video nebo pracovní nabídky“.

Alternativní definici poskytuje *Rishabh Mall* ve svém článku „*Recommender Systems*“ na webovém portále *Towards Data Science* (41, pozn.: překlad autora)::

„Rekomendační systém je počítačový systém, který je schopen predikovat budoucí preference uživatele pro sadu položek, přičemž uživateli doporučí nejvhodnější položky.“

Rekomendační systémy tedy predikují uživatelovy zájmy a na jejich základě pro uživatele vytvářejí individualizovanou nabídku relevantních položek. Uživatelé internetových služeb mají díky nástupu a značnému rozšíření internetu přístup k mnoha různým službám, přičemž často jsou až zahlceni dostupnými službami a jejich obsahem. Řešením tohoto problému jsou právě rekomendační systémy.

Rekomendační systémy lze rozdělit do dvou hlavních kategorií dle způsobu vzniku doporučení:

- *Collaborative Filtering*
- *Content Based*

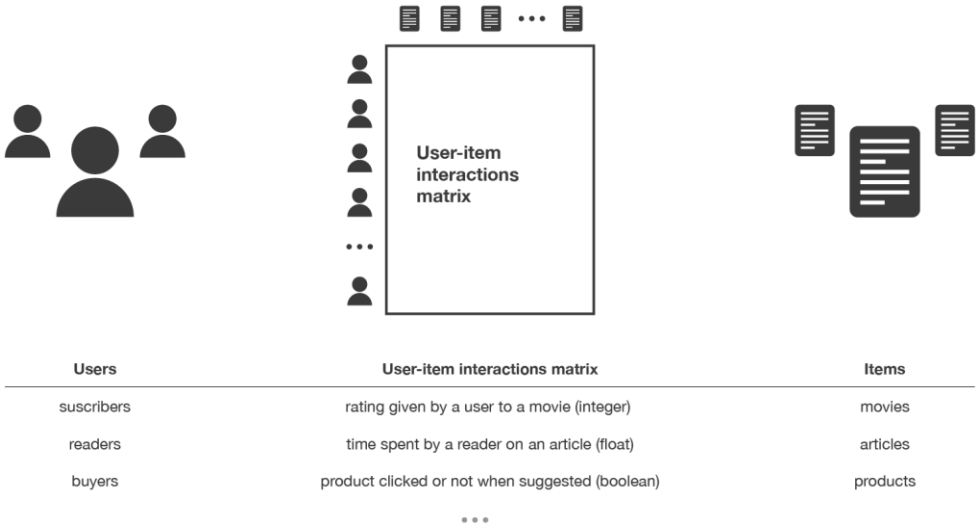
2.4.1 Collaborative Filtering

„Hlavní myšlenkou rekomendačních systémů založených na „Collaborative filtering“ je využití informací o historickém chování nebo názorech existující komunity uživatelů, pro predikci položek, jež budou vybraného uživatele systému zajímat, nebo se mu budou líbit. „ (8, str. 13, pozn.: překlad autora)

Rekomendační systémy založené na principu „*Collaborative Filtering*“ (dále pouze CF), patří v současné době mezi ty nejrozšířenější. V mnoha případech dosahují lepších výsledků než „*Content Based*“ systémy (42). Mezi nejznámější příklady společností, jenž je ve svých produktech využívají patří *YouTube*, *Netflix* a *Spotify*.

Tyto systémy jsou založeny na analýze dat o historických interakcích zaznamenaných mezi uživateli a položkami, na jejichž základě jsou predikována nová doporučení. Zmíněné interakce jsou uloženy v takzvané „*user-interaction matrix*“, což lze do českého jazyka

přeložit jako „matice interakcí uživatele“. Obrázek č.6 poskytuje grafické zobrazení konceptu systémů CF a „*user-interaction matrix*“. V levé části obrázku jsou vyobrazeni uživatelé služeb (např. čtenář), v pravé části jednotlivé položky (např. články) a uprostřed je zobrazena matice, zachycující interakce mezi uživatelem a položkou (v našem případě matice zobrazuje na ose Y čtenáře a na ose X jednotlivé články, přičemž v matici je zaznamenán údaj, indikující jestli vybraný čtenář četl daný článek).



Obrázek 6 Schéma "user-interaction matrix", zdroj: (43)

Na základě dat o historických interakcích uživatele uložených v „*user-interaction matrix*“, jsou detekováni podobní uživatelé / položky a vytvářeny nové predikce a doporučení dle odhadovaných blízkostí.

Hlavním přínosem těchto systémů je, že předem nevyžadují žádné doplňující informace o uživatelích či položkách. Čím více uživatelů interaguje s položkami, tím více jsou nová doporučení přesná. Pro fixní sadu uživatelů a položek přináší nové interakce zaznamenané v čase, nové informace na jejichž základě dochází k zefektivnění systému.

Hlavním negativním úkazem u tohoto druhu systému je takzvaný „*cold start problem*“. V případě, kdy vycházíme z historických dat o interakci mezi uživatelem a položkami, vzniká problém při vložení nového uživatele, nebo nové položky do systému. V takovém případě není možné novému uživateli nic doporučit / není možné novou položku doporučit žádnému uživateli. Existuje řada strategií, jak tento problém řešit, mezi něž patří například:

- Doporučovat novým uživatelům náhodné položky, nebo nové položky náhodným uživatelům (*random strategy*)
- Doporučovat populární položky novým uživatelům nebo nové položky nejaktivnějším uživatelům (*maximum expectation strategy*)
- Doporučovat sadu položek novým uživatelům nebo doporučovat novou položku sadě uživatelů (*exploratory strategy*)
- Alternativně použít jiný druh systému na počátku životního cyklu uživatele / položky

Tuto kategorii rekomendačních systémů lze ještě rozdělit na dvě subkategorie: „*Memory Based*“ a „*Model Based*“ (43).

Memory Based

„*Memory Based*“ systém přímo zpracovává data o zaznamenaných interakcích mezi uživatelem a položkou, přičemž základním předpokladem je, že neexistuje model chování uživatele. Tyto systémy jsou nejčastěji založeny na vyhledávání nejbližšího souseda (v anglickém originále “*Nearest Neighbor Search*”) a jsou využívána pouze data z „*user-interaction matrix*“ (43).

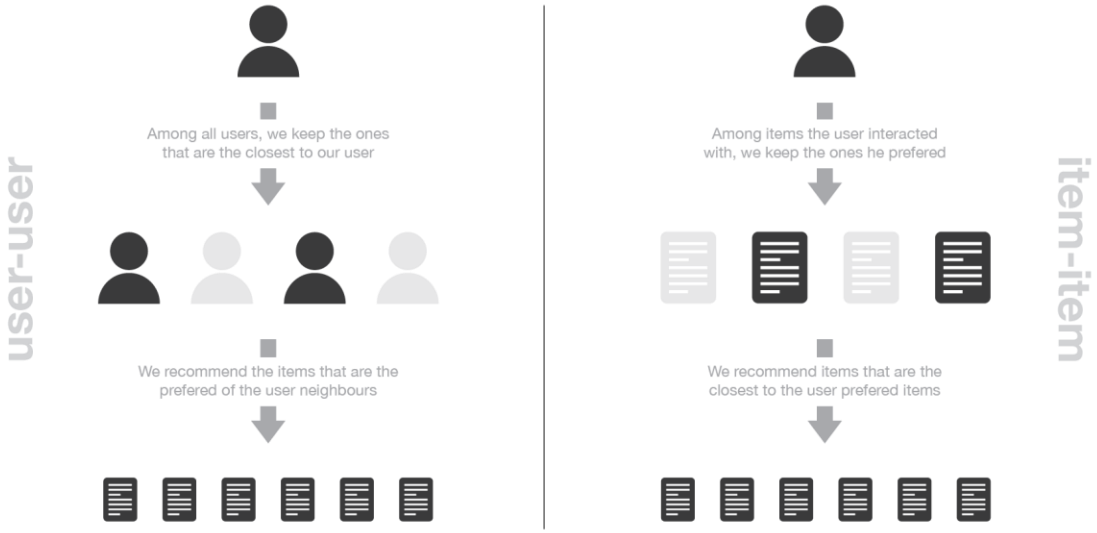
Lze je rozdělit do dvou kategorií:

- *User-user* – Nová doporučení jsou pro uživatele vytvářena na základě identifikace uživatelů s co nejvíce podobným profilem interakcí, jako je tomu v případě zvoleného uživatele (je využita metoda nejbližšího souseda), přičemž jsou uživateli doporučeny položky populární mezi jeho sousedy, které uživatel doposud neviděl. Tato metoda je považovaná za uživatelsko-centrickou, jelikož prezentuje uživatele na základě jejich interakcí s položkami a evaluuje vzdálenost mezi vektory jednotlivých uživatelů.
- *Item-item* – V tomto případě, jsou nová doporučení založena na uživatelově předchozích „pozitivních“ interakcích s podobnými položkami. Dvě položky jsou považovány za podobné v momentě, kdy s nimi většina uživatelů interaguje podobným způsobem (např. položka od většiny uživatelů dostane hodnocení 5 hvězdiček). O této metodě se říká že je položko-centrická, jelikož prezentuje jak uživatelé s položkami interagovali a evaluuje vzdálenosti mezi vektory jednotlivých položek.

Metoda „*user-user*“ je založena na hledání podobných uživatelů z hlediska interakcí s položkami. Ve většině případů provádí uživatel interakce pouze s několika položkami, proto je tato metoda citlivá na zaznamenané interakce (vysoký rozptyl interakcí). Vzhledem k tomu, že konečná doporučení jsou vytvářena pouze na interakcích uživatelů se zájmy podobných vybranému uživateli, poskytuje tato metoda více personalizované výsledky (nízké zkreslení výsledků).

Metoda „*item-item*“ funguje na principu hledání podobných položek, na než už uživatel „pozitivně“ reagoval. Vzhledem k tomu, že s položkou již ve většině případů interagovala řada uživatelů, vyhledávání podobných položek je méně citlivé na jednotlivé interakce (menší rozptyl interakcí) (43). Negativním efektem u této metody je, že zohledněny jsou interakce od všech uživatelů, což způsobuje že metoda je méně personalizovaná (vyšší zkreslení výsledků).

Obrázek č. 7 poskytuje grafické porovnání metod „*user-user*“ a „*item-item*“. V levé části obrázku je vykreslená metoda „user-user“, kdy jsou nejdříve z výběru uživatelů zachováni uživatelé nejblíže cílovému uživateli, a následně jsou tomuto uživateli zobrazeny preferované položky nejbližších uživatelů. V pravé části obrázku je vykreslená metoda „*item-item*“, kde jsou z výběru ponechány položky, jenž uživatel preferoval, a následně jsou uživateli zobrazeny podobné položky.



Obrázek 7 Porovnání metod „user-user“ a „item-item“, zdroj: (43)

Model Based

„Model Based“ systémy předpokládají existenci takzvaného „generativního“ modelu chování uživatele, jenž vysvětluje interakce mezi uživatelem a položkou, a snaží se daný model odhalit a popsat, aby bylo na jeho základě možné dělat nová doporučení. Oproti „*Memory Based*“ systému využívá tento systém informací o takzvaných „*user-item*“ interakcích.

2.4.2 Content Based

V případě předchozí metody (CF) byly brány v potaz pouze data o interakci uživatele s položkami, v případě „*Content Based*“ metod (dále pouze CB), jsou využívána i další data o uživatelích a/nebo položkách (v případě uživatele se může jednat např. o pohlaví uživatele, věk uživatele, nebo jakékoliv osobní informace).

Cílem této metody je vytvořit model, založený na dostupných „*features*“, které vysvětlují pozorované interakce mezi uživatelem a položkou. Na rozdíl od metod CF, metody CB mají mnohem lepší odolnost vůči „*cold start problem*“. Nové uživatele a položky lze charakterizovat dle jejich vlastností a parametrů, tudíž je možné pro ně vytvářet relevantní doporučení. V případě metod CB lze úlohy rozdělit následovně:

- Klasifikace – Predikujeme, jestli se položka uživateli bude líbit nebo ne.
- Regrese – Predikujeme hodnocení (např. počet hvězdiček), jenž uživatel dá položce.

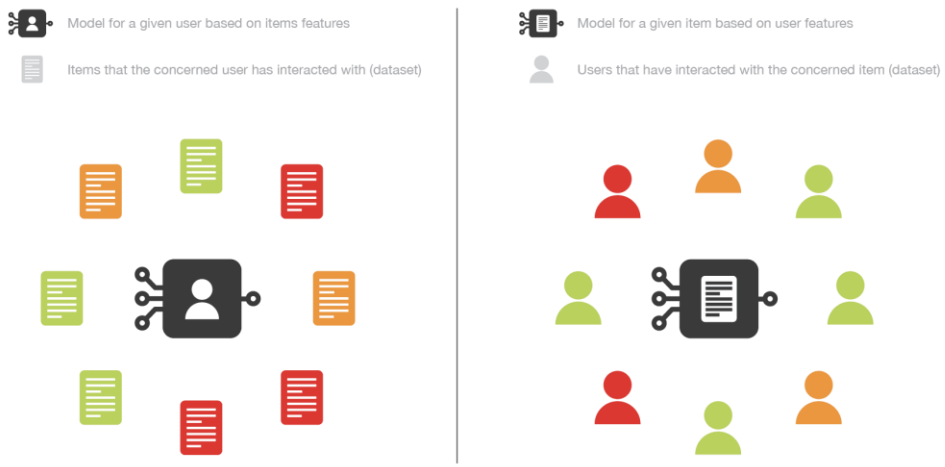
Dále je možné úlohy rozdělit dle zaměření modelu:

- *Item-centered* – Úloha je založena na uživatelských atributech. V tomto případě vzniká jeden model, podle položky na základě uživatelských atributů. Snažíme se odpovědět na otázku „Jaká je pravděpodobnost, že se každému uživateli bude tato položka líbit?“ (klasifikace), případně „Jaké hodnocení dá této položce každý

uživatel?“ (regrese). Vzhledem k tomu, že s každou položkou interagovalo mnoho uživatelů (kteří mohou ale nemusí mít stejné charakterní atributy), vznikají touto metodou poměrně robustní modely, které ale neposkytují vysokou míru personalizace (vyšší zkreslení výsledků).

- *User-centered* – Úloha je založena na atributech položek. V tomto případě trénujeme jeden model, podle uživatele na základě atributů položek. Odpovídáme na otázku „Jaká je pravděpodobnost, že tento uživatel bude mít rád každou položku?“ (klasifikace), případně “Jaké hodnocení dá tento uživatel každé položce?” (regrese). Tento model bude více personalizovaný, jelikož bere v potaz pouze interakce od konkrétního uživatele (nižší zkreslení výsledků), na druhou stranu bude model méně robustnější, jelikož vybraný uživatel bude ve většině případů interagovat s méně položkami.

Obrázek č. 8 poskytuje grafické porovnání úloh „*user-centered*“ a „*item-centered*“.



Obrázek 8 Porovnání úloh „user-centered“ a „item-centered“, zdroj: (43)

Ve většině případů je mnohem jednodušší využívat atributy položek, jelikož popisky nových položek bývají daleko obsáhlejší než atributy nových uživatelů. Noví uživatelé ve většině případů neradi vyplňují své osobní údaje, kdežto uživatelé vytvářející popisky položek se snaží položky co možná nejlépe popsat, aby došlo k jejich správné klasifikaci (43).

2.4.3 Evaluace rekomendačního systému

Stejně tak jako je tomu u většiny algoritmů strojového učení, je nutné mít i u rekomendačních systémů způsob evaluace výkonnosti daného algoritmu, abychom byli schopni provést výběr nejvhodnějšího algoritmu pro potřeby konkrétního zadání. Pro rekomendační systémy existují dva způsoby evaluace: *Metrics Based Evaluation* a *Human Based Evaluation*.

Metrics Based Evaluation

V případech, kdy je rekomendační systém založen na modelu, jehož výstupem je numerická hodnota, lze pro ověření výkonnosti modelu využít klasické metriky pro ověření chyb např. *Mean Square Error* (do českého jazyka lze přeložit jako střední kvadratická chyba). Další možností je převést hodnoty výstupu do binární formy na základě stanovených hranic (hodnoty nad hranicí jsou kladné, hodnoty pod hranicí jsou záporné). Následně, po porovnání se souborem uživatelských interakcí s položkami, můžeme vyhodnotit přesnost modelu na testovacím datovém vzorku (43). V obou případech je model natrénován pouze na části dat, a část dat je oddělena pro testování výkonnosti modelu.

Rekomendační systém založený na modelu, jehož výstupem nejsou numerické hodnoty (jeho výstupem je např. seznam doporučení), definujeme přesnost odhadem podílu doporučených položek, jež ve skutečnosti vyhovují danému uživateli. Při odhadu přesnosti nelze brát v potaz nové položky se kterými uživatel neinteragoval, ale pouze položky z testovacího datového vzorku, kde uživatel poskytl zpětnou vazbu (43).

Human Based Evaluation

Cílem každého rekomendačního systému je dělat relevantní a užitečná doporučení. Při evaluaci rekomendačních systému z „lidského" pohledu je zapotřebí ověřit, že rekomendace jsou dostatečně diverzifikované a jsme schopni je vysvětlit.

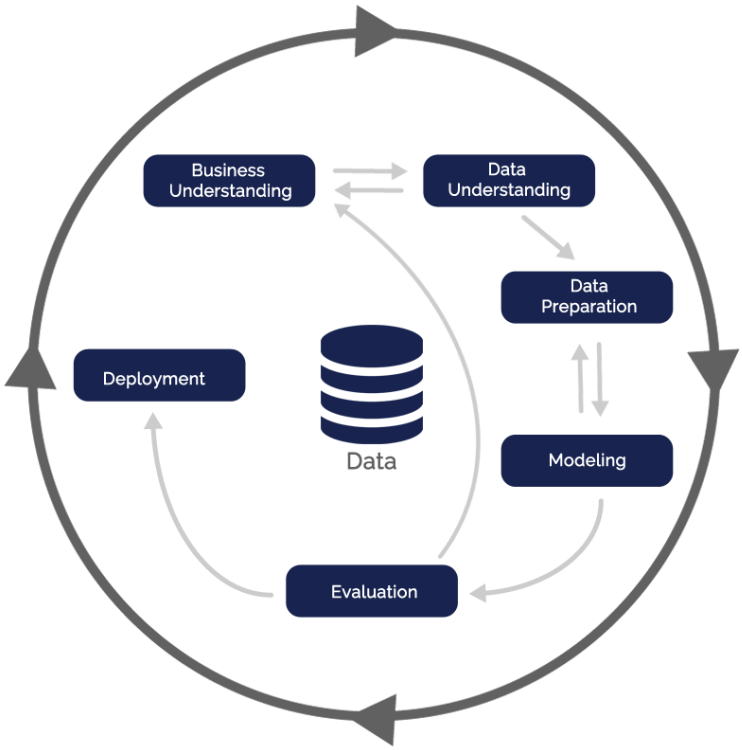
Diverzita doporučení modelu je u rekomendačních systémů důležitá, jelikož chceme zamezit u uživatele vytvoření takzvané „*information confinement area*" (do českého jazyka lze přeložit jako „informační bublina"). Při výpočtu vzdáleností jednotlivých doporučení chceme zajistit, aby si doporučení nebyla moc blízká (příliš úzký okruh podobných položek) a zároveň aby si nebyla moc vzdálená (model nebere v potaz uživatelovy preference). Vhodný příklad je uveden na webovém portále *Towards Data Science* (43). Pokud vezmeme v potaz rekomendační systém pro filmový obsah, nechceme aby rekomendační systém doporučoval příliš sourodé položky (např. *Star Wars 1*, *Star Wars 2*, *Star Wars 3*, …), ale je vhodné lépe diverzifikovat doporučení (např. Star Wars 1, Indiana Jones 3, Star Trek, …).

Doporučení, jež vzniknou v rekomendačním systému, by měla být vždy vysvětlitelná. V minulosti bylo dokázáno, že pokud uživatelé nerozumí tomu, proč jim byla daná položka doporučena, ztratí důvěru v rekomendační systémy (43). Z tohoto důvodu je vhodné uvádět, proč byla daná položka uživateli doporučena (např. můžeme u doporučované položky uvést větu „lidem, kterým se líbila zobrazovaná položka, se také líbila tato položka" a jiné).

3 Metodika CRISP-DM

Jak již bylo zmíněno v kapitole 2.2.2, v rámci praktické části této diplomové práce bude pro implementaci vlastního projektu využita metodika CRISP-DM. Metodika „*Cross-industry standard process for data mining*“ (dále jen CRISP-DM), je v současné době jednou z nejvyužívanějších metodik pro implementaci procesu dolování dat. Samotná metodika se skládá ze šesti fází/částí: „*Business Understanding*“, „*Data Understanding*“, „*Data Preparation*“, „*Modelling*“, „*Evaluation*“ a „*Deployment*“.

Mezi přednosti této metodiky patří její jednoduchost, flexibilita a přenositelnost. V podnikovém prostředí často dochází k záměně jednotlivých fází metodiky. To značí, že jednotlivé fáze metodiky jsou vykonávány v různém pořadí, často je nutné se vrátit k předchozí fázi, či úkony prováděné v současné fázi zopakovat (44). Obrázek č. 9 zobrazuje jednotlivé fáze metodiky CRISP-DM.



Obrázek 9 Grafické znázornění metodiky CRISP-DM, zdroj: (45)

Cílem této kapitoly není podrobně zdokumentovat jednotlivé fáze metodiky CRISP-DM. Tato metodika již byla podrobně zdokumentována v mnoha knižních publikacích, mezi něž patří například kniha „*Dobývání znalostí z databází*“, z níž bylo čerpáno při psaní této kapitoly (7). První část této kapitoly uvádí jednotlivé fáze této metodiky, jejich cíle a obsah. Druhá část této kapitoly uvádí tuto metodiku do kontextu praktické části této diplomové práce, přičemž obsahuje tabulku, jenž dokumentuje, kterými částmi této metodiky se zabývají vybrané kapitoly této práce.

3.1 Fáze metodiky CRISP-DM

3.1.1 Business Understanding

"*Tato úvodní fáze je zaměřena na pochopení cílů úlohy a požadavků na řešení formulovaných z manažerského hlediska. Manažerská formulace musí být následně převedena do zadání úlohy pro dobývání znalostí z databází".* (7, str. 25)

Název inicializační fáze této metodiky „*Business Understanding*" lze do českého jazyka přeložit jako porozumění businessu nebo obchodní porozumění. V této fázi jsou na základě obchodních cílů a dostupných prostředků stanoveny cíle procesu dolování dat. Velmi často má zadavatel projektu (klient) zkreslenou představu o výstupu procesu dolování dat, tudíž je zapotřebí sjednotit požadavky zadavatele projektu s reálnými možnostmi výstupu. Mezi výstupy této fáze patří mimo jiné následující (44):

- Nastavené cíle implementace procesu dolování dat
- Zmapování dostupných prostředků
- Nastavená kritéria úspěchu procesu implementace dolování dat z pohledu klienta

3.1.2 Data Understanding

"Fáze porozuměním datum začíná prvotním sběrem dat. Následují činnosti, které umožní získat základní představu o datech, která jsou k dispozici (posouzení kvality dat, první "vhled" do dat, vytipování zajímavých podmnožin záznamů v databázi …). Obvykle se zjišťují různé deskriptivní charakteristiky dat (četnosti hodnot různých atributů, průměrné hodnoty, minima, maxima apod.), s výhodou se využívají i různé vizualizační techniky". (7, str. 25)

Fáze "Data Understanding" (do českého jazyka můžeme přeložit jako porozumění datům) je zaměřena na zjištění jaká data máme k dispozici, jejich sběr a kontrolu kvality dat.

3.1.3 Data Preparation

„Příprava dat zahrnuje činnosti, které vedou k vytvoření datového souboru, který bude zpracováván jednotlivými analytickými metodami. Tato data by tedy měla

- *obsahovat údaje význačné pro danou úlohu,*
- *mít podobu, která je vyžadována vlastními analytickými algoritmy.*

Příprava dat tedy zahrnuje selekci dat, čištění dat, transformaci dat, vytváření dat, integrování dat a formátování dat. Tato fáze je obvykle nejpracnější částí řešení celé úlohy. Jednotlivé úkony jsou obvykle prováděny opakovaně, v různém pořadí." (7, str. 26)

Fáze „*Data Preparation*" (do českého jazyka lze přeložit jako čištění dat) se zabývá zejména různými transformacemi a úpravou dat do takového stavu, aby na nich v další fázi mohl být

natrénován model. V rámci této fáze je rovněž ověřeno, že s dostupnými daty bude možné splnit cíle stanovené ve fázi „*Business Understanding*“.

3.1.4 Modelling

“V této fázi jsou nasazeny analytické metody (algoritmy pro dobývání znalostí). Obvykle existuje řada různých metod pro řešení dané úlohy, je tedy třeba vybrat ty nejvhodnější (doporučuje se použít více různých metod a jejich výsledky kombinovat) a vhodně nastavit jejich parametry. Jde tedy opět o iterativní činnost (opakovaná aplikace algoritmů s různými parametry), použití analytických algoritmů může navíc vést k potřebě modifikovat data, a tedy k návratu k datovým transformacím z předcházející fáze”. (7, str. 26 - 27)

Fáze “*Modelling*” se tedy zabývá vytvářením a trénováním vlastních modelů pro dolování dat. Prvním krokem je výběr konkrétního algoritmu, jenž bude na data aplikován (např. rozhodovací stromy, neuronové sítě atd.), přičemž na základě vybraného modelu je následně zvolena technika ověření správnosti výsledků modelu. Dále je vytvořen/natrénován vlastní model se zadanými parametry.

3.1.5 Evaluation

“V této fázi jsme se dopracovali do stavu, kdy jsme nalezli znalosti, které se zdají být v pořádku z hlediska metod dobývání znalostí. Dosažené výsledky je ale ještě třeba vyhodnotit z pohledu manažerů, zda byly splněny cíle formulované při zadání úlohy. ” (7, str. 27)

Fáze „*Evaluation*“ (v českém jazyce evaluace nebo vyhodnocení) je tedy kontrolní fáze, kdy dochází k vyhodnocení modelu z pohledu obchodních kritérií stanovených v první fázi. Rovněž je zkoumáno, jestli model není vadný z pohledu obchodních cílů. Techniky a metriky samotného ověření se liší dle použitého algoritmu.

3.1.6 Deployment

„Vytvořením vhodného modelu řešení úlohy obecně nekončí. Dokonce i v případě, kdy řešenou úlohou byl pouhý popis dat, je třeba získané znalosti upravit do podoby použitelné pro zákazníka (manažera – zadavatele úlohy). Podle typu úlohy tedy využití (nasazení) výsledků může na jedné straně znamenat prosté sepsání závěrečné zprávy, na straně druhé pak zavedení (hardwarové, softwarové, organizační) systému pro automatickou klasifikaci nových případů.

Ve většině případů je to zákazník a nikoliv analytik, kdo provádí kroky vedoucí k využívání výsledků analýzy. Proto je důležité, aby pochopil, co je nezbytné učinit pro to, aby mohly být dosažené výsledky využívány efektivně“. (7, str. 27)

Ve fázi „*Deployment*“ dochází k vlastnímu nasazení modelu do produkčního prostředí. To značí, že do modelu jsou dodávána nová data, na jejichž základě model následně produkuje výstupy. Implementace modelu do produkčního prostředí (produkcionalizace) je finálním

krokem metodiky CRISP-DM, přičemž je zapotřebí model pravidelně kontrolovat, aby byla ověřena jeho funkčnost a korektnost. V průběhu času se některé faktory ovlivňující model mění a je zapotřebí model přetrénovat.

Poznámka autora.: Fáze „*Deployment*“ není součástí této diplomové práce a proto nebude v textu diplomové práce popsána. Samotný model bude u IPTV poskytovatel implementován do produkčního prostředí, ale z důvodu bezpečnosti a nebylo možné v této diplomové práci uvést detaily produkčního prostředí IPTV poskytovatele a vlastní implementace modelu do produkčního prostředí.

3.2 Korespondující kapitoly diplomové práce

V rámci tabulky č. 3 jsou uvedeny jednotlivé fáze metodiky CRISP-DM a čísla kapitol, jenž se danou problematikou zabývají.

Tabulka 3 Korespondující kapitoly DP s metodikou CRISP-DM

Název fáze	Číslo kapitoly	Popis kapitoly
„Business Understanding”	Úvod	Úvodní kapitola je zaměřena na popis současného stavu, omezení a cíle této diplomové práce, tudíž z pohledu této fáze poskytuje porozumění obchodním cílům a prostředí IPTV poskytovatele.
„Business Understanding”	2	Druhá kapitola této diplomové práce je zaměřena na teoretickou charakteristiku technologií a konceptů, jenž budou využity v rámci této diplomové práce. Z pohledu této fáze poskytuje zmíněná kapitola informace o technologiích, využitých v rámci praktické části této práce.
„Data Understanding”	5	První subkapitola páté kapitoly této diplomové práce je zaměřena na analýzu dostupných datových zdrojů. V rámci této kapitoly jsou popsány datové zdroje, je vysvětlen jejich význam a uvedeny datové typy. Z pohledu této fáze uvádí tato kapitola o datech využívaných v praktické části této diplomové práce.
„Data Preparation”	5	Druhá subkapitola páté kapitoly této diplomové práce uvádí postup transformace dat ze zdrojových souborů, do cílových

		tabulek. V rámci této fáze uvádí zmíněná kapitola informace o transformacích, jenž nad využívanými daty proběhli.
„*Data Preparation*	6	Druhá subkapitola šesté kapitoly této práce obsahuje popis transformace dat z tabulek do CSV souborů, jenž budou využity pro převod dat do numerického (vektorového) formátu.
„*Modeling”*	7	Sedmá kapitola této práce je zaměřena na vytvoření prediktivního modelu. V rámci této fáze uvádí zmíněná kapitola informace o činnostech spojených s vytvářením modelu.
„*Evaluation*“	8	Osmá kapitola této práce je zaměřena na evaluaci modelu, vytvořeného v rámci sedmé kapitoly. Z pohledu této fáze uvádí tato kapitola informace o využitých metrikách pro evaluaci modelu a ověřuje funkčnost vytvořeného modelu.

4 Vývojové prostředí

Čtvrtá kapitola této práce je zaměřena na popis technologií, které byly využity v praktické části této diplomové práce. Obsahově tato kapitola nepokrývá detailní popis zmíněných technologií, nýbrž se soustředí na jejich základní charakteristiku a popis jejich využití v praktické části této diplomové práce. Pro podrobnější informace doporučuje autor této práce webové portály jednotlivých technologií a různé technologické blogy, kde se těmto technologiím autoři článků věnují do větších podrobností.

4.1 Apache Kafka

„*Apache Kafka*“ je nástroj pro distribuované ukládání dat, jenž je optimalizované pro příjem a zpracování streamovaných dat v reálném čase. Streamovaná data jsou data, která jsou neustále generována velkým množstvím datových zdrojů, přičemž tyto datové zdroje generují data simultánně. Streamovací platforma musí zvládat zpracování neustálého přísunu dat a zpracovat data sekvenčně a inkrementálně. Samotný nástroj poskytuje tři základní funkcionality:

- Vytváření a sběr streamovaných dat.
- Efektivní uložení streamovaných dat v pořadí, v němž byla data generována.
- Zpracování streamovaných dat v reálném čase.

Primárním využitím tohoto nástroje je vytváření aplikací a mechanismů pro přenos a zpracování dat v reálném čase z datových toků. Kombinuje příjem, ukládání a zpracování dat tak, aby byla umožněno ukládání a analýza historických dat, jakož i dat v reálném čase (46).

Využití v praktické části diplomové práce: Poskytovatel IPTV využívá nástroj *Kafka* pro sběr a dočasné uložení dat o tom, jak uživatelé využívají jeho IPTV a VoD služby. Data jsou uložena do takzvaných „session“, jenž jsou zaznamenávány od počátku využívání služeb do ukončení využívání služeb IPTV poskytovatele. Získávaná data jsou poměrně detailní. V rámci sběru dat je monitorováno, kdo dané služby využíval, jaké služby byly využívány a jak dlouho. Data jsou pomocí tohoto nástroje následně doručena do vzdáleného úložiště na platformě *Azure* od společnosti *Microsoft*, kde jsou uložena a dále zpracována.

4.2 Microsoft Azure

„Microsoft Azure, původně známé jako Windows Azure, je veřejná „cloud computing“ platforma od společnosti Microsoft. Poskytuje řadu cloudových služeb, včetně výpočetních, analytických, úložných a síťových. Uživatelé si dle svých preferencí vybírají služby pro vývoj a škálování nových aplikací, nebo spouštění a provoz stávající aplikace ve veřejném cloudu.“ (47)

Tato cloudová platforma poskytuje služby „*Infrastructure-as-a-Service*“ (do českého jazyka lze přeložit jako infrastruktura jako služba) a „*Platform-as-a-Service*“ (do českého jazyka lze přeložit jako platforma jako služba). Vzdálený provoz IT aplikací a zařízení umožnuje mnohým zákazníkům této společnosti ušetřit na nákladech spojených s lokálním provozem. Platforma sama o sobě je multifunkční a poskytuje mnohé nástroje a služby pro vývoj, provoz a správu nových a stávajících aplikací zákazníka.

Využití v praktické části diplomové práce: Poskytovatel IPTV využívá platformu Azure od společnosti Microsoft jako „*Infrastructure-as-a-Service*“ řešení. Na této platformě jsou ukládána a zpracovávána data, které sem pošle nástroj *Kafka* zmíněný v kapitole 4.1. Poslední záznamy převzaté z nástroje *Kafka* jsou každých 15 minut uloženy do nového souboru v datovém formátu *Parquet* do úložiště na platformě Azure. Následně, poté co jsou převzata všechna data za daný den, proběhne jejich transformace (jsou vybrána relevantní pole) a anonymizace (odstranění citlivých dat zákazníků), a jsou uloženy do jednoho většího souboru v datovém formátu *Parquet*, obsahující záznamy za celý den. Konkrétní služby a jejich popis, jenž byly při psaní této diplomové práce využity, jsou uvedeny v kapitole 4.2.1.

4.2.1 Využité služby a nástroje

Jak již bylo uvedeno v předchozích kapitolách, portál Azure od společnosti Microsoft poskytuje mnoho různých služeb určených pro vývoj, implementaci a provoz softwarových aplikací. Poskytovatel IPTV má na portále vedenou mimo jiné i velkou část své „*Big Data*“ platformy. V rámci této diplomové práce bohužel není možné (z důvodu bezpečnosti) rozvádět konkrétní řešení a implementaci zmíněné platformy, ani to není cílem této diplomové práce. Na druhou stranu je možné popsat nástroje, které byly využity pro vytvoření datové pumpy, která slouží k přenosu a transformaci dat mezi nástrojem *Kafka* a úložištěm na portálu Azure. Dále je také možné uvést služby portálu Azure, jenž budou využity při vytváření prediktivního modelu. Zmíněné služby jsou uvedeny v tabulce č. 4.

Tabulka 4 Služby portálu Azure využité v praktické části diplomové práce, zdroj: (autor)

Anglický název	Popis služby
Data Factory	Služba „*Data Factory*“ je nástroj pro integraci datových zdrojů a vytváření ETL datových pump, přičemž tato služba obsahuje více jak 90 konektorů pro různé zdroje dat (databáze, datové sklady atd.), a nabízí možnost využití vlastních datových konektorů (48). V rámci této služby je nakonfigurována ETL datová pumpa, která přenáší data z nástroje *Kafka* do úložiště portálu *Azure*. Dále tato datová pumpa provádí vybrané transformace dat (výběr relevantních sloupců, čištění dat, anonymizace dat).
Blob Storage	Služba „*Blob Storage*“ funguje jako škálovatelné úložiště pro velké objemy nestrukturovaných dat, přičemž její hlavní výhodou

	je značné zefektivnění a snížení nákladů (v porovnání s lokálním ukládáním dat u zákazníka) (49). Tato služba pro ukládání nestrukturovaných dat je využita při příjmu dat z nástroje *Kafka*. Data jsou z tohoto nástroje streamována do zmíněného úložiště, odkud jsou postupně přesunuta, zpracována a uložena v datovém úložišti *Data Lake*.
Data Lake	Služba „*Data Lake*“ slouží jako rozšíření služby *Blob Storage*, zaměřené na pokročilé možnosti analýzy strukturovaných i nestrukturovaných dat a je optimalizována pro analytické úlohy (50). Poskytovatel IPTV využívá službu *Data Lake* pro uložení již strukturovaný dat. Ve službě *Blob Storage* jsou data uložena v nestrukturalizované formě, přičemž v rámci zpracování a přesunu dochází k jejich strukturalizaci. Struktura uložených dat je uvedena v kapitole 5.
Data Bricks	Služba „*Databricks*“ poskytuje možnosti vytvoření *Spark* (podrobnosti v kapitole 4.3) clusteru v prostředí online cloudu, přičemž nabízí rozšířené možnosti škálovatelnosti a kolaborace na projektech strojového učení a pokročilé analytiky. Tato služba podporuje programovací jazyky Python, R, Java a SQL a mnoho rozšiřujících balíčků a knihoven určených pro zpracování dat a strojové učení (51). Tato služba bude využita při analýze a transformaci dat, při vytváření datových tabulek a datových vzorků pro převod do numerické (vektorové) reprezentace dat.
Machine Learning Service	Služba „*Machine Learning Service*“ poskytuje nástroje a zdroje pro zpracování úloh strojového učení, v rámci celého životního cyklu úloh strojového učení. Poskytuje zejména nástroje na vytváření, trénování a produkcionalizaci modelů strojového učení (52). Tato služba bude využita při převodu dat z relační databázové formy do vektorové formy, a následně při vytváření prediktivního modelu.

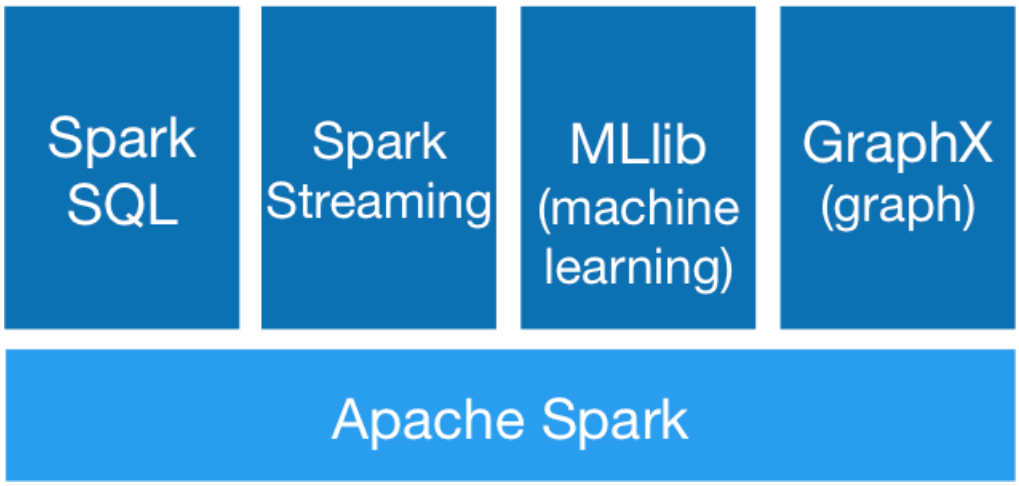
Pozn.: Kromě služeb uvedených v tabulce č. 4, využívá IPTV poskytovatel i další služby při zpracování datových pump (například virtuální počítačové jednotky, vytváření privátních sítových spojení atd.). Podrobnější popis těchto služeb ale není z bezpečnostních důvodů v této diplomové práci možný, ani to není cílem diplomové práce.

4.3 Apache Spark

Open-source nástroj „*Apache Spark*“ je v současné době jeden z nejpoužívanějších výpočetních nástrojů pro pokročilou datovou analytiku a strojové učení, určený pro zpracování velkých objemů dat. Nástroj poskytuje API pro programovací jazyky Python, R, Java a SQL. Tento nástroj vznikl v roce 2009 na univerzitě *UC Berkley* v americké Kalifornii, později byl předán *Apache Software Foundation*, jenž jej spravuje dodnes. Skládá se z pěti komponent jejichž popis je uvedený v kapitole 4.3.1. Podrobnější informace lze nalézt na webových stránkách s dokumentací nástroje (53).

4.3.1 Moduly Apache Spark

Nástroj „*Spark*“ se skládá z pěti komponent, jež jsou vzájemně úzce integrované a propojené. *Spark* je ve své podstatě pokročilý výpočetní nástroj, který obsahuje funkce pro plánování, distribuci a monitorování aplikací, jenž se skládají z výpočetních úkonů, které běží nad mnoha výpočetními stanicemi nebo takzvaným výpočetním „clusterem“. Výhodou jádra nástroje *Spark* je jeho rychlost a univerzálnost, což mu umožnuje provozovat vybrané komponenty pokročilé datové analytiky (např. úlohy strojového učení a SQL). Zmíněné komponenty jsou navrženy k úzké spolupráci tak, aby je bylo možné kombinovat jako balíčky v softwarovém projektu (54).



Obrázek 10 Komponenty nástroje *Spark*, zdroj: (55)

Obrázek č. 10 poskytuje grafické znázornění komponent nástroje *Spark*, přičemž popis jednotlivých komponent je uvedený v tabulce č. 5.

Tabulka 5 Popis komponent nástroje *Spark*, zdroj: (54)

Anglický název	Popis komponenty
Spark Core	Tato komponenta obsahuje základní funkce pro plánování úkolů, správu paměti, opravu chyb, interakci s úložnými systémy, a API,

	které definuje „*Resilient Distributed Datasets*“ (unikátní datovou strukturu nástroje *Spark*).
Spark SQL	Tato komponenta je zaměřena na práci se strukturovanými daty, za pomoci skriptovacího jazyka SQL nebo HQL, přičemž umožnuje manipulaci s daty v různých datových formátech např. Parquet, JSON, CSV a další. Rovněž je v této komponentě implementována podpora pokročilé manipulace s daty z programovacích jazyků python, R, Java a Scala.
Spark Streaming	Komponenta „*Spark Streaming*“ umožnuje zpracování dat streamovaných v reálném čase. Příkladem služby streamující data je nástroj *Kafka*.
MLlib	Komponenta „*MLlib*“ obsahuje knihovnu často využívaných funkcí strojového učení. Tato knihovna obsahuje mnoho algoritmů strojového učení (např. klasifikace, regrese, rozhodovací stromy a jiné), a další funkce pro evaluaci modelu a import dat.
GraphX	Součástí komponenty „GraphX“ je knihovna, jenž obsahuje funkce pro vykreslování grafů a provádění paralelních grafových výpočtů. Tato komponenta rovněž obsahuje i funkční prvky pro ovládání grafů a knihovnu s běžnými grafickými algoritmy.

Využití v praktické části diplomové práce: Jak již bylo zmíněno v kapitole 4.2.1, v praktické části této diplomové práce bude pro manipulaci s daty využita služba *Data Bricks*. Služba *Data Bricks* je založena na nástroji *Spark*, přičemž hlavní rozšíření spočívá v možnosti pracovat s daty online, *Spark* je ve své tradiční verzi provozován na lokální infrastruktuře.

4.4 Python

Python je open-source interpretovaný programovací jazyk, jenž vznikl v roce 1991. V současné době se jedná o jeden z nejoblíbenějších programovacích jazyků v oblastech datové analytiky a strojového učení. Mezi přednosti tohoto programovacího jazyka patří jeho univerzální využití při tvorbě softwarových aplikací, přehledná struktura zapisovaného kódu a jednoduchost implementace rozšiřujících softwarových balíčků (5, str. 2).

Využití v praktické části diplomové práce: Programovací jazyk python, resp. jeho upravená verze *PySpark* (určená pro práci s nástrojem *Spark*), bude využita při manipulaci s daty a vytváření prediktivního modelu v rámci praktické části této diplomové práce. Python byl pro tento úkol vybrán pro jeho snadnou dostupnost, zkušenosti autora této práce s tímto programovacím jazykem a mnoha rozšiřujícím balíčkům, určeným pro zpracování dat a strojové učení. Zmíněné rozšiřující baličky a jejich popis je uvedený v tabulce č. 6.

Tabulka 6 Baličky a knihovny pythonu, zdroj: (autor)

Anglický název	Popis balíčku
Gensim	„Gensim je python knihovna určená pro tematické modelování, indexaci dokumentů a vyhledávání podobnosti ve velkých korpusech (56). Tato knihovna je určena pro experty z komunity zpracování přirozeného jazyka (NLP) a získávání informací (IR).“ V rámci praktické části této diplomové práce bude využita implementace algoritmu *Doc2vec* z této knihovny, pro převod dat z relační datové formy do numerické (vektorové) formy.
NumPy	*„NumPy, zkratka pro numerický python, je v pythonu základním balíčkem pro numerické výpočty. Poskytuje pythonu rozšíření podporovaných datových struktur a algoritmů, které jsou zapotřebí pro různé vědecké aplikace.“* (5, str. 4) V rámci praktické části této diplomové práce bude tato knihovna využita pro matematické operace a práci s vektory.
Pandas	*„Pandas poskytuje datové struktury a funkce, navržené pro rychlou, snadnou a flexibilní práci se strukturovanými nebo tabulkovými daty. Od vzniku v roce 2010 tento balíček pomohl pythonu stát se oblíbeným prostředím pro datovou analýzu.“* (5, str. 4) V rámci praktické části této diplomové práce bude tato knihovna využita pro analýzu, manipulaci a transformaci dat.
SciKit Learn	*„Open source knihovna SciKit-Learn poskytuje nástroje pro zpracování úloh strojového učení, přičemž obsahuje nástroje pro učení s učitelem (supervised) i učení bez učitele (unsupervised). Dále nabízí různé nástroje pro přizpůsobení modelu, předzpracování dat, výběr modelu a jeho evaluaci.“* (57, pozn: překladu autora) Z této knihovny bude využita řada nástrojů při zpracování a evaluaci prediktivního modelu, zejména nástroje pro model kNN.

Poznámka autora.: Tabulka č. 6 neposkytuje kompletní přehled knihoven využitých v rámci praktické části této práce, nicméně uvádí nejpoužívanější knihovny a jejich význam. Kromě zmíněných knihoven, bude využita i řada dalších knihoven, které jsou výchozí součástí balíku programovacího jazyku Python.

5 Analýza a transformace zdrojových dat

5.1 Struktura zdrojových dat

5.1.1 Data o uživatelské aktivitě

Jak již bylo zmíněno v kapitole 4.1, data jsou do úložiště na portále Azure streamována pomocí nástroje *Kafka*. Data jsou z nástroje *Kafka* streamována v reálném čase do *Blob Storage* úložiště dat na portálu Azure. Tato data jsou zde ukládána do souborů datového typu *parquet*, přičemž daný soubor je vytvořen a uložen po splnění jedné ze dvou podmínek:

- Počet řádků přesáhl stanovený počet řádků (v době psaní této práce 1000000 řádků).
- Proběhlo nastavené časové okno (v době psaní této práce 10 minut).

Každých 15 minut kontroluje nástroj *Data Factory*, jestli byl vytvořený nový datový soubor. Pokud nalezne tento datový soubor, spustí nad tímto souborem datovou pumpu s přednastavenými transformacemi (výběr relevantních dat, čištění dat, komprese dat) a uloží je do úložiště *Data Lake*. Po dokončení transformací a uložení dat, mají data strukturu odpovídající tabulce č. 7.

Tabulka 7 Struktura dat v úložišti *Data Lake*, zdroj: (autor)

Název	Datový tip	Popis atributu
internal_id	int	Automaticky generované ID pole identifikující událost vyskytující se v uživatelově relaci.
subscription_code	string	ID pole identifikující koncového uživatele.
message_type	string	Pole popisující typ vzniklé události. Může nabývat hodnot „start“ – počátek příjmu televizního pořadu uživatelem, „update“ – změna přicházející od uživatele (např. uživatel přepnul na jiný televizní kanál) a „stop“ – uživatel ukončil svou televizní relaci.
event_timestamp	int	Hodnota ve formátu *Unix time* zaznamenávající konkrétní čas, kdy k události na straně uživatele došlo.
edr_id	int	ID pole položky ve videotéce. Pro potřeby této diplomové práce není relevantní.
epg_id	int	ID pole popisující položku EPG.

source	string	Pole popisující zdroj datového záznamu. Může nabývat hodnot: „*Web*“ – webový prohlížeč, „*App*“ – mobilní aplikace nebo „*EKT* “ – druh set-top boxu. Po potřeby této diplomové práce jsou relevantní pouze data ze zdroje EKT, jelikož dosahují nejlepší kvality ze jmenovaných.
broadcast_type	string	Hodnota popisující způsob šíření televizního vysílání. Může nabývat hodnot *unicast* nebo *multicast*. Pro potřeby této diplomové práce není relevantní.
device_id	string	ID pole koncového zařízení. Pro potřeby této diplomové práce není relevantní.
device_detail	string	Pole poskytující detailnější informace o přijímacím zařízení (např. druh set-top boxu, Apple TV, herní konzole atd.)
isp_id	int	Pole identifikující internetového poskytovatele, přičemž toto pole nabývá dvou stavů „0“ – poskytovatel internetového připojení je jiný než poskytovatel IPTV, „1“ – poskytovatel internetového připojení je stejný jako poskytovatel IPTV.
channel_id	int	Identifikační číslo televizního kanálu.
channel_name	string	Název televizního kanálu.
event_type	string	Pole popisující, jestli se jedná o promítání „živého“ televizní vysílání, či o zpětné promítání. Může nabývat dvou hodnot: *linear* – příjem živého televizního vysílání, *nonlinear* – příjem zpětného televizního vysílání (z archivu).
ip_adress	string	Pole s IP adresou koncového uživatele. V této fázi už je tato hodnota anonymizována a pole má pouze hodnotu „null“.

Pozn.: Data jsou uložena v datových typech „*int*“ a „*string*“. Datový tip *integer* (zkráceně *int*) slouží pro ukládání pozitivních nebo negativních celých čísel, přičemž v programovacím jazyce python ve verzi 3 nemají nastavený limit velikosti (58). Datový tip *string* je určen pro ukládání textu ve formátu „*Unicode*“ v programovacím jazyku Python ve verzi 3 (59).

5.1.2 Data elektronického programového průvodce

Dříve zmíněný poskytovatel IPTV využívá pro získávání EPG dat externí společnost "TvProfi", jenž exporty EPG data poskytuje ve formě XML souborů. Exporty dat jsou dostupné ve dvou variantách:

- Po jednotlivých dnech pro jednotlivé televizní kanály.
- Kompletní export, pro všechny televizní kanály, jenž zaznamenává vždy data za předchozí den, současný den a plánovaná EPG na dalších deset dní dopředu.

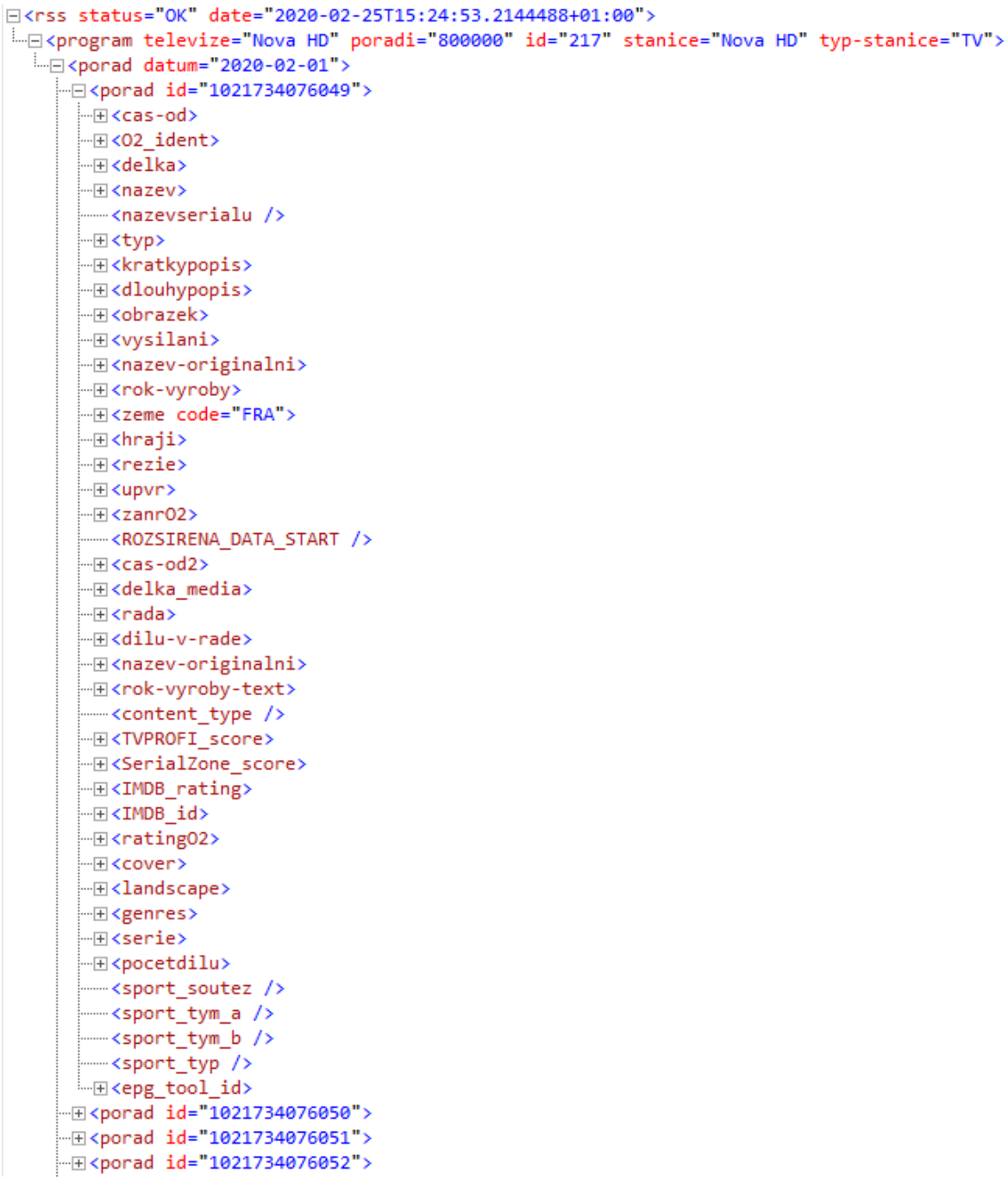
Pro potřeby této diplomové práce byly využity exporty dat po jednotlivých dnech, jelikož nebylo zapotřebí zpracovávat velké datové soubory, které obsahují nerelevantní informace pro tuto diplomovou práci (např. EPG data o televizních kanálech ze zahraničí jenž IPTV poskytovatel neposkytuje, EPG data o radiových pořadech atd.) Schéma jednodenního exportu ve formátu XML, je graficky znázorněno na obrázku č. 11.

V tuto chvíli je nutné zmínit, že ačkoliv poskytují datové exporty o EPG detailní informace, jsou zároveň velmi často nekonzistentní, a to jak ve své struktuře, tak i v úplnosti a kvalitě dat. Zmíněné exporty obsahují různorodá pole, jenž popisují daný televizní pořad jak z pohledu poskytovatele IPTV (např. id položky poskytovatele), tak z pohledu uživatele koncové služby (např. žánr pořadu).

Jak již bylo zmíněno, datová kvalita je u zmíněného extraktu různorodá, rozšiřující informace nejsou vždy vyplněny a nejsou dostupné pro všechny pořady. Velmi často jsou rovněž jednotlivá pole prázdná (zejména rozšiřující pole), takže neobsahují doplňující informace. K tomu dochází z různých důvodů, mimo jiné například:

- EPG poskytovatel nemá dostupná detailnější data o konkrétním pořadu.
- Rozšiřující pole, detailněji specifikující seriálové pořady, nejsou dostupná pro filmy, dokumenty a sportovní utkání a opačně.

Pro potřeby praktické části této diplomové práce jsou důležitá následující pole, která jsou až na výjimky dostupná pro většinu pořadů: Název televizní stanice (stanice), Počáteční čas vysílání pořadu (cas-od), Název pořadu (nazev), Typ pořadu (content-type), TvProfi rating (TVPROFI_score), IMDB rating (IMDB_rating) a žánry pořadu (genres). V případě, kdy zmíněná pole nejsou ve zdrojovém XML souboru dostupná, budou nahrazena textovou hodnotou „None", toto nahrazení zajišťuje zdrojový kód XML překladače, jenž je dále popsán v kapitole 5.2.3.



Obrázek 11 Schéma XML souboru obsahující EPG extrakt, zdroj: (autor)

5.2 Transformace dat

V rámci kapitoly 5.1.1 jsou popsána dostupná zdrojová data, včetně jejich struktury a formátu. Tato kapitola je zaměřena na popis transformačních aktivit, jež byly dále nad daty prováděny, mezi které patří například výběr relevantních sloupců, překlad z formátu XML do CSV, výpočet dodatečných sloupců a jiné. Dále jsou v rámci této kapitoly uvedena omezení, jenž slouží k výběru relevantních hodnot pro datový vzorek.

5.2.1 Omezení datového vzorku a datových zdrojů

Výchozí omezení

Jak již bylo zmíněno v úvodní kapitole, existují určitá omezení týkající se zdrojových dat, s nimiž bude pracováno v praktické části diplomové práce. V rámci tabulky č. 8 jsou uvedena omezení, s nimiž bylo v určité míře počítáno v době zadávání projektu diplomové práce. Následující subkapitola obsahuje rozšířená omezení, jenž se projevila v rámci průběhu zpracování diplomové práce.

Tabulka 8 Omezení datových zdrojů diplomové práce, zdroj: (autor)

Omezení	Odůvodnění
Omezení dat na konkrétní Set-top Box (ETK)	Jak již bylo zmíněno, IPTV poskytovatel poskytuje uživatelům svých služeb přístup z mnoha forem zařízení (set-top boxy, herní konzole, webové stránky a jiné), nicméně data ve vysoké kvalitě, jsou získávána pouze ze set-top boxů IPTV poskytovatele. Z tohoto důvodu byla pro praktickou část této diplomové práce vybrána data pouze z nejrozšířenějšího set-top boxu IPTV poskytovatele (ETK).
Kompletní identifikátor uživatele	Aby IPTV poskytovatel získal nové uživatele, pořádá různé promo akce a nabídky, kdy na časově omezenou dobu nabízí novým uživatelům své služby k vyzkoušení zdarma. V rámci toho jsou distribuovány promo kódy, které slouží k zaznamenávání id uživatele namísto standardního „subscription_code“, přičemž data o aktivitě jsou rovněž ukládána. Jelikož se jedná o krátkodobé, časově omezené uživatelské účty, nemá smysl pro ně predikovat zajímavé pořady.
Výběr datového vzorku za určité časové období	V rámci rozsahu této diplomové práce, budou data natrénována na datovém vzorku za určité časové období. K tomuto opatření bylo přistoupeno z důvodu rozsahu práce, ale i dalších důvodů (např. data o uživatelské aktivitě za prosinec 2019, by byla zkreslující, jelikož přes vánoční svátky sleduje většina domácností pohádky a jiné).
Výběr datového vzorku pro vybrané televizní kanály	V rámci rozsahu této diplomové práce, nelze zpracovávat data a vytvářet predikce pro všech 267 televizních / radiových kanálů, jenž poskytovatel IPTV svým klientům nabízí v rámci různých balíčků. Po konzultaci s odborníky ze společnosti IPTV poskytovatele, byl vybrán vzorek 25 televizních kanálů, kam byly zařazeny nejsledovanější televizní kanály, kanály parterů IPTV poskytovatele a kanály s dokumentárním obsahem.

	Mezi vybrané televizní kanály patří: AMC, Barrandov, ČT 1, ČT 2, ČT 24, Film box, Film box Premium, HBO, HBO 2, HBO 3, JOJ Cinema, Nova, Nova 2, Nova Cinema, Nova Gold, Prima, Prima Cool, Prima Zoom, Prima Krimi, Prima Love, Prima Max, Spektrum, Televize Seznam.
Omezení rádiových pořadů	IPTV poskytovatel poskytuje svým uživatelům i možnost přijímání radiových přenosů, skrze své produkty. Z pohledu této diplomové práce nejsou tato data relevantní (v teoretické rovině by se daly uživateli predikovat zajímavé radiové pořady, distribuce predikcí a doporučení by byla ovšem komplikovanější), přičemž z pohledu IPTV poskytovatele se nejedná o relevantní obchodní cíl, tudíž tato data nebudou v rámci praktické části využita.
Omezení predikcí obsahu pro dospělé	Big Data oddělení IPTV poskytovatele má zakázáno vytvářet modely na základě „sexuální, náboženské a politické orientace“ uživatele, pro kteroukoliv ze svých služeb. Z tohoto důvodu nebudou v rámci praktické části této diplomové práce, využita data o pořadech s obsahem pro dospělé.
Omezení zahraničních kanálů	IPTV poskytovatel nabízí svým uživatelům mnohé zahraniční televizní kanály (např. *CNN*, *Al Jazeera*, *BBC World News* a jiné). Data o uživatelské aktivitě z těchto kanálů nebudou využita, jelikož v mnoha případech je dostupných málo dat o uživatelské aktivitě a často jsou tato data získávána z hotelů a ubytovacích zařízení, kde nemá smysl predikovat zajímavé pořady.
Omezení kanálů se statickým obsahem	Kromě standardních televizních kanálů nabízí IPTV poskytovatel i televizní kanály se statickým obsahem (např. televizní vysílání obrazu s ohništěm, pláží a jiné) a různé informační kanály (např. informační kanál poskytovatele IPTV). Na základě těchto dat nelze predikovat relevantní a zajímavé pořady pro daného uživatele, tudíž tato data v této diplomové práci nebudou využita.
Omezení pořadů s “content_type = None”	EPG data poskytují kromě jiného i klasifikaci obsahu pořadu. V případě, kdy je tato hodnota prázdná/nevyplněná, je nahrazena hodnotou „None“, přičemž k tomu dochází zejména v případě pořadů o „Teleshoppingu“, „Přestávkách ve vysílání“ a jiných. O těchto pořadech nejsou dostupné detailní informace, tudíž tato data nebudou v rámci této diplomové práce využita.
Omezení pořadů s “content_type = zprávy”	EPG data poskytují kromě jiného i klasifikaci obsahu pořadu. V případě, kdy je tato hodnota rovna textu „zprávy“, jedná se o pořady jako jsou např. Odpolední počasí, Televizní noviny a jiné. V teoretické rovině by se uživatelům služeb IPTV poskytovatele daly predikovat tyto pořady jako zajímavé, ale z pohledu obchodních cílů IPTV poskytovatele se jedná o nerelevantní

	položky, tudíž tato data nebudou v rámci této diplomové práci využita.

Rozšířená omezení

Jak již bylo zmíněno v předchozí kapitole, v období zpracování (zejména praktické částí) této diplomové práce se objevila další omezení, s nimiž nebylo počítáno v době zadávání projektu diplomové práce. Tato diplomová práce byla zpracovávána v první polovině roku 2020, v době značného rozšíření takzvaného „coronaviru“ (COVID-19) v Evropě. Tato pandemie měla velký dopad na různá odvětví a společnosti, mimo jiné i na poskytovatele cloud prostředí Microsoft Azure. Z důvodu rozšíření zmíněného viru značně narostly požadavky na zdroje v prostředí Azure, kdy v některých datových centrech dané společnosti narostly požadavky na výpočetní zdroje (60). Například v datovém centru “*europe-west*” umístěném v Amsterdamu v Holandsku, jehož zdroje poskytovatel IPTV využívá, byl nárůst požadavků přes 700%.

Nedostatek výpočetních zdrojů značně ovlivnil i zpracování praktické části této diplomové práce, v určitých případech bylo nutné čekat na přidělení výpočetních zdrojů i několik dní a přiděleny byly pouze základní výpočetní jednotky (4 vCPU, 14 GB RAM), kde trénování Doc2vec modelů při větším počtu iterací trvalo 5 a více dnů. Aby bylo možné zpracování urychlit a umožnit odevzdání diplomové práce v požadovaném termínu, bylo nutné pro potřeby této diplomové práce omezit rozsah a zaměření datového vzorku.

Hlavním rozšiřujícím omezením je zaměření na typ obsahu „film“, přičemž ostatní typy obsahu „seriál“, „dokument“, „sport“, „živý přenos“ a „zábava“ budou zpracovány mimo tuto diplomovou práci, v době kdy budou výpočetní zdroje na platformě Azure dostupnější. Toto omezení datového vzorku, značně urychlilo trénování modelu Doc2vec pro generování numerické (vektorové) reprezentace dat.

5.2.2 Transformace dat o uživatelské aktivitě

Z původního zdrojového souboru o uživatelské aktivitě jsou vybrány řádky a sloupce, jenž obsahují relevantní informace pro tuto diplomovou práci. Tyto informace budou spolu s informacemi o jednotlivých pořadech, získaných z EPG dat v následující kapitole převedeny do numerické reprezentace při využití vektorů.

Výběr dat ze zdrojové tabulky byl proveden následujícím skriptem:

```
CREATE TABLE IF NOT EXISTS iptv_data(
    internal_id string,
    message_type string,
    event_timestamp bigint,
    channel_name string,
    epg_id double,
    subscription_code string,
    date integer
```

```
);

INSERT INTO iptv_data
SELECT internal_id, message_type, event_timestamp, channel_name, epg_id,
subscription_code, date from iptv_norm where source = "ETK";
```

Popis funkce skriptu: Skript v programovacím jazyce SQL, nejprve založí tabulku „*iptv_data*“ a následně do ní vloží data, jenž splňují podmínku, že datovým zdrojem byl set-top box ETK. Cílová tabulka „*iptv_data*“ obsahuje následující pole:

- *Internal_id* - identifikátor uživatelské relace
- *Subscription_code* - identifikátor uživatele
- *Message_type* - prováděná aktivita (start, update, stop)
- *Event_timestamp* - časový záznam ve formátu *“unix time”*
- *Epg_id* - identifikátor pořadu
- *Channel_name* - název televizního kanálu

Obrázek č. 12 poskytuje názornou ukázku zdrojových dat:

internal_id	message_type	event_timestamp	channel_id	epg_id	subscription_code	date
S_525532:portal-dmdproxy:1:231a7d6678d00c65f6f3b2aaa699a0d0	update	1581160338206	18	26175065	8000153735	20200208
S_705188:portal-dmdproxy:1:231a7d6678d00c65f6f3b2aaa699a0d0	start	1581160338207	18	26203440	8000854253	20200208
S_77767:portal-dmdproxy:1:231a7d6678d00c65f6f3b2aaa699a0d0	update	1581160338479	18	26203440	8000269416	20200208
S_100256:portal:1:231a7d6678d00c65f6f3b2aaa699a0d0	update	1581160336526	18	26203440	8000245622	20200208
S_281903:portal:1:231a7d6678d00c65f6f3b2aaa699a0d0	update	1581160338490	18	26203440	8000856821	20200208

Obrázek 12 Schéma tabulky obsahující data o uživatelské aktivitě, zdroj: (autor)

5.2.3 Transformace dat z elektronického programového průvodce

Extrakt dat z výchozích XML souborů

Transformace EPG dat byla ve srovnání s transformací dat o uživatelské aktivitě mnohem komplikovanější. Data nejsou poskytovatelem přímo ukládána do úložiště portálu Azure, a samo úložiště Azure nemá povolený přístup do internetu, tudíž nebylo možné nastavit přímou konektivitu. Z tohoto důvodu byla data do portálu Azure přenášena z externího serveru IPTV poskytovatele, jenž má speciálně nastavenou konektivitu. Zároveň na tomto serveru probíhala transformace ze zdrojových XML souborů do cílového souboru CSV, jelikož portál Azure ve výchozí konfiguraci neumí zpracovávat datové soubory ve formátu XML.

V rámci zmíněné transformace dat bylo nutné, vytvořit automatizovaný skript v programovacím jazyce Python, jenž zajistí vlastní překlad zdrojových souborů a transformaci dat. Tento skript je k této diplomové práci přiložen jako Příloha A. Postup automatizovaného skriptu je následující:

1. Uživatel zvolí televizní kanály, jejichž EPG budou zpracována, a žádoucí časový rozsah.
2. Skript vytvoří seznam obsahující datumy, v rámci zadaného časového období a v případě kdy zdrojová složka obsahuje soubory, vymaže je.
3. Skript stáhne zdrojové soubory ve formátu XML a uloží je do zdrojové složky.
4. Skript postupně zpracuje zdrojové XML soubory, vybere relevantní položky a zapíše je do jednoho textového souboru.
5. Následně jsou data ze zdrojového souboru importována do formátu *Data Frame* z python knihovny *Pandas* a jsou dopočítány a transformovány relevantní sloupce.
6. Výsledek je následně uložen do jednoho souboru ve formátu CSV.

Kromě jiných transformací jsou v rámci zmíněného skriptu dopočítávána následující pole:

- *start_timestamp* – Složení původních polí „*date*“ a „*start*_hr“ do textového pole a následný převod do datového formátu *„timestamp”*.
- *start_unix* – Vyjádření pole „start_timestamp“ ve formátu „*unix time*“.
- *lenght_unix* – Vyjádření původního pole „*length*“ ve formátu „*unix time*“.
- *end_unix* – Pole vznikne součtem polí „start_unix“ a „lenght_unix“, přičemž vyjadřuje čas konce pořadu ve formátu „*unix time*“.
- *end_timestamp* – Transformace pole „end_unix“ z formátu „*unix time*“ do datového formátu *timestamp*.

Poznámka.: Konkrétní informace a transformace dat je možné nalézt v příloze A této diplomové práce. Zdrojový kód obsahuje popisky a dokumentaci v anglickém jazyce. Konfiguraci skriptu na externím serveru nebylo možné v rámci této diplomové práce uvést z důvodu bezpečnosti, tudíž byly zdrojové složky a proměnné nahrazeny popisky jejich významu.

Data získaná ze souboru CSV jsou pomocí nástroje *Data Factory* přenesena do úložiště portálu Azure. Z tohoto úložiště je převezme nástroj *Databricks*, který pomocí následujícího kódu data přeloží a uloží do dočasné tabulky, jenž je následně přidána do tabulky s EPG daty:

```
# File location and type
file_location = "/FileStore/tables/data.csv"
file_type = "csv"

# CSV options
infer_schema = "false"
first_row_is_header = "true"
delimiter = ";"

# The applied options are for CSV files. For other file types, these will be ignored.
df = spark.read.format(file_type) \
  .option("inferSchema", infer_schema) \
  .option("header", first_row_is_header) \
  .option("sep", delimiter) \
```

```
  .load(file_location)

temp_table_name = "data_csv"

df.createOrReplaceTempView(temp_table_name)
```

Vytvořená tabulka obsahuje následující pole:

- Id - identifikátor pořadu poskytovatele EPG dat
- Date - datum vysílání pořadu
- Start_hr - čas počátku plánovaného vysílání
- Lenght - délka pořadu v minutách
- Type - typ obsahu (film, seriál, dokument, ...)
- Name - název pořadu
- Tvprofi - TvProfi hodnocení pořadu
- Imdb - IMDB hodnocení pořadu
- Genres - žánry pořadu
- Start_timestamp - čas počátku vysílání pořadu
- Start_unix - čas počátku vysílání pořadu v *Unix time*
- Lenght_unix - délka pořadu v sekundách
- End_unix - čas konce vysílání pořadu v *Unix time*
- End_timestamp - čas konce vysílání pořadu

Grafická ukázka výsledné tabulky je znázorněna na obrázcích č. 13 a č. 14.

id	date	start_hr	lenght	type	name	tvprofi	imdb
1021734076049	2020-02-01	05:55	10	None	Oggy a Škodíci V	77	73
1021734076050	2020-02-01	06:05	55	serial	Tlapková patrola III (15)	52	63
1021734076051	2020-02-01	07:00	50	serial	Looney Tunes: Nové příběhy (6)	65	58
1021734076052	2020-02-01	07:50	20	serial	Show Toma a Jerryho II (20)	61	69
1021734076053	2020-02-01	08:10	70	film	Kocour v botách	46	64
1021734076054	2020-02-01	09:20	110	film	Hořké sladkosti	54	58
1021734076055	2020-02-01	11:10	35	zábava	Volejte Novu	13	None
1021734076056	2020-02-01	11:45	80	zábava	Výměna manželek XII	32	None
1021734076057	2020-02-01	13:05	135	film	Město andělů	74	67

Obrázek 13 První část tabulky EPG dat od poskytovatele EPG, zdroj: (autor)

genre	start_timestamp	start_unix	lenght_unix	end_unix	end_timestamp
Komedie_Animovaný_Rodinný_	2020-02-01 05:55:00	1580536500	600	1580537100	2020-02-01 06:05:00
Akční_Komedie_Animovaný_Rodinný_	2020-02-01 06:05:00	1580537100	3300	1580540400	2020-02-01 07:00:00
Komedie_Animovaný_	2020-02-01 07:00:00	1580540400	3000	1580543400	2020-02-01 07:50:00
Komedie_Animovaný_Krátkometrážní_	2020-02-01 07:50:00	1580543400	1200	1580544600	2020-02-01 08:10:00
Fantasy_Romantický_Pohádka_	2020-02-01 08:10:00	1580544600	4200	1580548800	2020-02-01 09:20:00
Komedie_Romantický_	2020-02-01 09:20:00	1580548800	6600	1580555400	2020-02-01 11:10:00
Talk-show_	2020-02-01 11:10:00	1580555400	2100	1580557500	2020-02-01 11:45:00
Reality-TV_RealityShow_	2020-02-01 11:45:00	1580557500	4800	1580562300	2020-02-01 13:05:00
Drama_Fantasy_Romantický	2020-02-01 13:05:00	1580562300	8100	1580570400	2020-02-01 15:20:00

Obrázek 14 Druhá část tabulky EPG dat od poskytovatele EPG, zdroj: (autor)

Poznámka: Obrázky č. 13 a 14 jsou součástí jedné tabulky. Pro lepší zobrazení v tištěné formě diplomové práce byla tabulka rozdělena do dvou částí.

Propojení EPG dat s interním ID pořadu poskytovatele IPTV

Poskytovatel EPG dat využívá pro označení konkrétního pořadu generované „porad id" pole, jenž je spojené s konkrétním pořadem, jeho popisnými informacemi a plánovaným počátkem vysílání. Oproti tomu poskytovatel IPTV využívá pro označení pořadu vlastní „epg_id" pole, jenž v jeho systémech neurčuje konkrétní pořad, ale časový úsek, v němž bude daný pořad vysílán. Zde je nutno podotknout, že počátek a konec vysílání pořadu se může od přednastaveného plánu vysílání měnit, jelikož se průběžně mění i doba a počet reklamních sdělení, vysílaných jednotlivými televizními stanicemi.

Lze tedy říci, že v závislosti na tom, kolik reklamního času jednotlivé televizní stanice prodají, dochází k posunu času počátku a konce vysílání pořadu. Tyto změny se samozřejmě reflektují v datech, kdy vznikají rozdíly mezi plánovaným začátkem a koncem vysílání pořadu a reálnými hodnotami. Kromě již zmíněných posunů v čase, může rovněž televizní stanice provést dle potřeby změnu televizního programu a vysílat jiný obsah.

Vzhledem k tomu, že, jsou na obou stranách využívány jiné identifikátory pořadů a poskytovatel IPTV nebyl schopen v požadovaném čase poskytnou přístup do databáze, kde probíhá mapování/překlad hodnot identifikátorů, bylo nutné pro překlad id polí využít alternativní řešení. To spočívá v extrakci hodnot z alternativního datového zdroje IPTV poskytovatele, z databáze Teradata, kde jsou data využívána zejména pro generování různých reportů BI oddělením poskytovatele IPTV.

Data byla z alternativního datového zdroje exportována ve formátu CSV, který byl pomocí nástroje *Data Factory* přenesen do úložiště portálu Azure. Z tohoto úložiště jej převzal nástroj *Databricks*, který pomocí následujícího kódu data přeložil, uložil do dočasné tabulky, a následně je přidal do tabulky s EPG daty poskytovatele IPTV:

```
# File location and type
file_location = "/FileStore/tables/export_epg.csv"
file_type = "csv"

# CSV options
infer_schema = "false"
first_row_is_header = "true"
delimiter = ","

# The applied options are for CSV files. For other file types, these will be ignored.
df = spark.read.format(file_type) \
  .option("inferSchema", infer_schema) \
  .option("header", first_row_is_header) \
  .option("sep", delimiter) \
  .option("encoding", "Cp1250") \
  .load(file_location)
```

```
temp_table_name = "export_epg_csv"

df.createOrReplaceTempView(temp_table_name)
```

Data mají po uložení do tabulky následující strukturu:

- Epg_id - Id pořadu poskytovatele IPTV
- Channel_name - Název televizního kanálu
- Start_time - Čas počátku vysílání pořadu
- End_time - Čas konce vysílání pořadu
- Extract_dttm - Čas uložení dat do původní databáze

Výsledná tabulka EPG dat poskytovatele IPTV je graficky znázorněna na obrázku č. 15.

epg_id	channel_name	start_time	end_time	extract_dttm
26235753	nova 2 hd	2020-02-12 06:25:00	2020-02-12 06:50:00	2020-02-10 00:00:00
22898409	prima hd	2019-01-14 19:55:00	2019-01-14 20:15:00	2019-11-20 00:00:00
22610536	čt2 hd	2018-12-12 21:00:00	2018-12-12 21:30:00	2019-11-20 00:00:00
25608232	prima hd	2019-11-29 06:25:00	2019-11-29 06:50:00	2019-11-27 00:00:00
25574830	prima love hd	2019-11-26 04:00:00	2019-11-26 06:15:00	2019-11-24 00:00:00
19757786	čt1 hd	2018-02-08 12:30:00	2018-02-08 14:00:00	2019-11-20 00:00:00
25836275	čt1 hd	2019-12-27 20:00:00	2019-12-27 21:25:00	2019-12-25 00:00:00
20830259	prima max hd	2018-05-28 07:30:00	2018-05-28 08:00:00	2019-11-20 00:00:00
25380658	prima love hd	2019-10-24 22:48:31	2019-10-27 19:48:31	2019-11-20 00:00:00
19082711	nova hd	2017-11-29 18:25:00	2017-11-29 19:30:00	2019-11-20 00:00:00
25169785	čt1 hd	2019-10-10 10:35:00	2019-10-10 11:30:00	2019-11-20 00:00:00

Obrázek 15 Tabulka s EPG daty od poskytovatele IPTV, zdroj: (autor)

Propojení EPG dat z obou datových zdrojů

V této fázi existují tedy dvě tabulky s EPG daty, přičemž jedna poskytuje data od EPG poskytovatele a druhá data od IPTV poskytovatele. Tyto dvě tabulky byly následně propojeny pomoci následujícího skriptu, přičemž vznikla jedna souhrnná tabulka:

```
CREATE TABLE epg_database AS
SELECT B.epg_id, A.id, A.name, A.type, A.genre, A.tvprofi, A.imdb, A.start_unix as
start_planned, A.end_unix as end_planned, A.avg_length
FROM epg_edw B
LEFT JOIN epg_tvp A
ON A.channel_name = B.channel_name and (A.start_timestamp = B.start_time or
A.end_timestamp = B.end_time);
```

Nově vzniklá tabulka obsahuje následující hodnoty:

- Epg_id - Id pořadu poskytovatele IPTV
- Id - Id pořadu poskytovatele EPG
- Název - Název pořadu
- Typ - Typ obsahu (film, seriál, dokument, ...)
- Genre - Žánry pořadu
- Tvprofi - TvProfi hodnocení pořadu
- Imdb - IMDB hodnocení pořadu
- Start_planned - Plánovaný čas počátku vysílání pořadu
- End_planned - Plánovaný čas konce vysílání pořadu
- Avg_length - 40% celkové délky pořadu (vyjádřené počtem sekund)

6 Numerická reprezentace dat

Z pohledu úloh strojového učení dělíme datové proměnné na kategoriální a kvantitativní (numerické). Pokročilé statistické modely, zaměřené na úlohy zpracování přirozeného jazyka, dosahují nejlepších výsledků při zpracování dat v kvantitativní (numerické) formě.

Z tohoto důvodu je žádoucí vyjádřit data o uživatelské aktivitě a podrobnosti o sledovaných pořadech v numerické formě. K tomu bude využita metoda nazývaná odborným pojmem „*Embeddings*“ (v anglickém originále). Tato kapitola je zaměřena na základní charakteristiku metody *Embeddings*, popis vlastní implementace v praktické části této diplomové práce a ověření funkčnosti (tzv. „*Sanity Check*“) zvolené metody.

6.1 Charakteristika embeddings

Webový portál *Towards Data Science* definuje pojem „*Embeddings*“ následovně:

„Embedding je vyjádření diskrétní – kategorické – proměnné na vektor spojitých čísel. V kontextu neuronových sítí jsou embeddingy nízko rozměrné, naučené, spojité vektorové reprezentace diskrétních proměnných. Embeddingy z neuronových sítí jsou užitečné, jelikož umožnují snížit rozměrnost (dimenzionalitu) kategorických proměnných a umožnují smysluplnou reprezentaci v transformovaném prostoru.“ (61, pozn.: překlad autora)

Mezi primární využití embeddings patří mimo jiné:

- Nalezení nejbližšího souseda (podobné položky) ve vektorovém prostoru.
- Vstupní data pro algoritmy strojového učení pro úlohy s učitelem.
- Vizualizace konceptů a vztahů mezi kategoriemi.

Embeddings vznikají jako vedlejší produkt při trénování neuronových sítí, kdy v procesu učení přiřadí neuronová síť kategorickým hodnotám numerickou (vektorovou) hodnotu. V programovacím jazyce Python existuje řada knihoven určených pro strojové učení, přičemž řada z nich umožnuje vytváření a trénování neuronových sítí (např. *Tensorflow*, *NLTK, SpaCy, Keras, Pytorch, Gensim* a jiné), každá z těchto knihoven využívá vlastní implementaci při vytváření embeddingů v závislosti na typu a algoritmu neuronové sítě.

6.2 Implementace embeddings

V rámci praktické části této diplomové práce bude pro vytvoření embeddings využita knihovna *Gensim* a její implementace algoritmu *Doc2vec*. Tato knihovna byla vybrána jelikož poskytuje jednoduchou a přehlednou implementaci algoritmu *Doc2vec*, která umožnuje jednotlivým dokumentům přiřadit náhodně vygenerované nebo uživatelem

definované „Tag ID“, jenž identifikuje daný dokument. V případě této práce jsou namísto dokumentů využity věty, jenž obsahují identifikaci uživatele, pořadu, typu pořadu a žánru.

Gensim implementace algoritmu *Doc2vec* ovšem bere při trénování modelu v potaz pouze kategorické (textové) hodnoty, proto byly numerické identifikátory uživatele a pořadu využity jako „Tag ID“, jenž identifikuje unikátní kombinaci uživatele sledujícího daný pořad. V rámci této implementace jsou využity pouze unikátní výskyty uživatele sledujícího pořad, aby nedošlo k ovlivňování natrénovaného modelu opětovným výskytem záznamů.

6.2.1 Příprava a popis datového vzorku

Souhrnné informace o vybraném datovém vzorku jsou uvedeny v tabulce č. 9.

Tabulka 9 Charakteristika datového vzorku pro Doc2vec embeddings, zdroj: (autor)

Parametr datového vzorku	Hodnota
Časové období datového vzorku	2. 3. 2020 až 15. 3. 2020
Počet unikátních záznamů o uživatelské aktivitě	2 738 507
Počet unikátních uživatelů	134 749
Počet unikátních pořadů	3091
Datový zdroj	EKT
Typ obsahu	Film
Počet unikátních žánrů	33

Prvním krokem při přípravě embeddings je připravit datový soubor, v podporovaném datovém formátu s definovanou strukturou dat. Data jsou zpracována pomocí skriptu v příloze B, kde dochází k výběru relevantních polí ze zdrojových tabulek, transformaci vybraných hodnot a uložení ve formátu CSV do úložiště *Data Lake*. V rámci daného zdrojového kódu jsou vytvářeny dva CSV soubory, přičemž v této fázi bude zapotřebí pouze jeden z nich a to soubor „*user_activity.csv*“. Všechny transformace jsou zpracovávány v nástroji *Databricks*. Výsledný CSV soubor má následující strukturu:

Identifikátor zákazníka, identifikátor pořadu, typ pořadu, žánry pořadu

Ukázka z výsledného CSV souboru je graficky znázorněna na obrázku č. 16.

```
8000372871,26371462,film,Drama Rodinný Sportovní
8000598741,26358829,film,Dobrodružný Drama Fantasy Rodinný
```

Obrázek 16 Grafické znázornění datového vzorku pro embeddings, zdroj: (autor)

6.2.2 Implementace modelu Doc2vec

Jak již bylo zmíněno v předchozích subkapitolách, vektorové reprezentace dat bylo dosaženo využitím algoritmu *Doc2vec* implementovaného v knihovně *Gensim*. Modelu *Doc2vec* lze nastavit vybrané parametry, jejich kompletní seznam je dostupný na webových stránkách s dokumentací knihovny *Gensim* (62), přičemž v rámci této implementace byly nastaveny parametry uvedené v tabulce č. 10.

Tabulka 10 Parametry modelu Doc2vec, zdroj: (autor)

Parametr	Význam	Hodnota
Corpus file	Textový soubor obsahující zdrojová data pro trénování modelu.	user_ activity.csv
Vector size	Parametr definující dimenzionalitu (počet) vektorů.	100
Window	Nejdelší vzdálenost mezi vybraným a predikovaným slovem v rámci věty.	5
Alpha	Parametr definující počáteční míru učení algoritmu.	0.025
Min_alpha	Parametr definující nejnižší míru učení algoritmu, přičemž míra učení se v rámci trénování lineárně snižuje z hodnoty „*Alpha*“ na hodnotu „*Min_alpha*“.	0.0025
Min_count	Minimální počet výskytu daného slova, pokud je míra výskytu nižší slovo bude při trénování modelu ignorováno.	25
Dm	Výběr trénovacího algoritmu (1 = PV-DM, 0 = PV-DBOW).	1
Workers	Parametr definující počet CPU, paralelně využitých při trénování.	8
Epochs	Počet iterací (průchodů) při trénování nad datovým vzorkem.	100

V tuto chvíli je nutno poznamenat, že tabulka č. 10 obsahuje pouze finální verzi parametrů natrénovaného modelu Doc2vec, jenž optimálně zachycuje charakter dat. V rámci ověření vhodnosti využití algoritmu Doc2vec, byl model natrénován s různými nastaveními konkrétních parametrů, přičemž bylo experimentováno zejména s následujícími parametry:

- *Epochs* – Algoritmus Doc2vec je svým primárním účelem určen pro zpracování nestrukturovaných dat, kdy doporučovaný počet epoch (čili iterací) nad vzorkem dat je 10 – 20. Nicméně v rámci praktické části této diplomové práce jsou data uložena ve strukturované formě, proto bylo zapotřebí navýšit počet iterací, s postupným snižováním míry učení algoritmu. Model s vyšším počtem iterací lépe zachycuje (v rámci embeddings) vazby/váhy mezi jednotlivými žánry.

- *Window* – V rámci dat, jenž jsou pomocí algoritmu Doc2vec zpracovávána, má každý pořad slovně definován žánr, přičemž každý pořad definuje 1 – 7 názvů žánru. Parametr *window* zachycuje nejdelší vzdálenost mezi vybraným žánrem a predikovaným slovem v rámci věty. Během experimentování se jako optimální nastavení parametru projevilo pět slov, jelikož vice či méně slov hůře zachycovalo vazby/váhy mezi jednotlivými žánry.

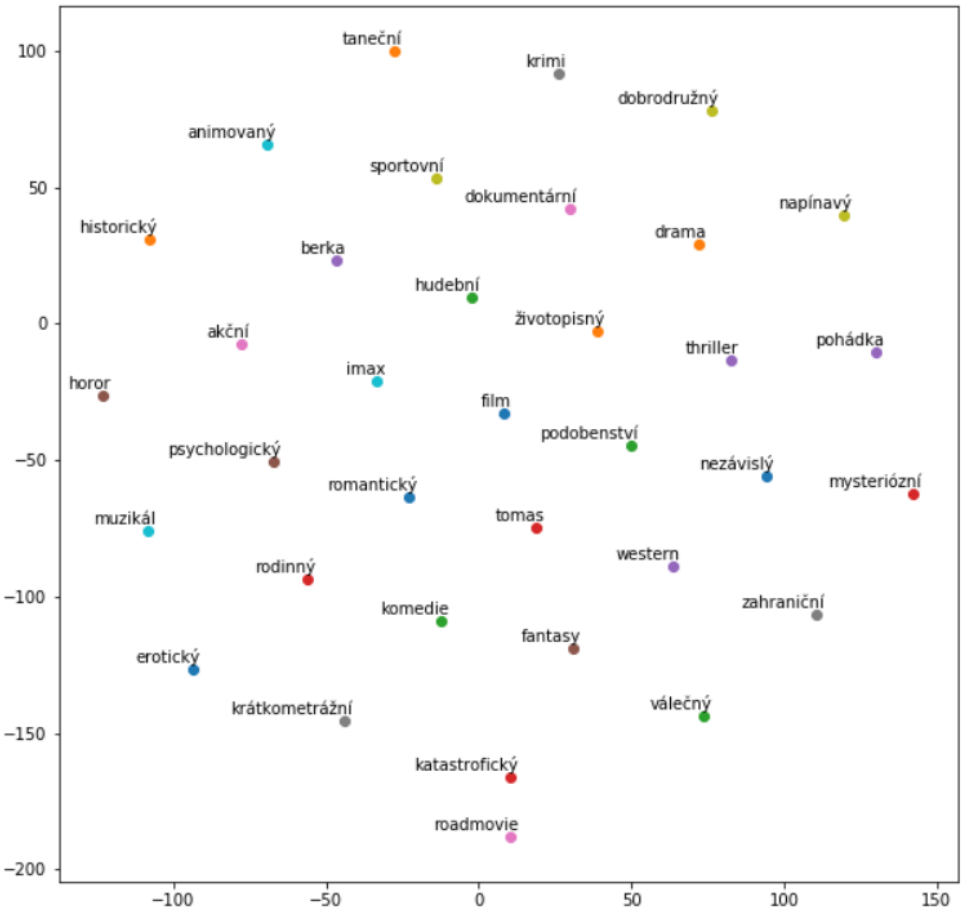
Zdrojový kód zmíněné implementace je dostupný v příloze C, přičemž vlastní implementace je provedena v nástroji *Machine Learning Service*. Z tohoto důvodu jsou v rámci zdrojového kódu, uvedeny i funkce pro zaznamenávání parametrů ve zmíněném nástroji. Postup zdrojového kódu lze charakterizovat následovně:

1. Načtení zdrojového souboru z uložiště *Data Lake*.
2. Zpracování dat ze zdrojového CSV souboru a vytvoření takzvaného „corpusu“, jenž pro Doc2vec udržuje strukturu dat.
3. Následně je vytvořen Doc2vec model s definovanými parametry, přičemž tomuto modelu je následně předán „corpus“ a je vytvořena knihovna modelu.
4. Je spuštěno trénování modelu, jenž proběhne v počtu iterací definovaným v rámci parametrů modelu.
5. Natrénovaný model je uložen.
6. Následně je pro každého uživatele z modelu extrahován vektor, který je uložen do výsledného CSV souboru.

Poznámka autora.: Zdrojový kód v příloze C, obsahuje doporučenou metodu pro čtení řádků v rámci „*corpus file*“ definovanou v (63), jenž je doporučena pro algoritmus Doc2vec v dokumentaci knihovny Gensim

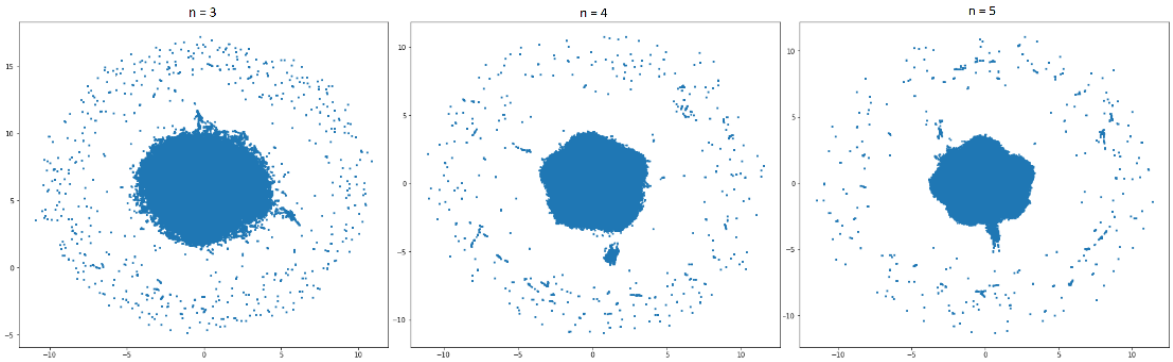
Z vytvořeného modelu lze na základě identifikátorů uživatele (*subscription_code*) a pořadu (*epg_id*) následně načíst vektorové vyjádření jednotlivých uživatelů a televizních pořadů. Podkladový model Word2vec (jenž slouží jako základ modelu Doc2vec) obsahuje celkem 33 unikátních vektorů, jenž numericky vyjadřují jednotlivé žánry. Grafické znázornění vektorového prostoru podkladového Word2vec modelu lze nalézt na obrázku č. 17, přičemž je na obrázku zachyceno všech 33 žánrů (včetně popisku), ve vektorovém prostoru.

Centrálním prvkem na daném obrázku je slovo „film“, jenž je obsaženo ve všech záznamech ve zdrojových datech pro trénování algoritmu Doc2vec. Ostatní žánry jsou v daném prostoru umístěny na základě vazeb/vah, naučených při trénování modelu. Vzájemné vazby/váhy jednotlivých vektorů nelze jednoduše interpretovat, jelikož se je model učí v rámci fáze trénování, přičemž konkrétní odůvodnění vzájemných vazeb mezi vybranými položkami nelze z modelu extrahovat.



Obrázek 17 Vizualizace vektorů v modelu word2vec, zdroj: (autor)

Model Doc2vec následně obsahuje celkem 139 132 unikátních vektorů (unikátní vektor pro každého uživatele a každý televizní pořad). Grafické znázornění vektorového prostoru Doc2vec modelu lze nalézt na obrázku č. 18, přičemž daný obrázek slouží pouze pro představu vektorového prostoru. Při zmíněném počtu vektorů, nelze v rámci této práce zobrazit popisek každého vektoru, jako je tomu v případě obrázku č. 17. Jednotliví uživatelé a pořady, jsou ve vektorovém prostoru umístěni na základě vazeb/vah, naučených při trénování Doc2vec modelu.



Obrázek 18 Vizualizace vektorů v modelu doc2vec, zdroj: (autor)

Jak je patrné z výše uvedeného obrázku, klíčovou roli při vykreslování vektorového prostoru hraje parametr „n“, jenž stanovuje počet nejbližších sousedů, které jsou v rámci trénování algoritmu *UMAP* brány v potaz. Stejně tak jako je tomu v případě obrázku č. 17, ani zde není možné interpretovat/odůvodnit konkrétní umístění vektorů v rámci vektorového prostoru, jelikož vazby a váhy jednotlivých vektorů jsou uloženy v rámci doc2vec modelu, ze kterého je není možné extrahovat.

Zdrojový kód v programovacím jazyce Python, jenž slouží pro vygenerování obrázků č. 17 a č. 18, za pomoci knihoven *SciKit-Learn* a *UMAP* je uvedený v příloze D. Zdrojový kód pro vykreslení obrázku č. 17 obsahuje metodu definovanou v (64), jenž byla upravena pro potřeby této práce.

6.3 Sanity Check – ověření funkčnosti

Velmi důležitou součástí při vytváření embeddings je takzvaný „Sanity Check“, čili ověření funkčnosti vlastního řešení tak aby bylo potvrzeno, že model negeneruje pouze sekvenci náhodných čísel. Tato kontrola se provádí na menším vzorku dat, kdy není zapotřebí velký výpočetní výkon a samotné trénování modelu netrvá dlouho.

V případě této implementace, byl datový vzorek omezen pouze na vybrané televizní kanály (Nova HD, Prima HD, ČT 1 HD), za časové období jednoho dne (1.3.2020), na několik specificky vybraných uživatelů (celkem 6 uživatelů - vybraných na základě vizuální kontroly aktivity v systémech poskytovatele IPTV). Na tomto datovém vzorku byl natrénován Doc2vec model, a následně ověřena správnost výsledků. Ověření výsledků bylo provedeno na základě metriky „*cosine similarity*“.

V kontextu strojového učení je metrika *cosine similarity* využívána zejména pro ověření míry podobnosti mezi jednotlivými položkami (dokumenty). V matematickém kontextu je tato metrika měřena kosinem úhlu mezi dvěma nenulovými vektory, a určuje zda dva vektory směřují zhruba stejným směrem.

6.3.1 Implementace Sanity Check

Zdrojový kód implementace Sanity Check se neliší od vlastní implementace algoritmu Doc2vec uvedeném příloze C této diplomové práce. Následně je natrénován model Doc2vec s níže uvedenými parametry:

- *Vector size* = 100
- *Window* = 5
- *Alpha* = 0,025
- *Min_alpha* = 0,00025
- *Min_count* = 3
- *Workers* = 8
- *Epochs* = 100

Popis atributů algoritmu Doc2vec z knihovny Gensim lze nalézt v kapitole 6.2.2. Implementace a průběh trénování modelu je stejný, jako u již zmíněné implementace Doc2vec. Rozdíl mezi implementacemi spočívá v natrénování modelu na menším datovém vzorku, jenž obsahuje kromě filmů i jiné typy pořadů. Oproti původní implementaci, jsou v tomto případě využity další funkce knihovny *Gensim*, přičemž se jedná primárně o funkci „*most_similar*“. Funkce „*most_similar*“ slouží k nalezení nejbližších položek, ve vektorovém prostoru k předem stanovené položce, přičemž samotný výpočet je založen na principu „*cosine similarity*“.

6.3.2 Evaluace Sanity Check

Jak již bylo zmíněno, evaluace vytvořeného modelu bude prováděna výpočtem metriky „*cosine similarity*“ pro vybrané parametry. Pomocí funkce „*most_similar*“ implementované v knihovně *Gensim*, budou nalezeny nejbližší položky v rámci vektorového prostoru. Zmíněná funkce zároveň ověří vzdálenost mezi vybranými ukazateli. Konkrétně se jedná o vzdálenost mezi následujícími ukazateli:

- Klíčová slova
- Skupiny uživatelů a pořadů

Cílem vlastní evaluace vytvořeného Doc2vec modelu není ověřit vlastní „správnost“ modelu, nýbrž ověřit, že je model schopen při trénování zachytit vazby, mezi jednotlivými klíčovými slovy (žánry), uživateli a pořady, a tyto vazby promítnout do výsledných vektorů. Promítnutí zmíněných vazeb do vzniklých vektorů ověříme blízkostí jednotlivých položek, v rámci vektorového prostoru, za pomoci metriky „*cosine similarity*“.

Klíčová slova

V první fázi jsou ověřena vybraná slova, z podkladového Word2vec modelu, přičemž za slova jsou v tomto případě považovány jednotlivé žánry pořadu. Například mezi nejbližší slova ke klíčovému slovo „akční“ by měla patřit slova „film“, „serial“, „krimi“, „dobrodružný“, a jiné. V rámci této kontroly není zapotřebí, přesně definovat nejbližší slova k vybranému slovu, postačí zde pouze vizuální ověření, že mezi nejbližší slova nepatří nevhodná spojení (například k již zmíněnému slovu „akční“ nejsou ve vektorovém prostoru nejblíže umístěny slova jako „dokumentární“, „talkshow“ a jiné.)

```
print('Closest vectors to akční:')
print(model.wv.most_similar('akční',topn=3))

print('Closest vectors to dobrodružný:')
dokument_vec = model.wv.most_similar('dobrodružný',topn=3)
print(dokument_vec)

print('Closest vectors to film:')
fiml_vec = model.wv.most_similar('film',topn=3)
print(fiml_vec)

print('Closest vectors to dokumentární:')
zabava_vec = model.wv.most_similar('dokumentární',topn=3)
print(zabava_vec)

print('Closest vectors to western:')
zabava_vec = model.wv.most_similar('western',topn=3)
print(zabava_vec)
```

```
Closest vectors to akční:
[('film', 0.9607947468757629), ('komedie', 0.9433221817016602), ('serial', 0.9367822408676147)]
Closest vectors to dobrodružný:
[('western', 0.9735305309295654), ('film', 0.9476935863494873), ('fantasy', 0.9408849477767944)]
Closest vectors to film:
[('komedie', 0.9771045446395874), ('akční', 0.9607948660850525), ('animovaný', 0.9554643630981445)]
Closest vectors to dokumentární:
[('dokument', 0.9941428899765015), ('publicistický', 0.9294097423553467), ('zábava', 0.8433785438537598)]
Closest vectors to western:
[('dobrodružný', 0.9735305309295654), ('akční', 0.9361445903778076), ('film', 0.9136016964912415)]
```

Obrázek 19 Sanity Check – Ověření klíčových slov, zdroj: (autor)

Obrázek č. 19 poskytuje grafické znázornění zdrojového kódu ověření, včetně výstupu z konzole. Jak je z výstupu patrné, natrénovaný model celkem dobře zachycuje vztahy mezi klíčovými slovy. Například ke klíčovému slovu „film“ mají nejblíže slova „komedie“, „akční“ a „animovaný“, což je na základě dostupných dat o pořadech reálné. Ovšemže ne vždy jsou výsledky optimální, například ke klíčovému slovu „dokumentární“, mají nejblíže slova „dokument“ a „publicistický“ což je v pořádku, ale třetím nejbližším slovem je „zábava“, což v daném kontextu moc nedává smysl.

Zde je ovšem nutné podotknout, že natrénovaný model je při trénování podkladového Word2vec modelu, značně ovlivněn malým datovým vzorkem. Při této kontrole je cílem pouze ověřit, že model je schopen zachytit závislosti mezi jednotlivými klíčovými slovy, což daný výstup potvrzuje.

Skupiny uživatelů

Ve druhé fázi bude provedena kontrola vzdáleností mezi vybranými skupinami uživatelů. Vybrané uživatele lze rozdělit do dvou skupin. První skupina uživatelů sleduje v rámci vybraného dne pouze filmy, druhá skupina uživatelů sleduje různorodý televizní obsah (filmy, seriály, dokumenty). Obě skupiny mají celkem 3 členy, přičemž aktivita uživatelů sledujících pouze filmy se nachází pouze na televizním kanále „Nova HD“, druhá skupina uživatelů sleduje televizní pořady ze všech tří televizních kanálů.

V rámci evaluace modelu bude ověřeno, že funkce „*most_similar*“ nalezne k vybranému uživateli nejblíže uživatele ze stejné skupiny uživatelů. Tudíž v případě, kdy vybereme uživatele z první skupiny uživatelů, ve vektorovém prostoru budou k tomuto uživateli nejblíže umístěni zbylí dva uživatelé z první skupiny.

V této diplomové práci, bohužel není možné z bezpečnostních důvodů a z důvodu regulace GDPR, uvést konkrétní příklady dat o uživatelské aktivitě. Z tohoto důvodu byla aktivita jednotlivých uživatelů ověřena v rámci systémů poskytovatele IPTV a pro potřeby této práce byly vybrány skupiny uživatelů s podobným chováním. Skupiny uživatelů jsou uvedeny v tabulce č. 11, přičemž skupina 1 obsahuje uživatele sledující filmy a skupina 2 uživatele, kteří sledují různorodý obsah.

Tabulka 11 Testovací skupiny uživatelů

Skupina 1	Skupina 2
8000309543	8000862793
8000113451	8000474222
8000667780	8000192501

Obrázek č. 20 poskytuje grafické znázornění zdrojového kódu ověření vzdálenosti mezi uživateli, včetně výstupu z konzole.

```
print('Closest user to series user:')
user = model.docvecs.most_similar('8000474222', topn = 3)
print(user)

print('Closest user to movies user:')
user2 = model.docvecs.most_similar('8000309543', topn = 3)
print(user2)
```

```
Closest user to series user:
[('8000192501', 0.9885459542274475), ('8000667780', 0.9634235501289368), ('8000113451', 0.9619981050491333)]
Closest user to movies user:
[('8000113451', 0.9459495544433594), ('8000862793', 0.9441484212875366), ('8000667780', 0.9406505823135376)]
```

Obrázek 20 Sanity Check – Ověření skupin uživatelů, zdroj: (autor)

V prvním případě patří zvolený uživatel (8000474222) do druhé skupiny uživatelů, sledujících různorodý obsah. První nejbližší uživatel patří rovněž do druhé skupiny uživatelů, druhý a třetí nejbližší uživatel patří do první skupiny uživatelů, sledujících primárně filmy. V druhém případě je vybrán uživatel (8000309543) z první skupiny, sledující převážně filmy. Stejně tak jako v předchozím případě, první nejbližší uživatel spadá do stejné skupiny jako zvolený uživatel, ovšemže druhý a třetí nejbližší uživatel spadají do opačné skupiny.

Stejně tak, jako je tomu u ověření klíčových slov, vytvořený model je negativně ovlivněn malým datovým vzorkem, tudíž získané výsledky mají nízkou vypovídací hodnotu. Ovšemže pro potřeby našeho ověření (model v rámci vektoru zachycuje aktivitu specifického uživatele, uživatelé s podobnou aktivitou jsou ve vektorovém prostou umístěni blíže), tento model splnil očekávání.

Skupiny pořadů

Třetí fáze bude podobná druhé fázi této evaluace, s tím rozdílem, že namísto kontroly vzdálenosti skupin uživatelů, bude provedena kontrola vzdáleností mezi vybranými skupinami pořadů. Vybrané pořady lze rozdělit do dvou skupin. První skupina pořadů jsou filmy, druhá skupina pořadů jsou seriály. Implementace kontroly a výstup z konzole je graficky znázorněn na obrázku č. 21.

```
print('Closest user to specified series:')
film = model.docvecs.most_similar('26385314', topn = 3)
print(film)

print('Closest user to specified movies:')
film2 = model.docvecs.most_similar('26384509', topn = 3)
print(film2)
```

```
Closest user to specified series:
[('26385323', 0.9977172613143921), ('26385322', 0.9976073503494263), ('26385315', 0.9966093301773071)]
Closest user to specified movies:
[('26384483', 0.9948762059211731), ('26385327', 0.8691651225090027), ('26385320', 0.6292577981948853)]
```

Obrázek 21 Sanity Check – Ověření skupin pořadů, zdroj: (autor)

V prvním případě jsou vyhledány nejbližší pořady ke zvolenému pořadu (26385314), přičemž se jedná o seriály. Jak je patrné z obrázku č.22, vybraný pořad je epizoda seriálu „Simpsonovi“, přičemž nejbližší pořady ke zvolenému pořadu jsou další díly tohoto seriálu stejné série.

```
select * from epg_x where epg_id in ('26385314','26385323','26385315')
```

▸ (3) Spark Jobs

epg_id	id	date	channel_name	name	type	genre
26385315	1021935023883	2020-03-01	prima cool hd	Simpsonovi XV (13/22)	serial	Komedie Animovaný Rodinný
26385323	1021935023891	2020-03-01	prima cool hd	Simpsonovi XV (17/22)	serial	Komedie Animovaný Rodinný
26385314	1021935023882	2020-03-01	prima cool hd	Simpsonovi XV (12/22)	serial	Komedie Animovaný Rodinný

Obrázek 22 Sanity Check – Výpis skupiny pořadů: Seriály, zdroj: (autor)

V druhém případě jsou vyhledány nejbližší pořady ke zvolenému pořadu (26384509), přičemž se jedná o filmy. Obrázek č. 23, poskytuje detaily o jednotlivých pořadech. Zvolený pořad je film „Tenkrát na východě“, přičemž jako nejbližší pořady jsou vybrány filmy „Tenkrát na východě“ (druhotné/noční promítání filmu) a „Sedm statečných“ (jenž má podobné žánrové zaměření).

```
select * from epg_x where epg_id in ('26384509','26384483','26385327')
```

(3) Spark Jobs

epg_id	id	date	channel_name	name	type	genre
26385327	1021935023895	2020-03-01	prima cool hd	Sedm statečných	film	Akční Dobrodružný Western
26384509	1021735021534	2020-03-02	nova hd	Tenkrát na východě	film	Akční Dobrodružný Komedie Western
26384483	1021735021528	2020-03-01	nova hd	Tenkrát na východě	film	Akční Dobrodružný

Obrázek 23 Sanity Check – Výpis skupiny pořadů: Filmy, zdroj: (autor)

V obou skupinách bylo ověřeno, že žánrově podobné pořady jsou ve vektorovém prostoru umístěny blízko sebe, tudíž model splňuje požadavky zmíněného ověření.

Závěr evaluace Sanity Check

V rámci jednotlivých fází evaluace vytvořeného Doc2vec modelu bylo ověřeno, že zmíněný model je ve vytvořených vektorech schopen zachytit vazby mezi jednotlivými klíčovými slovy (žánry), uživateli a pořady. Tyto vazby byly ověřeny za pomoci metriky „*cosine similarity*“.

Pro potřeby praktické části této diplomové práce je tedy Gensim implementace algoritmu *Doc2vec* vyhovující, přičemž bude natrénován komplexní Doc2vec model (kapitola 6.2) na větším datovém vzorku a vektory odvozené z modelu, budou následně využity při vytváření prediktivního modelu.

7 Vývoj prediktivního modelu

Tato kapitola je zaměřena na popis vývoje prediktivního modelu, jenž je hlavním výstupem této diplomové práce. Natrénovaný model (nad embeddings daty z předchozí kapitoly), by měl na základě uživatelské aktivity (dříve sledovaných pořadů) predikovat zajímavý obsah pro daného uživatele. První část této kapitoly je zaměřena na popis a obecnou charakteristiku modelů kNN a KDTree. Druhá část popisuje implementaci daného modelu v rámci praktické části této práce. Evaluace natrénovaného modelu je uvedena v kapitole č. 8.

7.1 Charakteristika kNN a KDTree

Algoritmus modelu „*k nearest neighbor*“ (dále jen kNN), je založen na podobnosti jednotlivých datových bodů. Daný algoritmus při zpracování datových bodů předpokládá výskyt podobných datových bodů v těsné blízkosti.

Pojem kNN lze definovat různě, přičemž webový server *techopedia.com* jej definuje následovně (65, pozn.: překlad autora):

„Algoritmus „k-nejbližší soused“, často přezdíván kNN, je přístup ke klasifikaci dat, který odhaduje pravděpodobnost zařazení datového bodu do jedné ze skupin, na základě blízkosti bodu k jednotlivým skupinám.“

Algoritmus kNN může být využit jak pro prediktivní úlohy klasifikace, tak i regrese. Klíčovým faktorem, při trénování modelu kNN je vhodný výběr parametru „k“, jenž určuje počet „nejbližších sousedů“ branných v potaz při trénování algoritmu. Tento parametr je zvolen na základě charakteru úlohy, kterou má natrénovaný model plnit. Obecně platí, že modely s nižší hodnotou „k“ dosahují lepších výsledků/vyšší přesnosti.

V kontextu této diplomové práce, bude daný algoritmus využit k identifikaci hodnoty „k“, jenž poskytne (v rámci zpracovávaného datového vzorku) nejvyšší přesnost natrénovaného modelu. Rozsah hodnoty „k“ je v stanoven od 3 do 30 nejbližších sousedních uživatelů. K tomuto rozsahu bylo přistoupeno z povahy úkolu, jenž má výsledný prediktivní model plnit, jelikož jeho cílem je získat prvních 3 – 5 doporučených pořadů s nejvyšším hodnocením. Výběr větší hodnoty „k“ by pouze prodloužil dobu trénování modelu a navýšil požadavky na výpočetní zdroje při trénování modelu.

Algoritmus modelu „*k Dimensional Tree*“ (dále jen KDTree), je podobně jako je tomu u algoritmu kNN založen na podobnosti jednotlivých datových bodů. Implementace daného algoritmu v knihovně *SciKit Learn* je určena pro rychlé zpracování úloh s N datovými body (57). V rámci této diplomové práce, bude algoritmus KDTree využit k získání nejbližších „k“ identifikátorů sousedů pro zvoleného uživatele.

7.2 Implementace prediktivního modelu

7.2.1 Příprava datového vzorku

Datový vzorek vychází ze stejných parametrů, jako je tomu u datového vzorku uvedeného v kapitole 6.2.1 v tabulce č. 9. V tomto případě mají data pozměněný charakter, přičemž jsou rozdělena do tří souborů. Zpracování dat je prováděno v nástroji *Databricks* a výsledné soubory jsou uloženy ve formátu CSV do úložiště *Data lake*. Zdrojový kód pro generování prvního a druhého datového souboru je uveden v příloze B.

První datový soubor (user_activity.csv) je stejný jako datový soubor zpracovávaný v kapitole 6.2.1. Z původního datového souboru byly zachovány první dva sloupce: identifikátor zákazníka a identifikátor pořadu. Tento soubor využívá algoritmus kNN k zařazení zákazníků do shluků na základě jejich aktivity. Datový soubor má níže uvedenou podobu, přičemž jeho ukázka je graficky znázorněna na obrázku č. 24.

Identifikátor zákazníka, identifikátor pořadu

```
8000113451,26384471
8000192501,26385316
```

Obrázek 24 Grafické znázornění prvního datového souboru, zdroj: (autor)

Druhý datový soubor (epg_data.csv) zachycuje identifikátory všech pořadů, vysílaných v rámci zvoleného časového období, a vybranou metriku hodnocení daného pořadu (v tomto případě TvProfi hodnocení pořadu). Tento soubor bude využit v rámci seřazení doporučených pořadů, na základě jejich hodnocení. Datový soubor má níže uvedenou podobu, přičemž ukázka je graficky znázorněna na obrázku č. 25.

Identifikátor pořadu, hodnocení pořadu

```
26385318,87
26385327,65
```

Obrázek 25 Grafické znázornění druhého datového souboru, zdroj: (autor)

Třetí datový soubor je výstupem z natrénovaného Doc2vec modelu z kapitoly 6.2. Daný soubor uvádí identifikátor zákazníka a vektor, jenž charakterizuje jeho aktivitu. Tento datový soubor bude využit v rámci algoritmu KDTree, k umístění zákazníka do vektorového prostoru a nalezení nejbližších sousedících zákazníků. Datový soubor má níže uvedenou podobu, přičemž ukázka je graficky znázorněna na obrázku č. 26.

Identifikátor zákazníka; vektory (oddělené středníkem)

```
8000113451;-0.723847;0.42100266;-0.042009536;-0.35912275;-0.5170
```

Obrázek 26 Grafické znázornění třetího datového vzorku, zdroj: (autor)

7.2.2 Implementace kNN a KDTree

Pro potřeby této diplomové práce byla zvolena implementace algoritmů kNN a KDTree v knihovně *SciKit-Learn*. Zdrojový kód implementace tohoto projektu je dostupný v příloze E této diplomové práce, přičemž vlastní implementace je provedena v nástroji *Machine Learning Service*. Z tohoto důvodu jsou v rámci zdrojového kódu uvedeny i funkce pro zaznamenávání parametrů ve zmíněném nástroji.

Postup zdrojového kódu lze charakterizovat následovně:

1. Načtení zdrojových souborů z úložiště *Data Lake* do formátu *Data Frame*.
2. Natrénování modelu KNN s následujícími kroky:
    a. Rozdělení dat na popisné sloupce (*identifikátor zákazníka*) a hodnotové sloupce (*identifikátor pořadu*).
    b. Rozdělení dat na trénovací a testovací množiny (trénovací 70%, testovací 30%).
    c. Převedení hodnotového sloupce do standardizovaného formátu (rozptylu) využitím funkce „*StandardScaler*“ z knihovny *SciKit-Learn*.
    d. Definice rozsahu hodnoty „k“ (počtu nejbližších sousedících uživatelů).
        i. Z důvodu velikosti datového vzorku a počtu unikátních uživatelů, jsou hodnoty „k“ nastaveny v rozsahu od 3 do 20 nejbližších uživatelů. Při velikosti vzorku cca 135 000 unikátních uživatelů model kNN dosahuje relativně nízké přesnosti, nicméně vzhledem k tomu, že prediktivní model by měl generovat 3 – 5 doporučených pořadů, větší počet sousedících uživatelů by zbytečně zabíral výpočetní zdroje.
    e. Pro každou hodnotu „k“ je natrénován kNN model a vypočítána přesnost modelu.
    f. Číslo „k“ s nejvyšší přesností je následně předáno algoritmu KDTree.
3. Natrénování modelu KDTree, jenž obsahuje následující kroky:
    a. Nalezení počtu „k“ nejbližších uživatelů vybranému uživateli.
    b. Transformace dat do formátu *Data Frame*, jenž pro každého uživatele obsahuje identifikátory nejbližších uživatelů.
4. Do datové struktury slovník jsou pro každého uživatele uloženy pořady, které v daném časovém období zhlédl.
5. Predikce zajímavých pořadů pro každého uživatele, tu lze charakterizovat v následujících krocích:
    a. Pro každého uživatele jsou zaznamenány pořady, jenž shlédli jeho nejbližší sousedé.
    b. Z dat jsou odstraněny duplicity a pořady, které již uživatel shlédl (na základě dat z kroku č. 4).
    c. Data jsou seřazena na základě TvProfi hodnocení pořadů (hodnoty jsou srovnatelné s ČSFD hodnocením pořadů).
    d. Výsledná predikce je spolu s identifikátorem zákazníka uložena do CSV souboru.
6. Soubor s predikcemi je nástrojem *Data Factory* přenesen do úložiště *Data Lake*.

Obrázek č. 27 obsahuje ukázku výstupu z konzole z kroku č. 5 při generování doporučených pořadů pro jednotlivé uživatele.

```
--- Preparing recommendations ---
8000113451 [26385315, 26385323, 26368687, 26385327, 26368689, 26385322, 26381311, 26385316, 26376272, 26376408, 26368305, 26381350,
8000192501 [26376408, 26384479, 26376442, 26376409, 26384471, 26384470, 26374611, 26374633, 26385418, 26384482, 26376420, 26374612,
8000474222 [26384483, 26374633, 26384484, 26374634, 26376420, 26384509, 26384482, 26376442, 26384481, 26376409, 26384470, 26384471,
8000862793 [26385323, 26385328, 26385325, 26385349, 26385324, 26381389, 26385322, 26381387, 26382931, 26385320, 26376272, 26385317,
8000667780 [26385320, 26384484, 26382931, 26385324, 26385323, 26385325, 26385322, 26385328, 26385318, 26385327, 26376272, 26385317,
8000309543 [26385315, 26385323, 26368687, 26385327, 26368689, 26385322, 26385316, 26381389, 26376272, 26368305, 26381350, 26381311,
```

Obrázek 27 Ukázka výstupu z konzole při generování doporučených pořadů

Výsledný CSV soubor tedy obsahuje doporučení zajímavých pořadů pro vybranou skupinu uživatelů.

Řešení vytvořené v rámci kapitol 6 a 7 této práce, lze považovat za prediktivní model, jenž na základě uživatelské aktivity predikuje uživatelům zajímavý a pro uživatele relevantní obsah. Vytvořený prediktivní model je hlavním výstupem této diplomové práce, přičemž je schopen plnit úkoly stanovené hlavním cílem této diplomové práce. Evaluace vytvořeného modelu je rozebrána v kapitole č. 8.

7.2.3 Produkcionalizace modelu

Jak již bylo zmíněno ve třetí kapitole této diplomové práce, fáze „*deployment*“ čili produkcionalizace metodiky CRISP-DM není součástí této diplomové práce, nicméně je vhodné přiblížit, jak je vytvořený model následně využíván.

Získaná data jsou zpracovávána v online platformě, kterou využívá marketingové oddělení IPTV poskytovatele. Ze souboru s predikcemi jsou pro vybrané zákazníky extrahovány seznamy doporučených pořadů. Z těchto seznamů jsou následně vyřazeny pořady, které již není možné zpětně zhlédnout (opatření k zamezení doporučování pořadů, jenž si již uživatel nemůže přehrát). Pro každého konkrétního uživatele, je následně vybráno prvních 3 – 5 pořadů s nejvyšším hodnocením (počet vybraných pořadů volí marketingové oddělení IPTV poskytovatele).

Důvodů, proč je výsledný model implementován do produkčního prostředí na straně marketingového oddělení IPTV, je hned několik:

- Marketingové oddělení má k dispozici neanonymizovaná data o uživatelských účtech, tudíž na jejich straně bude probíhat distribuce e-mailů obsahující doporučení a „*upselling*“ služeb.
- Marketingové oddělení má k dispozici data o poskytovaných službách/balíčcích IPTV a VoD, a může lépe cílit „*upselling*“ služeb na konkrétní uživatele.
- Rozšiřující data o konkrétních pořadech (např. jestli je možné vybraný pořad zpětně zhlédnout, jestli pořad spadá do prémiového segmentu atd.) jsou dostupná pouze v online platformě marketingového oddělení.

Data jsou do online platformy marketingového oddělení přenášena nástrojem *Data Factory*, z úložiště *Data Lake* na SFTP server online platformy.

8 Evaluace modelu

Cílem této kapitoly je ověřit funkčnost a správnost implementace prediktivního modelu, jenž vzniklo jako hlavní výstup praktické části této diplomové práce. Ověřována je tedy přesnost natrénovaných modelů a korektní fungování prediktivního systému. Tuto kapitolu lze rozdělit do dvou částí: evaluace podpůrných modelů a evaluace rekomendačního algoritmu.

8.1 Evaluace podpůrných modelů

Prvním evaluovaným modelem je Doc2vec model, jenž je natrénován v kapitole č. 6. Jak již bylo ve zmíněné kapitole uvedeno, z pohledu praktické části této diplomové práce vlastní funkčnost daného doc2vec modelu není pro tuto práci relevantní. Potřebným výstupem jsou pouze vektory jednotlivých uživatelů, extrahovaných z daného modelu, jenž zaznamenávají aktivitu uživatele.

Jak již bylo ověřeno v kapitole č. 6.3.2, natrénovaný model Doc2vec je schopen zachytit aktivitu stanoveného uživatele v rámci vektoru, a ve vektorovém prostoru nalézt uživatele s podobnou aktivitou. Pro správné natrénování modelu a získání vektorů je zapotřebí při trénování uvést kompletní záznamy o aktivitě uživatelů. Z tohoto důvodu nebylo možné data rozdělit na trénovací a testovací vzorek. Vzhledem k tomu že zamýšlené využití modelu bylo ověřeno v kapitole 6.3.2 a data nebylo možné rozdělit na trénovací a testovací množiny, model nebude v této práci dále evaluován.

Druhým evaluovaným modelem je kNN model, jenž byl natrénován při různých hodnotách „k“ v kapitole č. 7. Data jsou při trénování modelu rozdělena na trénovací a testovací množiny (poměr 70%/30%). Relevantní metrikou, na jejímž základě je následně doporučena hodnota „k“ pro zpracování modelu KDTree, je metrika „*accuracy_score*“, jenž je implementována v knihovně *SciKit Learn*. Metrika *accuracy_score* slouží k určení přesnosti klasifikace vybrané podmnožiny hodnot (66). Obrázek č. 28 obsahuje grafické zobrazení výstupu z konzole při trénování modelu kNN, jenž uvádí hodnotu „k“ a dosažený výsledek *accuracy_score*.

Zde je nutno poznamenat, že výsledná přesnost modelu kNN je velmi nízká, což je zapříčiněno výběrem nízké hodnoty „k“ počtu nejbližších sousedů (algoritmus je vyhodnocován pro 3 až 30 nejbližších sousedů na datovém vzorku) na poměrně velkém vzorku dat (cca 135 000 unikátních uživatelů). Z pohledu praktické části této diplomové práce je ovšem zapotřebí získat pouze několik prvních predikcí pro každého uživatele, které mu budou distribuovány v rámci emailového newsletteru. Výběr větší hodnoty „k“ by pouze prodloužil dobu trénování modelu a navýšil požadavky na výpočetní zdroje při trénování modelu.

```
--- Loading user vectors data ---
--- Loading user activity data ---
--- Training KNN ---
K value: 4 , with accuracy: 3.651620771879599e-06
K value: 5 , with accuracy: 3.651620771879599e-06
K value: 6 , with accuracy: 7.303241543759198e-06
K value: 7 , with accuracy: 7.303241543759198e-06
K value: 8 , with accuracy: 7.303241543759198e-06
K value: 9 , with accuracy: 6.086034619799332e-06
K value: 10 , with accuracy: 6.086034619799332e-06
K value: 11 , with accuracy: 4.868827695839465e-06
K value: 12 , with accuracy: 4.868827695839465e-06
K value: 13 , with accuracy: 4.868827695839465e-06
K value: 14 , with accuracy: 4.868827695839465e-06
K value: 15 , with accuracy: 4.868827695839465e-06
K value: 16 , with accuracy: 4.868827695839465e-06
K value: 17 , with accuracy: 2.4344138479197324e-06
K value: 18 , with accuracy: 1.2172069239598662e-06
K value: 19 , with accuracy: 1.2172069239598662e-06
K value: 20 , with accuracy: 2.4344138479197324e-06
K value: 21 , with accuracy: 2.4344138479197324e-06
K value: 22 , with accuracy: 2.4344138479197324e-06
K value: 23 , with accuracy: 1.2172069239598662e-06
K value: 24 , with accuracy: 1.2172069239598662e-06
K value: 25 , with accuracy: 1.2172069239598662e-06
K value: 26 , with accuracy: 1.2172069239598662e-06
K value: 27 , with accuracy: 1.2172069239598662e-06
K value: 28 , with accuracy: 0.0
K value: 29 , with accuracy: 2.4344138479197324e-06
K value: 30 , with accuracy: 2.4344138479197324e-06
{4: 3.651620771879599e-06, 5: 3.651620771879599e-06,
Number of neighbours chosen from KNN: 6
--- Training KDTree ---
--- Selecting neighbors for each user ---
```

Obrázek 28 Výstup z konzole, trénování modelu kNN, zdroj: (autor)

Jak je z obrázku patrné, nejvyšší přesnosti dosáhl model při hodnotě „k“ = 6 s přesností 0.000007303241543759198. Hodnota „k“ bude následně předána modelu KDTree, jenž pro každého uživatele určí „k“ (v tomto případě 6) nejbližších sousedících uživatelů. Z pohledu této diplomové práce, je tedy model kNN evaluován na základě metriky *accuracy_score*.

8.2 Evaluace rekomendačního algoritmu

Jak již bylo zmíněno v kapitole č.2.4.3, evaluaci rekomendačního systému (i algoritmu) lze rozdělit na dvě části: *Metrics Based Evaluation* a *Human Based Evaluation*. V tuto chvíli, je zapotřebí uvést, že evaluace vytvořeného algoritmu je značně ovlivněna prostředím, kde dané řešení vznikalo. Marketingové oddělení, v době odevzdání této diplomové práce, distribuuje ze své online platformy emaily doporučeními pouze omezené skupině uživatelů. Z důvodu bezpečnosti nebylo možné získat identifikátory konkrétních zákazníků

(marketingové oddělení má data o zákaznících v neanonymizované formě), pro ověření, jestli daný uživatel doporučený pořad shlédl.

Metody *Metrics Based Evaluation* a *Human Based Evaluation* jsou primárně založeny na zpětné kontrole, jestli zvolený uživatel jednal na základě doporučení (v případě této práce je to shlédnutí doporučeného pořadu) a jestli jsou pro něj daná doporučení relevantní (na základě hodnocení získaného od uživatele). Vzhledem k tomu, že v době psaní této diplomové práce není možné z již zmíněných důvodů využít standardizované metody evaluace rekomendačních systémů, soustředí se tato subkapitola na ověření funkčnosti vlastního modelu. Tudíž bude v rámci této kapitoly ověřeno, že vybraný uživatel daný pořad ve stanoveném časovém rozsahu neshlédl, a doporučené pořady jsou seřazeny na základě hodnocení TvProfi.

Pro další evaluaci je zapotřebí zvolit si reprezentativní skupinu uživatelů, přičemž byly zvoleni uživatelé: 8000735276, 8000342489 a 8000533310. Ukázka doporučených pořadů pro dané uživatele, je uvedena na obrázku č. 29. Zmíněné kontroly budou provedena v nástroji *Databricks*.

```
8000735276,[26438660, 26334214, 26402320, 26402322, 26384405,
8000342489,[26438657, 26438658, 26438659, 26438660, 26438661,
8000533310,[26434314, 26410771, 26410010, 26410011, 26410014,
```

Obrázek 29 Grafické znázornění doporučených pořadů pro vybrané uživatele

Doporučení uživatele 8000735276

Mezi doporučené pořady pro uživatele 8000735276 patří následující pořady: 26438660, 26334214, 26402320, 26402322 a 26384405. Jak je patrné z obrázku č. 30, uživatel ve zvoleném časovém období neshlédl žádný z doporučených pořadů.

Cmd 1

```
1 select count(*) from o2tv_norm_ext where subscription_code = '8000735276'
2 and (date between '2020-03-02' and '2020-03-15')
3 and epg_id in ('26438660','26334214','26402320','26402322','26384405');
```

▸ (2) Spark Jobs

count(1)
0

Obrázek 30 První uživatel - Kontrola shlédnutí doporučených pořadů

Doporučení uživatele 8000342489

Mezi doporučené pořady pro uživatele 8000342489 patří následující pořady: 26438657, 26438658, 26438659, 26438660 a 26438661. Jak je patrné z obrázku č. 31, uživatel ve zvoleném časovém období neshlédl žádný z doporučených pořadů.

Cmd 4

```
1 select count(*) from o2tv_norm_ext where subscription_code = '8000342489'
2 and (date between '2020-03-02' and '2020-03-15')
3 and epg_id in ('26438657','26438658','26438659','26438660','26438661');
```

▸ (2) Spark Jobs

count(1)
0

Obrázek 31 Druhý uživatel - Kontrola shlédnutí doporučených pořadů

Doporučení uživatele 8000533310

Mezi doporučené pořady pro uživatele 8000342489 patří následující pořady: 6434314, 26410771, 26410010, 26410011 a 26410014. Jak je patrné z obrázku č. 32, uživatel ve zvoleném časovém období neshlédl žádný z doporučených pořadů.

Cmd 6

```
1 select count(*) from o2tv_norm_ext where subscription_code = '8000533310'
2 and (date between '2020-03-02' and '2020-03-15')
3 and epg_id in ('26434314','26410771','26410010','26410011','26410014');
```

▸ (2) Spark Jobs

count(1)
0

Obrázek 32 Třetí uživatel - Kontrola shlédnutí doporučených pořadů

Ověření pořadí záznamů

Doporučené pořady pro každého uživatele jsou seřazeny na základě hodnocení TvProfi (hodnocení je srovnatelné s hodnotami na ČSFD). Obrázek č. 32 uvádí identifikátory pořadů, názvy a jejich TvProfi hodnocení, jenž byly doporučeny uživateli 8000342489. Jak je názorně uvedeno na obrázku č. 29, danému uživateli byly doporučeny pořady: 26434314, 26410771, 26410011, 26410011 a 26410014 v tomto pořadí. V rámci obrázku je ověřeno, že pořady jsou seřazeny na základě hodnocení TvProfi (od nejvyššího k nejnižšímu hodnocení).


Cmd 7

```
1 select epg_id, name, tvprofi from epg_x where epg_id in ('26434314','26410771','26410010','26410011','26410014') order by tvprofi;
```

▸ (1) Spark Jobs

epg_id	name	tvprofi
26434314	Kolja	86
26410771	Aquaman	72
26410010	Báječný Mike	62
26410011	Hannibal - Zrození	58
26410014	Romance z továrny	38

Obrázek 33 Kontrola seřazení doporučených pořadů

8.3 Zhodnocení evaluace

V rámci této kapitoly byla ověřena zejména technická funkčnost rekomendačního algoritmu. Z důvodu nedostatku dat bohužel nebylo možné evaluovat rekomendační algoritmus standardními metodami pro evaluaci rekomendačních systémů (*Metrics Based Evaluation* a *Human Based Evaluation*), jak již bylo zmíněno v kapitole 8.2.

Marketingového oddělení IPTV poskytovatele schválilo výstup praktické části této diplomové práce a nasadilo vytvořený prediktivní model do produkčního prostředí ve své online platformě. Data získaná z prediktivního modelu budou využita v rekomendačním systému IPTV poskytovatele. Daný rekomendační systém byl do současné chvíle založen pouze na rekomendaci obsahu doporučeného partnery IPTV poskytovatele (zejména od společnosti HBO). Nyní bude rozšířen i o individualizované doporučení pořadů pro koncové uživatele. Získaná doporučení budou rozesílána jako emailový newsletter s doporučenými pořady pro každého uživatele.

Marketingové oddělení dále navrhlo rozšíření prediktivního modelu na základě dalších parametrů pořadů (rok výroby, země původu, hlavní herci a jiné.). Další informace jsou uvedeny v závěrečné kapitole této práce. Je ovšem nutné zmínit, že před samotným rozšířením modelu, bude zapotřebí spolupráce s EPG poskytovatelem dat, jelikož dostupná EPG data nejsou v současné podobě dostatečně kvalitní ke zpracování zmíněných rozšíření.

Závěr

Rostoucí míra digitalizace a technologického rozvoje značně mění mnohá zavedená odvětví. Nejinak je tomu i na trhu s audiovizuálním obsahem. IPTV a VoD služby se těší mezi běžnými uživateli veliké oblibě, což je znát i na růstu počtu subjektů poskytujících zmíněné služby v minulých letech. Aby tito poskytovatelé získali konkurenční výhodu, snaží se své služby zlepšovat a nabízet uživatelům vyšší přidanou hodnotu. V mnohých případech jsou dnes koncoví uživatelé zmíněných služeb zahlceni množstvím obsahu, který mají k dispozici, přičemž mají problém vybrat si obsah, který je pro ně zajímavý a relevantní. Nejinak je tomu i u uživatelů jednoho z největších českých poskytovatelů IPTV služeb. Z tohoto důvodu se zmíněný poskytovatel rozhodl vytvořit rekomendační systém obsahu pro uživatele svých služeb.

Hlavním cílem této diplomové práce bylo vytvořit prediktivní model, jehož součástí bude takzvaný rekomendační algoritmus, který generuje doporučení pořadů pro koncové uživatele. Výsledná doporučení byla implementována do rekomendačního systému v online platformě marketingového oddělení IPTV poskytovatele, který distribuuje doporučené pořady koncovým uživatelům ve formátu emailového newsletteru. Vedlejším cílem této diplomové práce byla teoretická charakteristika vybraných algoritmů, postupů a technologií, na úrovni potřebné k pochopení konceptů a aktivit prováděných v praktické části diplomové práce.

Teoretická část této diplomové práce je zaměřena primárně na splnění vedlejšího cíle této práce. Prvním krokem bylo provedení rešerše o tématech souvisejících s rekomendačními systémy a rekomendačními algoritmy. Kromě klasických zdrojů, jako jsou například knižní publikace a odborné články, bylo zapotřebí provést i rešerši různých technologických webových serverů. Vzhledem k tomu že oblast strojového učení a rekomendačních systémů se neustále vyvíjí, knižní publikace často obsahují neaktuální informace. Na základě provedené rešerše byla zpracována druhá kapitola, jenž je zaměřena zejména na technologie související s praktickou částí této práce. Mimo jiné charakterizuje zmíněná kapitola technologie a oblasti technologií jako jsou například: IPTV, VoD, Data Mining, Zpracování přirozeného jazyka, Word2vec, Doc2vec a Rekomendační systémy. Třetí kapitola je soustředěna na popis metodiky CRISP-DM, pomocí které byla zpracována praktická část této diplomové práce. Je zde uveden popis jednotlivých fází dané metodiky a dále jsou uvedeny kapitoly související s jednotlivými fázemi vybrané metodiky. V rámci páté kapitoly byl proveden popis vývojového prostředí, ve kterém byla zpracována praktická část této práce. Mimo jiné je v této kapitole uveden popis technologií, služeb a nástrojů, jako jsou například: Apache Kafka a Spark, Microsoft Azure a Python. Na základě zpracování zmíněných kapitol lze považovat vedlejší cíl této práce za splněný, jelikož byly popsány všechny technologie, koncepty, služby a nástroje, využívané v praktické části diplomové práce.

Praktická část je zaměřena na splnění hlavního cíle této práce, čili vytvoření prediktivního modelu, jenž bude pro vybrané uživatele predikovat zajímavý obsah (pořady ke zhlédnutí). Prvním krokem ke splnění tohoto cíle byla identifikace zdrojů dat, výběr relevantních dat,

jejich zpracování a následná transformace do požadované podoby. Tyto aktivity byly popsány v páté kapitole, přičemž v rámci stejné kapitoly byla popsána i omezení vztahující se na výběr datového vzorku, jenž byl využit pro vytvoření prediktivního modelu. V rámci šesté kapitoly byla provedena transformace datového vzorku z relační databázové formy do numerické (vektorové) reprezentace dat, jelikož algoritmy a modely strojového učení dosahují lepších výsledků při zpracování kvantitativních (numerických) dat. Tato transformace byla provedena pomocí implementace algoritmu Doc2vec knihovny Gensim, přičemž v rámci dat byla zaznamenána aktivita vybrané skupiny uživatelů za stanovené časové období. Rovněž byl v rámci této kapitoly proveden takzvaný Sanity Check, jenž ověřil že zmíněná implementace algoritmu Doc2vec v knihovně Gensim je schopna zachytit uživatelovu aktivitu ve vektorové formě, a uživatelé s podobnou aktivitou jsou ve vektorovém prostoru umístěni blíže k sobě. Osmá kapitola je zaměřena na popis vývoje prediktivního modelu (včetně rekomendačního algoritmu), přičemž v rámci této kapitoly jsou charakterizovány algoritmy kNN a KDTree založené na principu „*nearest neighbor*“, který je založen na podobnosti jednotlivých bodů a jejich umístění v prostoru. Ve stejné kapitole byl dále popsán princip fungování prediktivního modelu, jenž je hlavním výstupem této diplomové práce, a obecný popis produkcionalizace zmíněného modelu, přičemž vlastní programový kód modelu je dostupný v příloze E. Jak již bylo v rámci textu práce zmíněno, detaily produkcionalizace daného modelu není možné uvést z bezpečnostních důvodů. Na základě vytvořeného modelu lze pro vybraného uživatele predikovat doporučené pořady ke zhlédnutí. V osmé kapitole této práce byla provedena evaluace vzniklého prediktivního modelu a rekomendačního algoritmu. Jak již bylo v rámci kapitoly popsáno, z bezpečnostních důvodů nebylo možné k evaluaci využít klasické metody evaluace rekomendačních systémů (*Metrics Based Evaluation* a *Human Based Evaluation*), jelikož dané metody by vyžadovaly přístup k neanonymizovaným datům koncových uživatelů. Namísto toho byla evaluována technická funkčnost daného modelu, přičemž bylo ověřeno že uživatel doporučené pořady v rámci daného časového období neshlédl a doporučení jsou seřazena na základě hodnocení TvProfi (obdobné hodnocení jako ČSFD). Evaluovaný model byl shledán za funkční, tudíž byl splněn hlavní cíl této práce, vytvoření prediktivního modelu pro doporučování obsahu zákazníkům IPTV poskytovatele.

Je ovšem nutné podotknout, že hlavní cíl této práce byl splněn s určitými omezeními. Jak již bylo zmíněno v kapitole 5.2.1, tato práce byla zpracovávána v období výskytu viru COVID-19 v Evropě. Kromě dalších neblahých vlivů, měla pandemie negativní dopad i na platformu Microsoft Azure. Zmíněná platforma byla v době zpracovávání praktické částí této práce značně přetížená, přičemž bylo zapotřebí několik dní čekat na přidělení výpočetních zdrojů, a i tak byly přiděleny pouze zdroje se základní konfigurací. Z tohoto důvodu bylo nutné omezit typ pořadů zpracovávaný v rámci datového vzorku pouze na filmy (namísto filmů, seriálů, dokumentů, sportu a dalších), jelikož zpracování původního datového vzorku by zabralo příliš času a diplomovou práci by nebylo možné odevzdat v termínu. I přesto je vytvořený model univerzální a po opětovném natrénování modelu Doc2vec na větším datovém vzorku jej lze použít i na ostatní druhy pořadů. Omezení na filmy je naimplementováno při zpracování datového vzorku jednoduchou podmínkou na filmy v SQL kódu, přičemž po odebrání zmíněné podmínky budou zdrojová data pro Doc2vec generována pro všechny druhy pořadů.

Pro potřeby této práce lze tedy hlavní i vedlejší cíl diplomové práce považovat za splněný.

Hlavním problémem, jenž vyvstal při zpracování praktické části diplomové práce bylo obecně zpracování a transformace zdrojových dat. Jak již bylo popsáno v páté kapitole, problém vznikl u zpracování EPG dat, jelikož data v systémech IPTV poskytovatele nebyla kompletní. Tudíž bylo zapotřebí EPG data brát přímo od EPG poskytovatele a vytvořit vlastní překladač dat v programovacím jazyce Python, jenž extrahuje data z XML souborů, získaných z webových stránek poskytovatele EPG dat.

Diplomová práce obsahuje kromě vlastního textu i řadu příloh, jenž obsahují programový kód vybraných částí praktické části diplomové práce. Příloha A obsahuje zdrojový kód v programovacím jazyce Python, jenž slouží jako překladač dat od EPG poskytovatele. Data jsou stažena z webových stránek EPG poskytovatele, extrahována z formátu XML, zpracována a uložena do formátu CSV. Příloha B obsahuje zdrojový kód pro přípravu datového vzorku a datových souborů, jenž jsou dále využívány při vytváření prediktivního modelu, přičemž výstupem jsou dva CSV soubory. Příloha C obsahuje zdrojový kód implementace algoritmu Doc2vec z knihovny Gensim v programovacím jazyce Python. Výstupem je Doc2vec model a CSV soubor, jenž obsahuje vektory pro každého unikátního uživatele. Příloha D obsahuje zdrojový kód v programovacím jazyce Python jenž slouží pro vizualizaci vektorového prostoru a vektorů vytvořených v šesté kapitole. Výstupem jsou vizualizace (obrázky) umístění vektorů v prostoru. Příloha E obsahuje zdrojový kód v programovacím jazyce Python, jenž obsahuje implementaci modů kNN a KDTree, včetně modelu jenž generuje doporučené pořady pro konkrétní uživatele. V rámci zdrojového kódu jsou uvedeny i poznámky o funkčnosti v anglickém jazyce, jelikož poskytovatel IPTV zaměstnává mnoho zahraničních zaměstnanců. Zdrojové cesty k souborům byly v rámci kódu upraveny tak, aby nedošlo k zveřejnění názvu IPTV poskytovatele.

Primárním výstupem této diplomové práce je vytvořený prediktivní model, jenž je schopen pro konkrétního uživatele individuálně vybrat relevantní doporučené pořady. Na základě vytvořeného modelu bude marketingové oddělení IPTV poskytovatele distribuovat emailový newsletter s doporučenými pořady konkrétnímu uživateli. Jako další výstup této diplomové práce lze považovat charakteristiku vybraných technologií, konceptů a nástrojů, jenž může čtenáři posloužit k jejich pochopení. Závěrečným výstupem této práce je, že může v rámci prostředí IPTV poskytovatele sloužit jako dokumentace k seznámení s problematikou dané oblasti.

Jak již bylo zmíněno, marketingové oddělení navrhlo rozšíření vytvořeného modelu na základě dalších parametrů jako jsou například: rok výroby, země původu, hlavní herci a jiné. Tato rozšíření by mohla přinést pro vybrané skupiny uživatelů lépe zacílená doporučení pořadů. Aby bylo možné zmíněná rozšíření naimplementovat, bylo by v první řadě nutné získat kvalitnější zdrojová EPG data od poskytovatele EPG dat, jelikož v současném stavu jsou tato data nevyhovující. V době dokončení této diplomové práce začalo marketingové oddělení IPTV poskytovatele jednat s poskytovatele EPG dat o navýšení kvality dat, nicméně do termínu odevzdání této práce nebyla tato jednání ukončena.

Použitá literatura

[1] INPLAYER. *A Survey Reveals Rising Popularity Of Video Streaming Over Live TV in the U.S.*. Inplayer [Online] [citace 31.1.2020]. Dostupné z: https://inplayer.com/a-survey-reveals-rising-popularity-of-video-streaming-over-live-tv-in-the-u-s/.

[2] PWC. *The streaming shakeup. A battle for video consumers in 2020.* PWC [Online] [citace 31.1.2020]. Dostupné z: https://www.pwc.com/us/en/services/consulting/library/pdf/consumer-video-streaming-behavior.pdf

[3] MADAHAVAN, S. *Mastering Python for data science: explore the world of data science through Python and learn how to make sense of data*. Birmingham: Packt Publishing, 2015. ISBN 978-1-78439-015-0

[4] VANDERPLAS, J. *Python Data Science Handbook.* California: O'Reilly Media, Inc., 2016. ISBN 978-1-491-91205-8

[5] MCKINNEY, W. *Python for data analysis: data wrangling with Pandas, NumPy, and IPython.* California: O'Reilly Media, Inc., 2018. ISBN 978-1-419-95766-0

[6] JANNACH, Dietmar, Markus ZANKER, Alexander FELFERNIG a Gerhard FRIEDRICH. *Recommender Systems An Introduction*. New York: Cambridge University Press 2011. ISBN 978-0-521-49336-9

[7] BERKA, P. *Dobývání znalostí z databází*. 1. vyd. Praha: Academia, 2003. ISBN 80-200-1062-9

[8] PCHI, Van-Thuy, Liu CHEN a Yu HIRATE. *Distributed Representation-based Recommender Systems in E-commerce.* Nara Institute of Science and technology [Online] 2016 [citace 31.1.2020]. Dostupné z: https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf

[9] MIKOLOV, Thomas, Ilya SUTSKEVER, Kai CHEN, Greg CORRADO a Jeffrey DEAN. *„Distributed Representations of Words and Phrases and their Compositionality"*. Google. [Online] 16. Října 2013 [citace 31.1.2020]. Dostupné z: https://arxiv.org/pdf/1310.4546.pdf.

[10] MIKOLOV, Thomas, Quoc LE. *„Distributed Representations of Sentences and Documents"*. Google. [Online] [citace 31.1.2020]. Dostupné z: https://cs.stanford.edu/~quocle/paragraph_vector.pdf.

[11] TOWARDS DATA SCIENCE. "*Towards Data Science Homepage".* Towards data science. [Online] [citace 13.2.2020]. Dostupné z: https://towardsdatascience.com/.

[12] SPARK 2.4.4 Documentation. "*Spark 2.4.4 Overview".* Apache Software Foundation. [Online] [citace 12.2.2020]. Dostupné z: https://spark.apache.org/docs/2.4.4/.

[13] ŘEHŮŘEK, Radim. "*Gensim Documentation".* Radimrehurek. [Online] [citace 13.4.2020]. Dostupné z: https://radimrehurek.com/gensim/auto_examples/index.html.

[14] MICROSOFT AZURE. "*Get started with Azure".* Portal Azure. [Online] [citace 13.2.2020]. Dostupné z: https://docs.microsoft.com/en-us/azure/.

[15] APACHE SPARK. "*Welcome to Spark python API Docs!".* Apache Spark. [Online] [citace 13.2.2020]. Dostupné z: https://spark.apache.org/docs/latest/api/python/index.html.

[16] PYTHON. "*Python Documentation".* Python. [Online] [citace 13.2.2020]. Dostupné z: https://www.python.org/doc/.

[17] VIDOVATION. "*IPTV Detailed Definition".* Vidovation. [Online] [citace 3.2.2020]. Dostupné z: https://vidovation.com/iptv-definition/.

[18] TECHTARGET. "*IPTV (Internet Protocol Television)".* Techtarget. [Online] [citace 3.2.2020]. Dostupné z: https://searchnetworking.techtarget.com/definition/IPTV-Internet-Protocol-television.

[19] TECHOPEDIA. "*Video on Demand (VoD)".* Techopedia. [Online] [citace 3.2.2020]. Dostupné z: https://www.techopedia.com/definition/25650/video-on-demand-vod.

[20] IMAGEN. "*What are SVOD, TVOD, AVOD?".* Imagen. [Online] [citace 3.2.2020]. Dostupné z: https://imagen.io/resources/what-are-svod-tvod-avod/.

[21] TECHTARGET. "*Internet Group management Protocol (IGMP)".* Techtarget. [Online] [citace 3.2.2020]. Dostupné z: https://searchnetworking.techtarget.com/definition/Internet-Group-Management-Protocol

[22] TECHTARGET. "*Real Time Streaming Protocol (RTSP)".* Techtarget. [Online] [citace 3.2.2020]. Dostupné z: https://searchvirtualdesktop.techtarget.com/definition/Real-Time-Streaming-Protocol-RTSP.

[23] TECHTARGET. "*electronic programming guide (EPG)".* Techtarget. [Online] [citace 25.2.2020]. Dostupné z: https://searchnetworking.techtarget.com/definition/electronic-program-guide.

[24] CLIFTON Christopher. *„Data Mining".* Britannica. [Online] [citace 31.1.2020]. Dostupné z: https://www.britannica.com/technology/data-mining.

[25] TECHTARGET. "*Data Mining".* Techtarget. [Online] [citace 4.2.2020]. Dostupné z: https://searchsqlserver.techtarget.com/definition/data-mining.

[26] GURU99. "*Data Mining Tutorial: Process, Techniques, Tools, Examples".* Guru99. [Online] [citace 5.2.2020]. Dostupné z: https://www.guru99.com/data-mining-tutorial.html.

[27] TECHTARGET. "*Algorithm".* Techtarget. [Online] [citace 5.2.2020]. Dostupné z: https://whatis.techtarget.com/definition/algorithm.

[28] MICROSOFT. "*Mining Models (Analysis Services – Data Mining)".* Microsoft. [Online] [citace 5.2.2020]. Dostupné z: https://docs.microsoft.com/en-us/analysis-services/data-mining/mining-models-analysis-services-data-mining?view=sql-analysis-services-2019.

[29] FINANCE TRAIN. "*Difference between model and algorithm".* Finance train. [Online] [citace 5.2.2020]. Dostupné z: https://financetrain.com/difference-between-model-and-algorithm/.

[30] SALIAN, ISHA. *"What's the Difference Between Supervised, Unsupervised, Semi-Supervised and Reinforcement Learning?".* Nvidia. [Online] 2. Srpna 2018 [citace 5.2.2020]. Dostupné z: https://blogs.nvidia.com/blog/2018/08/02/supervised-unsupervised-learning/.

[31] MUTUVI, Steve. *"Introduction to machine learning model evaluation".* Nvidia. [Online] 16. Května 2019 [citace 5.2.2020]. Dostupné z: https://heartbeat.fritz.ai/introduction-to-machine-learning-model-evaluation-fa859e1b2d7f.

[32] SAS. "*What is Natural Language Processing?"*. SAS. [Online] [citace 7.2.2020]. Dostupné z: https://www.sas.com/en_us/insights/analytics/what-is-natural-language-processing-nlp.html.

[33] ROUSSE, Margaret. "*natural language processing (NLP)"*. Techtarget. [Online] [citace 7.2.2020]. Dostupné z: https://searchbusinessanalytics.techtarget.com/definition/natural-language-processing-NLP.

[34] PASCUAL, Federico. "*The essential guide to topic modeling"*. MonkeyLearn. [Online] [citace 7.2.2020]. Dostupné z: https://monkeylearn.com/blog/introduction-to-topic-modeling/.

[35] MALLET. "*Topic modeling"*. University of Massachusetts Amherst. [Online] [citace 7.2.2020]. Dostupné z: http://mallet.cs.umass.edu/topics.php.

[36] NICHOLSON, Chris. "*A Beginner's Guide to Word2Vec and Neural Word Embeddings"*. Pathmind. [Online] [citace 7.2.2020]. Dostupné z: https://pathmind.com/wiki/word2vec.

[37] KARANI, Dhruvil. "*Introduction to Word Embedding and Word2vec"*. Pathmind. [Online] [citace 7.2.2020]. Dostupné z: https://towardsdatascience.com/introduction-to-word-embedding-and-word2vec-652d0c2060fa.

[38] SHPERBER, Gidi. "*A gentle introduction to Doc2vec"*. Wisio. [Online] 26. Července 2017 [citace 7.2.2020]. Dostupné z: https://medium.com/wisio/a-gentle-introduction-to-doc2vec-db3e8c0cce5e.

[39] MA, Edward. "*Understand how to transfer your paragraph vector by doc2vec "*. Towards data science. [Online] [citace 10.2.2020]. Dostupné z: https://towardsdatascience.com/understand-how-to-transfer-your-paragraph-to-vector-by-doc2vec-1e225ccf102.

[40] ROUSSE, Margaret. "*recommendation engine"*. Techtarget. [Online] [citace 10.2.2020]. Dostupné z: https://whatis.techtarget.com/definition/recommendation-engine.

[41] MALL, Rishab. "*Recommender systems "*. Towards data science. [Online] [citace 10.2.2020]. Dostupné z: https://towardsdatascience.com/recommender-system-a1e4595fc0f0.

[42] RODRIGUEZ, Gaston. "*Introduction to recommender systems in 2019"*. Tryo labs. [Online] [citace 10.2.2020]. Dostupné z: https://tryolabs.com/blog/introduction-to-recommender-systems/.

[43] ROCCA, Baptiste. "*Introduction to recommender systems "*. Towards data science. [Online] [citace 10.2.2020]. Dostupné z: https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada.

[44] SMART VISION EUROPE. "*What is CRISP-DM methodology?"*. SV-Europe. [Online] [citace 11.2.2020]. Dostupné z: https://www.sv-europe.com/crisp-dm-methodology/.

[45] MOREIRA MAFRA, Gabriela. "*CRISP-DM and what I did wrong"*. Magrathea Labs. [Online] [citace 11.2.2020]. Dostupné z: https://blog.magrathealabs.com/crisp-dm-and-what-i-did-wrong-70c4e7e8656.

[46] AMAZON AWS. "*What is Apache Kafka?"*. Amazon AWS. [Online] [citace 11.2.2020]. Dostupné z: https://aws.amazon.com/msk/what-is-kafka/.

[47] ROUSSE, Margaret. "*Microsoft Azure"*. Techtarget. [Online] [citace 11.2.2020]. Dostupné z: https://searchcloudcomputing.techtarget.com/definition/Windows-Azure.

[48] MICROSOFT AZURE. "*Data Factory".* Portal Azure. [Online] [citace 12.2.2020]. Dostupné z: https://azure.microsoft.com/en-us/services/data-factory/.

[49] MICROSOFT AZURE. "*Blob Storage".* Portal Azure. [Online] [citace 12.2.2020]. Dostupné z: https://azure.microsoft.com/en-us/services/storage/blobs/.

[50] MICROSOFT AZURE. "*Azure Data Lake Storage".* Portal Azure. [Online] [citace 12.2.2020]. Dostupné z: https://azure.microsoft.com/en-us/services/storage/data-lake-storage/.

[51] MICROSOFT AZURE. "*Azure Databricks".* Portal Azure. [Online] [citace 12.2.2020]. Dostupné z: https://azure.microsoft.com/en-us/services/databricks/.

[52] MICROSOFT AZURE. "*Azure Machine Learning".* Portal Azure. [Online] [citace 13.4.2020]. Dostupné z: https://azure.microsoft.com/en-us/services/machine-learning/

[53] APACHE SPARK. "*Apache Spark".* Apache Software Foundation. [Online] [citace 12.2.2020]. Dostupné z: https://spark.apache.org/.

[54] O'Reilly Library. "*What is Apache Spark?".* O'Reilly. [Online] [citace 12.2.2020]. Dostupné z: https://www.oreilly.com/library/view/learning-spark/9781449359034/ch01.html.

[55] GITHUB. "*An introduction to Apache Spark".* GitHub. [Online] [citace 12.2.2020]. Dostupné z: https://github.com/dhesse/SparkTalk/blob/master/SparkTalk.md.

[56] PyPi. "*Gensim".* PyPi. [Online] [citace 13.2.2020]. Dostupné z: https://pypi.org/project/gensim/.

[57] SCIKIT LEARN. "*KDTree".* Scikit-learn. [Online] [citace 29.4.2020]. Dostupné z: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KDTree.html?highlight=kdtree.

[58] TUTORIALSPOINT. "*Python 3 - Numbers".* Tutorialspoint. [Online] [citace 13.2.2020]. Dostupné z: https://www.tutorialspoint.com/python3/python_numbers.htm.

[59] TUTORIALSPOINT. "*Python 3 - Strings".* Tutorialspoint. [Online] [citace 13.2.2020]. Dostupné z: https://www.tutorialspoint.com/python3/python_strings.htm.

[60] DONNELY, Caroline. "*Coronavirus: Microsoft Azure suffers datacentre capacity shortages in Europe".* Computer Weekly.[Online] 7. Dubna 2020 [citace 13.4.2020]. Dostupné z: https://www.computerweekly.com/news/252481265/Coronavirus-Microsoft-Azure-suffers-datacentre-capacity-shortages-in-Europe.

[61] KOEHRSEN, Will. "*Neural Network Embeddings Explained".* Towards data science. [Online] [citace 30.3.2020]. Dostupné z: https://towardsdatascience.com/neural-network-embeddings-explained-4d028e6f0526.

[62] ŘEHŮŘEK, Radim. "*Doc2vec paragraph embeddings".* Radimrehurek. [Online] [citace 13.4.2020]. Dostupné z: https://radimrehurek.com/gensim/models/doc2vec.html.

[63] ŘEHŮŘEK, Radim. "*Doc2vec model".* Radimrehurek. [Online] [citace 22.4.2020]. Dostupné z: https://radimrehurek.com/gensim/auto_examples/tutorials/run_doc2vec_lee.html.

[64] DELANEY, Jeff. "*Visualizing Word Vectors with t-SNE".* Kaggle. [Online] [citace 22.4.2020]. Dostupné z: https://www.kaggle.com/jeffd23/visualizing-word-vectors-with-t-sne.

[65] TECHOPEDIA. "*k-Nearest neighbor".* Techopedia. [Online] [citace 22.4.2020]. Dostupné z: https://www.techopedia.com/definition/32066/k-nearest-neighbor-k-nn.

[66] SCIKIT LEARN. "*accuracy_score".* Scikit-learn. [Online] [citace 29.4.2020]. Dostupné z: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html.

Přílohy

Příloha A: Zdrojový kód – XML překladač EPG dat

```
from datetime import timedelta, date, datetime
from decimal import Decimal, getcontext
import numpy as np
import os
import pandas as pd
import requests
import shutil
import xml.etree.ElementTree as ET

# variables with folders and file locations
input_folder = r'\path\to\folder\input_folder'
input_files = os.listdir(input_folder)
output_file = r'\path\to\folder\data.txt'

# define date range of the downloaded EPG files
start_date = date(YYYY, M, D)
end_date = date(YYYY, M, D)

# define channels for which the EPG files will be downloaded
channels = ["channe_code_1", "channe_code_2", "channe_code_3", ...]

# variables with URL structure
url_base = r'http://services.data.tvprofi.cz/iptv_provider_name/export-
komplet.aspx?type=program&date='
code = "&code="

# function prepares environment - creates date range & removes old files, then runs the
download function
def prepare_environment():
    dates = []

    def listofdates(dt1, dt2):
        for n in range(int((dt2 - dt1).days) + 1):
            yield dt1 + timedelta(n)

    for datum in listofdates(start_date, end_date):
        dates.append(datum)
```

```
    for filename in os.listdir(input_folder):
        filepath = os.path.join(input_folder, filename)
        try:
            shutil.rmtree(filepath)
        except EnvironmentError:
            os.remove(filepath)

    os.remove(output_file)
    open(output_file, "w")
    download(dates, channels)

# function downloads EPG files from EPG provider website in XML format, then runs file
processing
def download(dates, channels):
    for channel in channels:
        for d in dates:
            current_date = str(d)
            current_channel = str(channel)
            current_url = os.path.join(url_base + current_date + code + current_channel)
            print(current_url)
            current_output_name = str(current_channel + str(d.year) + "-" + str(d.month) +
"-" + str(d.day) + ".xml")
            myfile = requests.get(current_url)
            open(input_folder + "\\" + current_output_name, 'wb').write(myfile.content)

    process_files(input_files)

# function parses the downloaded XML files, chooses specified values and stores all data
into one TXT file
def process_files(input_files):
    for file in input_files:
        current_file = str(input_folder + "\\" + file)
        print(current_file)
        tree = ET.parse(current_file)
        root = tree.getroot()
        f = open(output_file, 'a', encoding='utf-8')

        incomplete_dt = "Přestávka ve vysílání"
        getcontext().prec = 3

        for program in root.find('program').find("porad").findall("porad"):

            # Parses EPG provider ID of the TV show and channel name
            f.write(program.attrib.get('id') + ';' +
root.find("program").attrib.get("stanice") + ';')
```

```
            name = program.find('nazev').text

            # Parses date out of the specified columns, whichever is avaiable
            if float(program.find('cas-od').text) <= 23.59:

f.write(str(datetime.strptime(root.find("program").find("porad").attrib.get("datum"), '%Y-
%m-%d'))[0:10] + ';')
            else:

f.write(str(datetime.strptime(root.find("program").find("porad").attrib.get("datum"), '%Y-
%m-%d') + timedelta(days=1))[0:10]  + ';')

            # Parses TV show name, and counts the length if name in incomplete_dt list
            if incomplete_dt in name:
                f.write("0" + str(Decimal(program.find('cas-od').text) - Decimal(24))[0:4]
+ ";"+ program.find('delka').text + ";None;" + name + ";0;0;None")
            else:
                f.write(program.find('cas-od2').text + ';' + program.find('delka').text +
';')

                # Parses content type value, if exists, else inserts as "None"
                content = program.find('content_type')
                if content is not None:
                    c = content.text
                    f.write(str(c) + ';')
                else:
                    f.write("None;")

                f.write(name + ";")

                # Parses the TVProfi score, if exists, else inserts as "None"
                tvprofi = program.find('TVPROFI_score')
                if tvprofi is not None:
                    a = tvprofi.text
                    f.write(str(a) + ";")
                else:
                    f.write("None;")

                # Parses the IMDB official rating, if exists, else inserts as "None"
                imdb = program.find('IMDB_rating')
                if imdb is not None:
                    b = imdb.text
                    f.write(str(b) + ";")
                else:
```

```
                    f.write("None;")

                # Parses genres of the TV show, if exists
                for genre in program.find('genres').findall('genre'):
                    f.write(genre.text + "_")

            f.write("\n")

# functions stores the data stored in the TXT file, adds defined columns and saves the
output to CSV file
def storeasCSV(output_file):
    # create the data frame header row
    headings = ['id', 'channel_name', 'date', 'start_hr', 'lenght', 'type', 'name',
'tvprofi', 'imdb', 'genre']

    # read the txt file containing the data
    df = pd.read_csv(r'C:\Users\polak\Documents\data.txt', names=headings, delimiter=";",
encoding="utf-8")

    # replace character "." with ":" where aplicable in column start_hr
    df["start_hr"] = df["start_hr"].str.replace('.', ':')

    # replace character "_" with white space where aplicable in genres
    df["genre"] = df["genre"].str.replace('_', ' ')

    # transfer value of channel name column to lowercase
    df['channel_name'] = df['channel_name'].str.lower()

    # create column combining date and time as string
    df["start_timestamp"] = df["date"] + " " + df["start_hr"]

    # convert the created columnt to timestamp
    df["start_timestamp"] = pd.to_datetime(df["start_timestamp"])

    # convert the timestamp to unixtime
    df['start_unix'] = df['start_timestamp'].astype(np.int64) // 10 ** 9

    # calculate the lenght of the show in seconds as integer
    df['lenght_unix'] = df['lenght'] * 60

    # calculate the  40% of lenght of the show in seconds as integer
    df['avg_length'] = df['lenght_unix'] * 0.4

    # calculate the show end time in unix time
    df['end_unix'] = df['start_unix'] + df['lenght_unix']
```

```
    # convert the calculated show end time from unix time to timestamp
    df["end_timestamp"] = pd.to_datetime(df['end_unix'].astype(int), unit='s')

    # save the output dataframe to datafile in csv format
    df.to_csv(r'\path\to\folder\data.csv', index=False, sep=";", encoding="utf-8-sig")

# runs the first stage of the script - prepare environment
prepare_environment()
```

Příloha B: Zdrojový kód – Příprava datového vzorku

```
from datetime import timedelta, date, datetime
from os.path import expanduser, join, abspath
from pyspark.sql.functions import *
from pyspark.sql import SparkSession, Row

spark = SparkSession \
    .builder \
    .appName("Python Spark SQL Hive integration example") \
    .enableHiveSupport() \
    .getOrCreate()

start_date = date(YYYY, M, D)
end_date = date(YYYY, M, D)

dates=[]

def listofdates(dt1,dt2):
  for n in range(int((dt2 - dt1).days) + 1):
    yield dt1 + timedelta(n)

for datum in listofdates(start_date,end_date):
  dates.append(datum.strftime('%Y%m%d'))# creates EPG dataframe containing only movies
df_epg = spark.sql("select * from epg_x where type = 'film'")

# save paths
save_usr_act_path = '/path/to/data/user_activity'
save_epg_dt_path = '/path/to/data/epg_data'

# selects relevant channels data
relevant_channel_ids =
['318','14','16','316','286','556','320','875','801','553','877','813','777','810','748','
1083','284','880','40','883','104','56','731','783','786','888','1066']
```

```
# leaves subscription code and epg id
dropped_fields_usr_act = ("device_type", "device_detail", "device_id", "event_type",
"session_start", "event_start", "event_end", "device_mode", "broadcast_type",
"channel_id", "edr_id", "entity_id", "isp_id", "slice", "name", "tvprofi", "imdb",
"start_planned", "end_planned", "avg_length", "start_timestamp", "end_timestamp",
"lenght_unix", "event_source", "channel_name", "id", "start_unix", "end_unix", "date")

# leaves epg id and tvprofi score
dropped_fields_epg_dt = ("device_type", "device_detail", "device_id", "event_type",
"session_start", "event_start", "event_end", "device_mode", "broadcast_type",
"channel_id", "edr_id", "entity_id", "isp_id", "slice", "name", "imdb", "start_planned",
"end_planned", "avg_length", "start_timestamp", "end_timestamp", "lenght_unix",
"event_source", "channel_name", "id", "start_unix", "end_unix", "date", "type", "genre",
"subscription_code")

# once the where condition is dropped, data will be available for all content types
df_epg = spark.sql("select * from epg_db where type = 'film'")

# loads parquet files for each day in the previously specified date range into dataframe
paths = []
for date in dates:
  paths.append('/mnt/iptvdatalake/curated/iptv_aggregated/date=' + date)

df_norm = spark.read.parquet(*paths)
df_norm = df_norm.filter(df_norm.channel_id.isin(relevant_channel_ids))

# creates single datframes and filters them, then saves them to specified path
df = df_norm.join(df_epg, df_norm.epg_id == df_epg.epg_id, how =
"inner").filter(df_norm.event_source == 'EKT').drop(df_norm.epg_id)
df = df.filter(df.genre.isNotNull())

df = df.filter(df.subscription_code.rlike("^[0-9]{10}$"))
df = df.filter(df.tvprofi.rlike("\d{2}"))

df_usr_act = df.drop(*dropped_fields_usr_act)
df_usr_act = df_usr_act.distinct()

df_epg_dt = df.drop(*dropped_fields_epg_dt)
df_epg_dt = df_epg_dt.distinct()

(df_usr_act.repartition(1).write.format('csv').mode('append').save(save_usr_act_path))
(df_epg_dt.repartition(1).write.format('csv').mode('append').save(save_epg_dt_path))
```

Příloha C: Zdrojový kód – Implementace modelu Doc2vec

```
import os
import datetime
import gensim
import pandas as pd
import re
import smart_open
import scipy.spatial as spt
import azureml.dataprep
from azureml.core.run import Run

# creates azure machine learning service context and loads datasets
run = Run.get_context()

base_path = run.input_datasets['iptv_user_activity_file']
train_file =
os.path.join(base_path,"iptvdatalake_curated","iptv_embeddings","user_activity","user_acti
vity.csv")
# defines function read_corpus, which reads the training/test data file line by line and
assigns tag ids, method taken (with modification) from official documentation [62]
def read_corpus(fname, tokens_only=False):
    with smart_open.open(fname, encoding="utf-8-sig") as f:
        for i, line in enumerate(f):
            list=[]
            list = line.split(sep=',')
            sub_id = list[0]
            epg_id = list[1]
            tokens = gensim.utils.simple_preprocess(line)
            if tokens_only:
                yield tokens
            else:
                yield gensim.models.doc2vec.TaggedDocument(tokens, [sub_id, epg_id])

# sets model parameters
vec_size = 100
wind = 5
a_alpha = 0.025
m_alpha = 0.0025
min_c = 25
alg = 1
threads = 8
no_epochs = 100

# logging info
```

```
run.log('Vector size',vec_size)
run.log('Words window',wind)
run.log('Alpha - learning rate',a_alpha)
run.log('Min Alpha - Minimum learning rate',m_alpha)
run.log('Words minimum count',min_c)
run.log('Algorithm used',alg)
run.log('Number of cores',threads)
run.log('Number of epochs', no_epochs)

# reads the corpus of data file
train_corpus = list(read_corpus(train_file))
print(train_corpus)

# creates the Doc2vec model and model vocabulary
model = gensim.models.doc2vec.Doc2Vec(vector_size = vec_size, window = wind, alpha =
a_alpha, min_alpha = m_alpha, min_count = min_c, dm = alg, workers = threads)
model.build_vocab(train_corpus)

# trains the created model over the specified number of epochs, while lowering alpha
(model learning rate) with each iteration
for e in range(no_epochs):
    print('iter - {0}'.format(e+1))
    model.train(train_corpus, total_examples=model.corpus_count,epochs=model.epochs)
    model.alpha -= 0.002
    model.min_alpha = model.alpha

print('Training Finished! - Saving model')

# saves the model
os.makedirs('./outputs', exist_ok=True)
model_ts = datetime.datetime.now().strftime('%Y%m%d_%H%M%S')
model.save(os.path.join('./outputs', model_ts + ".model"))
run.log('Model name', model_ts)

headings = ['user','epg', 'type', 'genre']

df = pd.read_csv(train_file, sep = ',', names = headings, low_memory=False)
df.drop(columns=['epg', 'type', 'genre'])
df = df.user.unique()

out = open(r'./outputs/user_vecs.csv', "w")

for i in df:
    usr = str(i)
    k = (usr + ";" + " ".join([str(x) for x in model.docvecs[usr]]))
```

```
    k = k.rstrip()
    k = re.sub(r"\s+", ';', k)
    out.write(k + '\n')

out.flush()
out.close()
print('User vectos Saved')
```

Příloha D: Zdrojový kód – Vizualizace Embeddings

```
import gensim
import pandas as pd
import numpy as np
import re
import nltk
import umap
from sklearn.manifold import TSNE
import matplotlib.pyplot as plt

# load gensim model
model_path = r'\path\to\model\20200416_031821.model'
model = gensim.models.doc2vec.Doc2Vec.load(model_path)
model.init_sims(replace=True)

# load word2vec vectors
X = model[model.wv.vocab]
X.shape

# load doc2vec vectors
Y = model[model.docvecs.doctags]
Y.shape

# function that fits the TSNE model - implementation taken (with modification) from Kaggle
article [63]
def plot_w2v(model):
    labels = []
    tokens = []

    for word in model.wv.vocab:
        tokens.append(model[word])
        labels.append(word)

    tsne_model = TSNE(perplexity=40, n_components=2, init='pca', n_iter=2500,
random_state=23)
```

```
    new_values = tsne_model.fit_transform(tokens)

    x = []
    y = []
    for value in new_values:
        x.append(value[0])
        y.append(value[1])

    plt.figure(figsize=(10, 10))
    for i in range(len(x)):
        plt.scatter(x[i],y[i])
        plt.annotate(labels[i],
                     xy=(x[i], y[i]),
                     xytext=(5, 2),
                     textcoords='offset points',
                     ha='right',
                     va='bottom')
    plt.show()

# function that plots UMAP for doc2vec model
def plot_d2v():
    emb = umap.UMAP(n_neighbors=[3,4,5], min_dist=0.0, n_components=2,
random_state=42).fit_transform(Y)

    plt.figure(figsize=(10,9))
    plt.scatter(emb[:,0], emb [:,1], s=3, cmap='Spectral')

# plots the word2vec model
plot_w2v(model)
#plots doc2vec model
plot_d2v()
```

Příloha E: Zdrojový kód – Implementace prediktivního modelu

```
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier, KDTree
from sklearn.preprocessing import StandardScaler

import os
import numpy as np
import pandas as pd
import azureml.dataprep
from azureml.core.run import Run
```

```
# imports user vectors data, returns dataframe with user id and vectors
def prep_user_data(base_path):
    user_data_path = os.path.join(base_path, "iptvdatalake_curated", "iptv_embeddings",
"user_vectors.csv")
    user_data = pd.read_csv(user_data_path, sep = ';', header = None, low_memory=False)
    user_data.head()
    #print('Number of distinct users:', len(user_data[0].unique()))
    return user_data

# imports epg data, returns epg sorted list (by tvprofi score)
def prep_epg_data(base_path):
    epg_data_path = os.path.join(base_path, "iptvdatalake_curated", "iptv_embeddings",
"epg_data", "epg_data.csv")
    df_epg = pd.read_csv(epg_data_path, sep=',', header = None)
    df_epg = df_epg.sort_values(by=[1], ascending=False)
    epg_data = df_epg[0].tolist()
    return epg_data

# imports user activity data, returns dataframe with user id and seen epg id
def prep_actvity_data(base_path):
    act_data_path = os.path.join(base_path, "iptvdatalake_curated", "iptv_embeddings",
"user_activity", "user_activity.csv")
    headings = ['user', 'epg', 'type', 'genre']
    activity_data = pd.read_csv(act_data_path, sep = ',', names = headings,
low_memory=False)
    activity_data = activity_data.drop(columns=['type', 'genre'])
    #print('Number of distinct user activity logs:', len(activity_data[0]))
    #print('Number of distinct shows:', len(activity_data[1].unique()))
    return activity_data

# train knn, to pick best k (number of neighbours), out of specified range
def knn(data):
    # X contains second column - vectors
    X = data.iloc[:,1:].values
    # Y contains first columns - names
    Y = data.iloc[:,0].values

    # split data into training and testing parts, 70% / 30% ratio
    X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.30,
random_state=42)

    # standardize data form
    scaler = StandardScaler()
    scaler.fit(X_train)
```

```
    X_train = scaler.transform(X_train)
    X_test = scaler.transform(X_test)

    # define range of nearest neighbours
    no_of_neighbours = range(4,21)
    k_dict = {}

    # train knn for every number "k" in specified range, add k and its accuracy to
dictionary
    for k in no_of_neighbours:
        knn = KNeighborsClassifier(n_neighbors=k)
        knn.fit(X_train, y_train)
        prediction = knn.predict(X_test)
        acc = accuracy_score(y_test, prediction)
        k_dict[k] = acc

    num = max(k_dict, key=k_dict.get)
    a = max(k_dict.values())
    print(k_dict)
    run.log('Number of neigbours in KNN',num)
    print('Number of neighbours chosen from KNN:', num - 1,'with accuracy:', a)
    return num

# trains kdtree, returns nearest neighbours for each user
def train_kdtree(data, best_k):
    kdt = KDTree(data)
    no_of_neighbors = best_k
    dis, idx = kdt.query(data, k=no_of_neighbors)
    return idx

# creates data frame, each row contains targeted user and first three neighbours
def neighbours_to_df(data, idx, best_k):
    headings = []
    for i in range(best_k):
        headings.append(str(i))
    df_n = pd.DataFrame(idx, columns=headings)
    for k in headings:
        df_n[k] = df_n[k].map(data[0])
    return df_n

# creates dictionary, with each users activity
def get_user_activity(data):
    watched = {}
    for i, row in data.iterrows():
        curr_id = data.iloc[i,0]
```

```
        curr_val = data.iloc[i,1]
        watched.setdefault(curr_id, [])
        watched[curr_id].append(curr_val)
    return watched

# creates lists with unseen movie, for every user
def recommend(df_neigbours, watched, epg_data, best_k):
    os.makedirs('./outputs', exist_ok=True)
    lsx = []
    for i in range(best_k):
        lsx.append(str('n'+str(i)))
    out = open(r'./outputs/predictions.csv', "w")
    for index in range(len(df_neigbours)):
        # get epg values in list for each user
        target_id = int(df_neigbours.iloc[index][0])
        target_seen = watched.get(target_id)

        curr_list = [] # contains all shows seen by nearest neigbours, one list = one
user, values repeat
        for i in range(best_k-1):
            n = watched.get(int(df_neigbours.iloc[index][i+1]))
            curr_list.append(n)
            if i == best_k:
                break

        mrg_list = [] # list containing merge of all curr_list, one list = all users,
values repeat
        for i in curr_list:
            mrg_list.extend(i)

        # combine lists with epg values, substract seen shows by targeted user
        predict_list = sorted(np.unique(mrg_list))
        recommended_list = list(set(predict_list) - set(target_seen))

        # sort the lists with predictions for each user, sorted by tvprofi score
        recommended_sorted = sorted(recommended_list, key=lambda x: epg_data.index(x))

        # write prediction to csv file
        out.write(str(target_id) + "," + str(recommended_sorted) + "\n")
    out.close()

if __name__ ==  '__main__':
    run = Run.get_context()
    base_path = run.input_datasets['iptv_vectors_file']
    base_path2 = run.input_datasets['iptv_user_activity_file']
```

```
base_path3 = run.input_datasets['iptv_epg_data']

print('--- Loading user vectors data ---')
user_vecs = prep_user_data(base_path)
print('--- Loading user activity data ---')
actv_data = prep_actvity_data(base_path2)
print('--- Training KNN ---')
best_k = knn(actv_data)
print('--- Training KDTree ---')
kdtree = train_kdtree(user_vecs, best_k)
print('--- Selecting neighbors for each user ---')
neighbor = neighbours_to_df(user_vecs, kdtree, best_k)
print('--- Loading user activity to dictionary ---')
activity = get_user_activity(actv_data)
print('--- Loading dictionary with epg values ---')
epg_data = prep_epg_data(base_path3)
print('--- Preparing recommendations ---')
recommend(neighbor, activity, epg_data, best_k)
print('--- Recommendations saved to output file ---')
```