Data science v prostředí Apache Spark

Roman Hanzlík

Diplomová práce
2021

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav informatiky a umělé inteligence

Akademický rok: 2020/2021

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Roman Hanzlík**
Osobní číslo:	**A19462**
Studijní program:	**N3902 Inženýrská informatika**
Studijní obor:	**Informační technologie**
Forma studia:	**Kombinovaná**
Téma práce:	**Data science v prostředí Apache Spark**

Zásady pro vypracování

1. Zpracujte literární rešerši na dané téma.
2. Proveďte popis základních komponent Data Science.
3. Popište prostředí Apache Spark pro distribuované výpočty.
4. Vytvořte funkční sady demonstračních příkladů pro prostředí Apache Spark na různých datasetech.
5. Proveďte celkové zhodnocení a závěr.

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:

1. *Data science & big data analytics: discovering, analyzing, visualizing and presenting data*. Indianapolis: Wiley, [2015], xviii, 410 s. ISBN 9781118876138.
2. GRUS, Joel. *Data science from scratch*. Sebastopol: O'Reilly, 2015, xvi, 311 s. ISBN 9781491901427.
3. OJEDA, Tony, Sean Patrick MURPHY, Benjamin BENGFORT a Abhijit DASGUPTA. *Practical data science cookbook: 89 hands-on recipes to help you complete real-world data science projects in R and Python*. Birmingham: Packt Publishing, 2014, 380 s. ISBN 9781783980246.
4. MILES, Matthew B., A. M. HUBERMAN a Johnny SALDAÑA. *Qualitative data analysis: a methods sourcebook*. Fourth edition. Los Angeles: SAGE, [2020], xxi, 380 s. ISBN 9781544371856.
5. KARAU, Holden, Andy KONWINSKI, Patrick WENDELL a Matei ZAHARIA. *Learning Spark*. Sebastopol: O'Reilly, 2015, xvi, 256 s. ISBN 9781449358624.
6. RYZA, Sandy, Uri LASERSON, Sean OWEN a Josh WILLS. *Advanced analytics with Spark*. Beijing: O'Reilly, 2015, xii, 260 s. ISBN 9781491912768.
7. DORSEY, Richard. *Data analytics*. [CreateSpace Independent Publishing Platform], [2017], 67 s. ISBN 9781547089291.
8. ANKAM, Venkat. *Big data analytics: a handy reference guide for data analysts and data scientists to help to obtain value from big data analytics using Spark on Hadoop clusters*. Birmingham: Packt, 2016, xv, 300 s. ISBN 9781785884696.

Vedoucí diplomové práce: **doc. Ing. Roman Šenkeřík, Ph.D.**
Ústav informatiky a umělé inteligence

Datum zadání diplomové práce: **15. ledna 2021**
Termín odevzdání diplomové práce: **20. srpna 2021**

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky
DĚKAN
E.S.

doc. Mgr. Milan Adámek, Ph.D.
děkan

prof. Mgr. Roman Jašek, Ph.D.
ředitel ústavu

Ve Zlíně dne 15. ledna 2021

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně;
- byl/a jsem seznámen/a s tím, že na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne 20. 8. 2021 Roman Hanzlík v.r.

ABSTRAKT

Tato diplomová práce představuje téma Data Science jako nový fenomén v oblasti počítačového zpracování dat. Hlavním cílem této práce je poskytnout prvotní náhled do problematiky Data Science a v krátkosti představit její dílčí oblasti se zaměřením na Big Data a Machine Learning jako dva pilíře, které hrají v posledních letech primární úlohu v rychle se měnící době, zejména v oblasti informačních technologií, což je odvětví, které zásadním způsobem zasahuje snad už do všech oblastí lidské činnosti.

Teoretická část nejprve podává přehled historie zpracování dat a informací a představuje faktory, které vedly k potřebě nového přístupu ve zpracování dat. Značná část je věnována představení metodik v oblasti zpracování dat. Neodmyslitelnou součástí je samotná definice Data Science a jejich základních komponent, Big Data včetně datového inženýrství a přehled možností a typů analýz dat.

Praktická část popisuje základní koncepty Apache Spark vč. několika možností instalací jako jsou on-premise či in-cloud. Dále se zaměřuje na představení možnosti Apache Spark v rámci jeho základních komponent přímo na reálných případech použití s využitím některých veřejně dostupných datových sad. Součástí práce je sada ukázkových příkladů s funkčními řádky kódů, které demonstrují využití dané technologie.

Klíčová slova:

Data, Data Science, Data Engineering, Big Data, Machine Learning, Data Mining, Matematika, Statistika, Analýza, DLM, CRISP-DM, DSMM, Apache Spark

ABSTRACT

This master thesis introduces the topic of Data Science as a new phenomenon in the field of computer data processing. The main objective of this thesis is to provide an initial insight into the area of Data Science and to briefly introduce its sub-areas, focusing on Big Data and Machine Learning as two pillars that have played a primary role in recent years in a rapidly changing era, especially in the field of information technology, an industry that has already fundamentally affected perhaps all areas of human activity.

The theoretical part first gives an overview of the history of data and information processing and presents the factors that led to the need for a new approach in data processing. A significant part is devoted to introducing methodologies in data processing. An essential part is the actual definition of Data Science and its basic components, Big Data including data engineering and a review of the possibilities and types of data analysis.

The practical part describes the basic concepts of Apache Spark including several installation options such as on-premise or in-cloud. It also focuses on presenting the capabilities of Apache Spark within its core components directly on real use cases using some of the publicly available datasets. This paper includes a set of sample examples with working lines of code that demonstrate the use of the technology.

Keywords:

Data, Data Science, Data Engineering, Big Data, Machine Learning, Data Mining, Mathematics, Statistics, Analytics, Analysis, DLM, CRISP-DM, DSMM, Apache Spark

PODĚKOVÁNÍ

Na tomto místě bych rád poděkoval všem, ať už z akademické obce, tak rodině a svým spolužákům a kolegům, kteří mě v průběhu celého studia podporovali a motivovali v tom, abych vydržel až do konce. Speciální dík patří těm, kteří v době svých zasloužených dovolených i v nočních hodinách si vždy našli čas na radu či se mnou absolvovali poslední zkoušky. Ačkoliv je akademický svět na hony vzdálený realitě a praxi v komerční sféře, je mou milou povinností přiznat, že i zde člověk narazí na osobnosti, se kterými by se rád potkal a spolupracoval na komerčním projektu, a to nejen z hlediska odborné erudovanosti, tak zejména po lidské stránce.

Scientia potentia est, sapientiae privilegium.
[Knowledge is power, wisdom privilege.]
Roman Hanzlík

Without data, you're just another person with an opinion.
W. Edwards Deming

OBSAH

ÚVOD

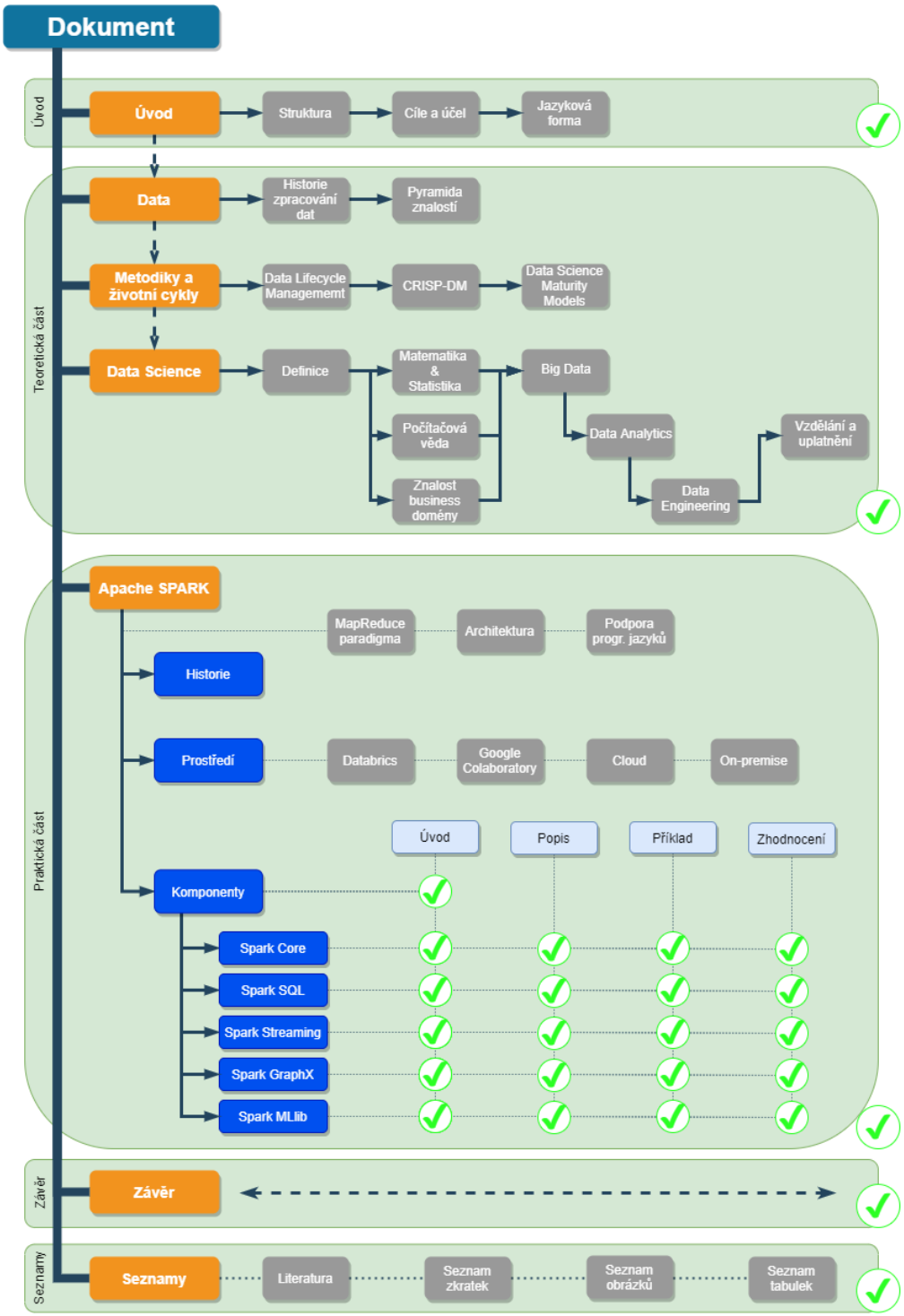
Rozhlédněte se a řekněte mi, co vidíte? Jeden by řekl „Města, pole, hory ...“, někdo jiný „Cvrlikání ptáků, šumění řek, shon lidí ...“ a já? Já vidím data všude kolem. Data v podobě teploty, větru, vlhkosti, když se probudím a otevřu okno. Vidím cenu na burze v ranních zprávách, když zapnu televizi. Při každodenní cestě do kanceláře vidím dopravní data. Jakmile se přihlásím, vidím data za docházkovým systémem. Vidím data v bezpečnostních kamerách. Vidím data na Twitteru a Facebooku, kdykoliv se připojím. Vidím ... data vidím všude. Pracuji s daty přes 25 let a prostě je VIDÍM.

Pamatuji si, jak jsem si ještě jako malý kluk dělal tabulky výsledků lyžařských závodů, evidoval je a zakládal do archivu, v té době jsem je jen shromažďoval. Když jsem byl starší, dělal jsem si tabulky výsledků fotbalových zápasů většiny národních lig a snažil se předpovídat výsledky dalších zápasů. A když jsem opustil střední školu a začal pracovat, obklopil jsem se daty, nebo lépe řečeno, data mě prostě obklopila. Začalo to jako koníček a změnilo se to ve vášeň, vášeň pro data. I teď, když se na vás podívám, vidím data, data která dokážu využít. Data jsou jako čas, jsou s námi od samého počátku věku. Data plynou s časem jako nerozlučná dvojčata, stárnou, jednou rychleji a jindy pomaleji. Jedna data jsou stará a nepoužitelná už v příští vteřině, druhá mohou být cenná ještě dlouho, dokonce navždy.

To, co byla pro 19. století pára a pro 20. století ropa, jsou pro 21. století data. Tak jako pára, která je ve své podstatě sublimující voda a nemá sama o sobě teoreticky žádnou hodnotu do té doby, dokud nepřemění svou energii horké stlačené páry na energii mechanickou. Ropa, ať už ve formě benzínu či nafty, musí svým zážehem ve válcích přeměnit svou energii na rozpohybování mechanických částí motoru, aby přinesla užitek. Stejný příměr platí i pro data samotné – bez toho abychom datům přinesly hlubší význam, jsou pro nás data prozatím jen bezcenné jedničky a nuly někde na disku. Teprve zpracováním dat do podoby informací a znalostí získávají data na ceně. V následující práci se společně podíváme na to, co jsou data, jakým způsobem jsme schopni je zpracovat do takového stavu, aby se opravdu staly tím, za co jsme je na prohlásily – tzn. nové zlato 21. století.

Struktura

Struktura a osnova této práce se skládá ze dvou části. V první – teoretické – části jsou představeny aspekty vědního oboru Data Science. Druhá – praktická – část se pokouší co v největší míře na praktických příkladech představit technologii Apache Spark – nejpoužívanější technologii pro distribuované zpracování velkých objemů dat.



Obrázek 1 - Struktura diplomové práce

Teoretická část nejprve podává přehled o historii a zpracování dat a informací. Následující část pojednává o metodikách a životních cyklech v oblasti zpracování dat. Dále se pokusí vydefinovat samotný pojem Data Science jak v úzkém, tak i širším slova smyslu. V kapitole o Big Data jsou představeny faktory, které vedly k potřebě tohoto nového přístupu, pokračuje se samotnou definicí velkých dat a jejich "V" a v závěru je krátce zmíněn CAP teorém. Další kapitola představí jednotlivé typy analýz, se kterými je možné se v oblasti zpracování a využívání dat setkat. Primárním úkolem této práce není představit jednotlivé algoritmy, které se v této oblasti používají, proto se jich dotkneme jen velmi okrajově. V poslední kapitole teoretické části jsou prezentovány možnosti studia oboru Data Science v akademické sféře, zejména v České republice a nebude chybět i představení alternativní možnosti studia pomocí MOOC či jiných primárně neziskových iniciativ vzdělávaní jako *Czechitas* či *Aj Ty v IT* určené primárně pro ženy. Krátce jsou též představeny možnosti uplatnění v praxi v oblasti Data Science.

Praktická část začíná základním popisem systému Apache Spark a důvodu proč byl vybrán pro tuto práci. Krok za krokem jsou uvedeny také pokyny pro instalaci či konfiguraci. Pro krátkém představení technologie práce prezentuje každou základní komponentu této technologie a na praktických příkladech jsou představeny její hlavní smysly a účely. Pro každou konkrétní oblast existuje sada ukázkových příkladů s funkčními řádky kódů, které demonstrují použití dané technologie.

Nakonec je v závěru shrnuta tato práce. Nedílnou součástí práce je bibliografie, seznam obrázků a tabulek.

Cíle a účel

Hlavním cílem této práce je pokusit se poskytnout ctěnému čtenáři úvod do oboru Data Science a Big Data nejenom v užším slova smyslu, ale představit i oblasti které do Data Science v rámci životního cyklu dat bezpochyby náleží a který se prokládá s oborem, pro který se zažil název Data Engineering.

Dalším, neméně důležitým cílem je rozšířit mé dosavadní znalosti z oblastí Datawarehouse a Business Intelligence, resp. Data Engineeringu o následující logickou vrstvu Data Science v oblasti Big Data, abych mohl ve své konzultační praxi poskytovat širší paletu znalostí a hlubší odbornost.

Jak už bylo zmíněno, tato práce si neklade za cíl představení jednotlivých vybraných algoritmů používaných v Data Science, nakolik některé algoritmy jsou starší než autor sám, ale hlavně byly již popsány v nespočtu bakalářských, diplomových či disertačních pracích. Tato práce nebude ani prezentovat jeden velký případ použití z pohledu životního cyklu datového projektu, ačkoliv toto by mohlo být pěkným tématem dalšího diplomanta, kterého tato oblast informatiky zaujme a rozhodne se pro toto téma ve své závěrečné práci. Při prezentací příkladů se vychází čistě z technologického pohledu, a pro každou komponentu je představen příklad pro ni typicky, kdy primárním účelem je pokrytí co největšího spektra, které nabízí technologie Apache Spark.

V neposlední řadě je cílem vybudovat si technologické zázemí, aby na něm mohl autor případně navázat v doktorském studiu v oblasti Data Science a Data Engineeringu.

Jazyková forma

Hned na úvod bych rád upozornil všechny milovníky českého jazyka, že ačkoliv je tato práce napsaná v českém jazyce, tak vzhledem k tomu, že pojednává o tématu spadající do oblasti informačních technologií, kde primárním jazykem ať již v psané podobě tak i v praxi v mluvené podobě, je angličtina, nebudu se vyhýbat používání „amerikanismu“ a v těch pasážích, kde nedává smysl překládat dané slovní obraty do českého, zůstaneme u původního anglického označení. Stejně tak jako při mluveném projevu se názvy technologií či názvů nepřekládají právě z důvodu lepší pochopitelnosti a srozumitelnosti mezi objekty komunikace, tak i my se zde vyhneme tomu, abychom umělým překladem do českého jazyka nezpůsobovaly zbytečné nedorozumění anebo snižovali srozumitelnost textu. Přednost před čistě gramatickou a čistou jazykovou formou dostává forma srozumitelnosti. Text práce bude občas zklouzávat k obecnější gramatice a formulaci i odlehčením tématu, tak aby byla udržena čtenářova pozornost v částech velmi teoretických a na první pohled nezábavných. Čtivost textu a zaujmutí čtenáře bude přáním autora textu.

I. TEORETICKÁ ČÁST

1 DATA

1.1 Historie zpracování dat

Od samého počátku věku (před 2,8 miliony let) se rod *Homo* ("člověk") liší od všech ostatních druhů v okolí. Homo se vyvíjel přes *Homo habilis* (před 2,3 - 1,4 mil. let), *Homo erectus* (před 1,8 mil. - 30 000 lety), *Homo neanderthalensis* (před 400 000 - 40 000 lety) až k současné verzi nás samotných: Homo sapiens konkrétně *Homo sapiens sapiens* ("člověk moudrý") (před 280 000 lety až do současnosti). Čím se tento rod liší od ostatních? Většina druhů si své znalosti předává učením potomků od svých rodičů. Člověk je ten, kdo své znalosti předává prostřednictvím učení, ale na rozdíl od ostatních druhů, které také učí své potomky, se liší přesností přenosu. Další věcí, a rozhodně tou, která dává člověku největší výhodu, byla postupná schopnost mluvit. Odhaduje se, že zhruba před 100 000 lety začal Homo používat jazyk – nejprve základní výrazy vyjadřující bolest nebo štěstí, později podstatná jména věcí v okolní přírodě, a nakonec slovesa každodenních činností. Tento výzkum [1] naznačuje, že některé druhy rodu Homo měly schopnost vydávat zvuky řeči, které se překrývají s rozsahem zvuků řeči moderního člověka, a že druhy jako neandertálci disponovaly geny, které hrají roli v řeči člověka. Další velký milník – člověk začal zaznamenávat své informace ve formě obrázků. Nejstarší obrázky v jeskyních jsou datovány do doby před 30 000 lety. Dalším logickým krokem (pro nás v dnešní době, ne v té době) byl vývoj obrázků do symbolů, které lze označit za první systém písma používaný především k záznamu a později i ke komunikaci. Písmu předcházely nejprve piktogramy a ideogramy. Nejznámějšími příklady jsou symboly Jiahu vyryté na želvích krunýřích v Jiahu (6600 př. n. l.), Vinča symboly někdy nazývané dunajské písmo (tabulky Tărtăria, cca 5300 př. n. l.). Vynález prvních písemných systémů je datován zhruba do konce 4. tisíciletí př. n. l.. Za nejstarší písemné systémy jsou obecně považovány sumerské klínové písmo a egyptské hieroglyfy (cca 3400 až 3200 př. n. l.). Obecně se má za to, že sumerské písmo bylo nezávislým vynálezem; diskutuje se však o tom, zda se egyptské písmo vyvinulo zcela nezávisle na sumerském, nebo šlo o kulturní difúzi. Podobná debata se vede i o čínském písmu, které se vyvinulo kolem roku (1200 př. n. l.). Takzvaný moderní jazyk se v archeologických záznamech objevil teprve nedávno, s nástupem lexikografického písma zhruba před 5 400 lety [2].

Nejstaršími příklady ukládání a analýzy dat, které máme k dispozici, jsou sčítací hůlky (40000-20000 př. n. l.). Kost z Ishanga byla objevena v roce 1960 na území dnešní Ugandy a je považována za jeden z prvních důkazů pravěkého ukládání dat. Paleolitičtí kmenoví

obyvatelé si do tyčí nebo kostí vyznačovali zářezy, aby měli přehled o obchodní činnosti nebo zásobách. Tzv. vlčí kost je pravěký artefakt objevený v roce 1937 v Československu při vykopávkách ve Vestonicích na Moravě. Kost je datována do aurignacienu, tedy přibližně do doby před 30 000 lety, a je na ní 55 značek, které někteří považují za sčítací znaky. V blízkosti kosti byla vykopána hlava slonovinové figurky Venuše [3][4].

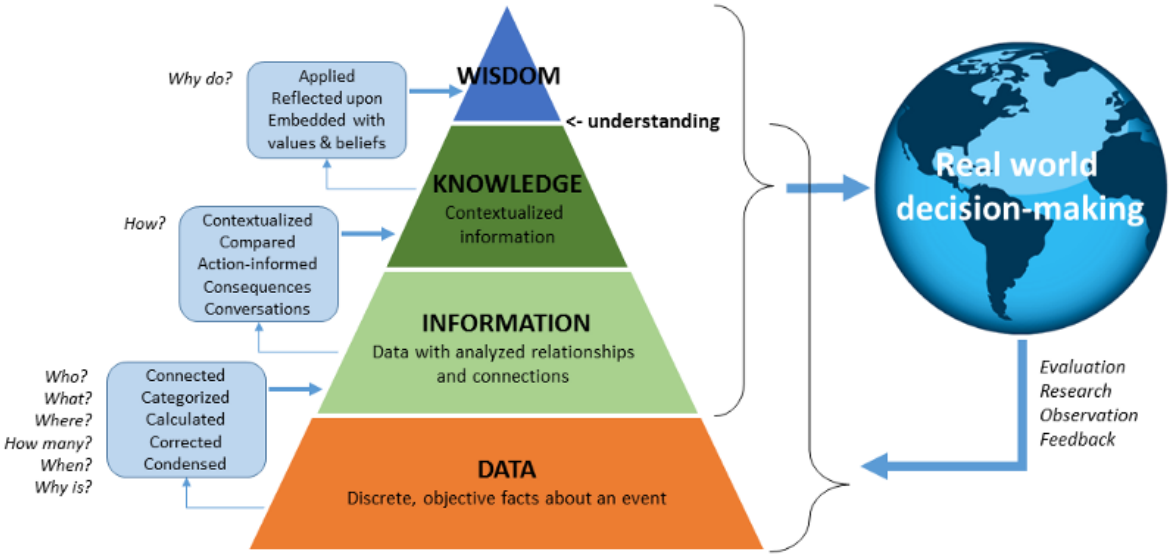
Nejstarší záznam o počítání pochází z doby kolem roku 5000 př. n. l., kdy sumerští obchodníci používali malé hliněné korálky k označování zboží pro obchod. Počítání ve větším měřítku bylo výsadou státu, kdy se vlády po tisíciletí snažily sběrem informací udržet přehled o svém obyvatelstvu. Staří Egypťané prováděli sčítání lidu, stejně jako Číňané. Zmiňuje se o nich Starý zákon a Nový zákon nám říká, že sčítání lidu nařízené císařem Augustem - "aby byl celý svět zdaněn" (Lukáš 2:1) - zavedlo Josefa a Marii do Betléma, kde se narodil Ježíš Nazaretský [5].

S oběma těmito schopnostmi (mluvit a psát stejným jazykem) začal věk dat. Schopnost vyjadřovat vlastní myšlenky vedla společenství a kultury ke shromažďování informací a sdílení a předávání znalostí a moudrosti dalším generacím [6].

1.2 Pyramida znalostí

Pro pochopení významu samotných dat je nutné začít definicí některých základních pojmů, jako jsou data, informace a znalosti, popsat jejich kontext a návaznost.

DIKW pyramida, Znalostní pyramida, Informační pyramida jsou některé z názvů odkazujících na populární znázornění vztahů mezi daty, informacemi, znalostmi a moudrostí. Každý stupeň pyramidy, graficky znázorněný na obrázku 2, odpovídá na otázky týkající se výchozích dat a přidává jim hodnotu. Čím více otázek zodpovíme, tím výše po pyramidě postupujeme. Jinými slovy, čím více obohacujeme data o význam a kontext, tím více znalostí a poznatků z nich získáváme. Na vrcholu pyramidy jsme znalosti a vhledy proměnili v učení, které řídí naše kroky.



Obrázek 2 - Znalostní pyramida [7]

Data – soubor faktů v nezpracované nebo neuspořádané podobě. V informatice se jako data chápe text, číslo, obrázek, zvuk nebo vše, co můžeme automaticky zpracovat pomocí strojů. Většinou však bez jakéhokoli kontextu mohou data znamenat jen málo.

Data lze rozdělit do různých skupin podle několika vlastností, jako je jejich struktura, původ nebo to, k čemu jsou určena. V tabulce 1 je uvedeno, jak jsou data rozdělena podle své struktury.

Tabulka 1 - Data podle struktury

Skupina	Popis	Příklad
Strukturovaná	data v předdefinované struktuře	relační data
Polo-strukturovaná	nepředddefinovaná struktura, mohou být lehce zprocesovaná	XML, JSON
Nestrukturovaná	data s nedefinovanou vnitřní strukturou	PDF, obrázky

Strukturovaná data – jsou data, jejichž prvky jsou adresovatelné pro efektivní analýzu nebo další zpracování. Data jsou uspořádána ve formátovaném úložišti (typicky databáze). Jde o všechna data, která lze uložit do relační databáze ve formě tabulky s řádky a sloupci. Mají relační klíč a lze je snadno mapovat do předem připravených polí. Tento druh dat je dnes nejčastěji zpracovávanou a snadno spravovanou informací. Strukturovaná data představují pouze 5 až 10 % všech informatických dat. Příkladem tohoto druhu dat mohou být např.: metadata (čas a datum vytvoření, velikost souboru, autor atd.), katalogy knihoven (datum,

autor, místo, předmět atd.), záznamy ze sčítání lidu (narození, příjem, zaměstnání, místo atd.), ekonomické údaje (sazby, HDP, DPH atd.).

Semi-strukturovaná data – jsou data, která nejsou uložena v relační databázi, ale mají určité organizační vlastnosti, které usnadňují jejich analýzu. Při určitém postupu je lze uložit do relační databáze, na druhou stranu u některých polostrukturovaných dat to může být velmi obtížné. Polostrukturovaná data také představují pouze 5 až 10 % všech dat v oblasti informatiky. Příkladem těchto dat mohou být např. osobní údaje uložené v souborech XML nebo JSON.

Nestrukturovaná data – jsou data, která nejsou organizována předem definovaným způsobem nebo nemají předem definovaný datový model, a proto se nehodí pro běžné relační databáze. To je jeden z důvodů, proč existují alternativní platformy pro nestrukturovaná data. Tato data stále více převládají v IT systémech a organizace je využívají v různých aplikacích business intelligence a analytických aplikacích. Odhaduje se, že dnes více než 80-95 % všech dat, která existují na celém světě, tvoří nestrukturovaná data. Příkladem těchto dat může být např. obsah generovaný uživateli ze sociálních médií (např. Facebook, Twitter, Instagram a Tumblr), obrázky, videa, data ze sledování, data ze senzorů, informace z call center, geolokační údaje, údaje o počasí, ekonomická data, vládní data a zprávy, výzkumy, trendy internetového vyhledávání a webové log soubory [8] [9].

Další možností, jak lze data rozlišit, je jejich původ, takže se dělí na externí/interní, primární/sekundární, jak je uvedeno v tabulce 2.

Tabulka 2 - Data podle původu

Skupina	Popis	Příklad
Externí	Data z veřejného sektoru, ext. organizací	Vládní data
Interní	Data pocházející interně z organizace	Objednávky
Primární	Data pocházející z daného systému	Zákazníci v CRM
Sekundární	Data získané z jiných systémů	ERP reporty

Co jsou horká, teplá a studená data?

Data lze během jejich životnosti klasifikovat pomocí teplotní stupnice. Často přistupovaná data se označují jako horká data, méně často přistupovaná data jsou teplá data a nejméně často přistupovaná data jsou studená data. Klasifikace dat obvykle závisí na obchodních pravidlech a může se v jednotlivých společnostech lišit.

- Data, která jsou potřebná k provádění každodenních obchodních činností a ke kterým se často přistupuje, se označují jako *horká* data. Data musí být optimalizována pro rychlý přístup uložením na úložiště typu Tier 0 nebo maximálně 1[1].
- Data, ke kterým se přistupuje zřídka, však musí být online, aby splňovala určitá obchodní pravidla a regulační požadavky, a označují se jako *teplá* data.
- Data, která již splnila svůj obchodní účel a nemají pro podnik žádnou vnitřní hodnotu, se označují jako *studená* data. Studená data se obvykle archivují a/anebo přímo odstraňují.

Informace – je dalším stavebním kamenem pyramidy DIKW. Jedná se o data, která byla "očištěna" od chyb a dále zpracována způsobem, který usnadňuje jejich měření, vizualizaci a analýzu pro konkrétní účel. V závislosti na tomto účelu může zpracování dat zahrnovat různé operace, jako je kombinování různých souborů dat (agregace), zajištění relevantnosti a přesnosti shromážděných dat (validace) atd. Kladením relevantních otázek na téma "kdo", "co", "kdy", "kde" atd. lze z dat získat cenné informace, které by pro nás mohly být cennější [10].

Znalost – když je položena otázka "jak", je to to, co dělá zásadní skok od informací ke znalostem. Znalost je strukturovaný souhrn vzájemně souvisejících poznatků a zkušeností z určité oblasti nebo k nějakému účelu. Má větší informační hodnotu než pouhá data nebo informace a menší informační hodnotu než moudrost.

Moudrost – jsou znalosti uplatněné v praxi. Můžeme také říci, že jestliže data a informace jsou jako pohled do minulosti, znalosti a moudrost jsou spojeny s tím, co děláme nyní a čeho chceme dosáhnout v budoucnosti [10].

[1] **Tier** 0, 1, 2, 3 je označení typu datového uložení podle rychlosti přistupu k datům z pohledu priorit. Tier 0 se používá pro vysoce výkonné transakční systémy, kde jsou vyžadovány ty nejrychlejší odezvy z diskového subsystému a naopak Tier 3 je nejpomalejší a nejlevnější typ datového subsystému např. pro zálohováni.

2 METODIKY A ŽIVOTNI CYKLY

2.1 Data Lifecycle Managememt

Správu životního cyklu dat (DLM) lze definovat jako ".. *různé fáze, kterými data procházejí během svého života od okamžiku vzniku až po zničení. Fáze životního cyklu dat zahrnují vytváření, využívání, sdílení, ukládání a mazání. Každá fáze životního cyklu dat je řízena prostřednictvím jiné sady zásad, které kontrolují ochranu dat, odolnost a dodržování předpisů* .." [11]. TechTarget definuje DLM velmi podobně [12]. Správa životního cyklu dat (DLM) je ".. *přístup založený na zásadách řízení toku dat informačního systému v průběhu jeho životního cyklu: od vytvoření a počátečního uložení až po okamžik, kdy se stanou zastaralými a jsou vymazány* ..".

Společnost Gartner zase definuje správu životního cyklu dat jako ".. *[proces] správy podnikových informací po celou dobu jejich životního cyklu, od požadavků až po vyřazení. Životní cyklus dat prochází napříč různými aplikačními systémy, databázemi a paměťovými médii. Cyklus se skládá z fází činností zahrnujících vytváření, používání, sdílení, aktualizaci, archivaci, ukládání a likvidaci. Osvědčené postupy správy dat naznačují, že je třeba, aby se každá fáze řídila rámcem, který zajišťuje co nejefektivnější podniková rozhodnutí.*" [13].

Tyto definice jsou velmi podobné, i když společnost Gartner lépe popisuje konkrétní kroky potřebné v jednotlivých fázích životního cyklu správy dat a neklade takový důraz na produkty DLM. Proč je tedy kolem tohoto termínu tolik nejasností? Protože "správa životního cyklu" může mít mnoho podob a správa životního cyklu dat se často zaměňuje se správou životního cyklu informací [13].

Produkty DLM automatizují příslušné procesy, obvykle organizují data do jednotlivých úrovní podle zadaných zásad a na základě těchto kritérií automatizují migraci dat z jedné úrovně do druhé. Novější data a data, ke kterým je třeba přistupovat častěji, se zpravidla ukládají na rychlejší, ale dražší úložná média, zatímco méně kritická data se ukládají na levnější, ale pomalejší média (viz data horká, teplá a studená a Tier 0-3).

Představme si typický život dat v organizaci. Data jsou vytvořena anebo zachycena a vložena do databáze. K těmto datům se následně přistupuje buď pro účely reportingu, analýzy, nebo pro jiné účely. Nebo zůstávají v databázi a časem zastarají. V průběhu obou těchto procesů mohou být na data aplikovány různé logické operace či validace. V určitém okamžiku však skončí jejich životnost a budou archivována, odstraněna nebo obojí.

Koncept definování a uspořádání tohoto procesu do opakovatelných kroků pro podnikové organizace je znám jako Správa životního cyklu dat (Data Lifecycle Management).

Obrázek 3 - Diagram správy životního cyklu dat [14]

1. Sběr dat

Životní cyklus dat začíná sběrem informací. To umožňuje vytvořit hodnoty, které ještě neexistují, ale které jsou potřebné v rámci činnosti podniku. Existují tři hlavní způsoby, jak lze data získat, a to je velmi důležité:

- Získávání dat: Využití existujících dat, která byla vytvořena společnostmi mimo organizaci.
- Zadávání dat: Vytváření nových dat lidmi nebo zařízeními uvnitř organizace.
- Příjem signálu: Získávání dat provedené zařízením, obvykle pomocí zařízení anebo IoT.

Mohou existovat i další způsoby, ale tři, které byly identifikovány výše, představují nejvýznamnější typy v oblasti správy dat.

2. Údržba dat

Jakmile organizace získá data, nastává fáze údržby dat neboli maintanance dat. Tu lze definovat jako poskytování dat do míst, kde dochází k syntéze a využití dat, ideálně v podobě, která je pro tento účel nejvhodnější. Data Maintenance je zpracování dat, aniž by z nich firma získala nějakou hodnotu. Často se jedná o úlohy, jako je přesun, integrace, čištění, obohacování, vyhledávání změněných dat a jsou známé jako procesy ETL/ELT.

3. Syntéza a využití dat

Tato fáze životního cyklu dat není společná všem zpracovávaným informacím, ale je důležitá v případech, kdy je třeba vytvořit z data informace a znalosti. Tento typ analýzy se používá také při modelování rizik, v účetnictví anebo při různých podnikových rozhodnutích.

Použití dat má zvláštní postavení v oblasti správy dat. Jedním z nich je, zda je legální používat data tak, jak si to přejí podnikatelé. Zde přichází na řadu pojmy jako GDPR[2] a Sarbanes-Oxley Act[3].

4. Zveřejňování dat

Využití dat informací může probíhat i mimo samotné podnikatelské prostředí. Příkladem může být situace, kdy někdo posílá měsíční zprávy svým klientům. Poté, co jsou data odeslána mimo firmu, není možné si je zapamatovat a případně opravovat chybné hodnoty. V tomto případě je potřeba, aby správa dat pomohla při rozhodování o tom, jak bude naloženo s nesprávnými daty, která byla odeslána z firmy.

[2] **GDPR** – Obecné nařízení o ochraně osobních údajů 2016/679 je nařízení v právu EU o ochraně údajů a soukromí v Evropské unii a Evropském hospodářském prostoru. Řeší také předávání osobních údajů mimo území EU a EHP [77].
[3] **Sarbanesův-Oxleyho zákon** z roku 2002 je federální zákon, který zavedl rozsáhlé předpisy pro audit a finance veřejných společností [78].

5. Ukládání dat

Ukládání neboli archivace dat je kopírování dat do prostředí, kde jsou uložena pro případ, že by byla znovu potřebná v aktivním produkčním prostředí, a odstranění těchto dat ze všech aktivních produkčních prostředí.

Archiv dat je místo, kde jsou data uložena, ale kde neprobíhá žádná údržba ani jiné další používání. V případě potřeby lze data obnovit do prostředí, kde je lze znovu používat.

6. Čištění dat

Jakmile data již nejsou pro společnost žádným způsobem užitečná, měla by být vymazána. Tento proces musí být proveden správně, aby byla zajištěna kvalitní správa dat. Je třeba si uvědomit, že po odstranění dat, zejména z archivu, jsou data nadobro ztracena a není cesty zpět.

Hlavní cíle správy životního cyklu dat

Výzvou číslo jedna, které společnosti čelí při růstu a hromadění dat, je jejich narušení či poškození, což znamená, že data je třeba efektivně spravovat po celou dobu jejich životního cyklu. Tři nejdůležitější cíle správy životního cyklu dat lze rozdělit do následujících kategorií:

- Ukládání a zabezpečení dat
  Jakmile jsou data získána, je třeba je bezpečně uložit, a tím omezit jejich zneužití. Strukturovaná data mohou být uložena v lokálních databázích nebo v cloudu, zatímco nestrukturovaná data jsou obvykle uložena na souborových serverech anebo v cloudu. Bez ohledu na to, kde jsou data uložena, musí být zabezpečena proti neoprávněnému přístupu a krádeži.
- Dostupnost dat
  Vzhledem k tomu, že podnikání je v podstatě řízeno daty, je nezbytné zajistit jejich dostupnost pro podnikání. Dostupnost zahrnuje také zpracování a vizualizaci dat podle požadavků podniku.
- Odolnost dat
  Jak data stárnou, mohou se v průběhu času měnit v důsledku úprav nebo čisticích operací. Takové činnosti mohou mít za následek také rozrůstání dat, což znamená, že stejná data

mohou existovat na více místech v mírně odlišných formách. Proto je nutné zavést proces, který zajistí integritu dat. Toto patří v současnosti k největším nešvarům i velkých korporátních společnostech, kdy data jsou „duplikována“, následně „upravována“ či „obohacována“ a když se ve finále opět potkají v systémech jako DWH či ODS, čert může tušit, která informace (zdroj dat) je ten správný – v praxi se problém nazývá “No place of true” a přáním všech kdo s daty pracují je, aby existovalo opravdu pouze jedno místo pravdy, protože je velmi těžké algoritmicky určit, které data jsou správná.

Výhody DLM pro podniky

Existuje řada důvodů, proč by podniková korporace měla chtít implementovat procesy DLM. Jsou to především následující důvody:

- Dodržování předpisů a správa

  Každé průmyslové odvětví má své vlastní předpisy pro uchovávání dat a zavedení spolehlivé strategie DLM pomáhá podnikům zachovat soulad s předpisy.
- Ochrana dat

  Z pohledu ochrany dat mají DLM a ILM svou vlastní roli. Dobrá strategie DLM nabízí redundanci, která může zajistit, že data zůstanou v případě nouze v bezpečí. Pomáhá také zajistit, aby byla data zákazníků chráněna před duplikací v různých částech datové infrastruktury, kde může být bezpečnost problémem.
- Strategie ILM[4]

  Základem ILM je DLM. Aby podniky mohly plně aplikovat strategii ILM, která udržuje data aktuální a bezpečná, musí mít nejprve funkční strategii DLM, která prochází životním cyklem dat.

[4] Správa životního cyklu informací (**ILM**) se týká strategií správy úložných systémů v počítačových zařízeních. ILM je praxe v uplatňování určitých zásad pro efektivní správu informací. ILM zahrnuje všechny fáze "záznamu" od jeho počátku až do konce. A přestože se obecně uplatňuje na informace, které odpovídají klasické definici záznamu (a souvisí tedy se správou záznamů), vztahuje se na všechna informační aktiva. Během své existence se informace může stát záznamem tím, že je identifikována jako dokumentující obchodní transakci nebo jako uspokojující obchodní potřebu. V tomto smyslu je ILM součástí celkového přístupu správy podnikového obsahu [79].

- Efektivita

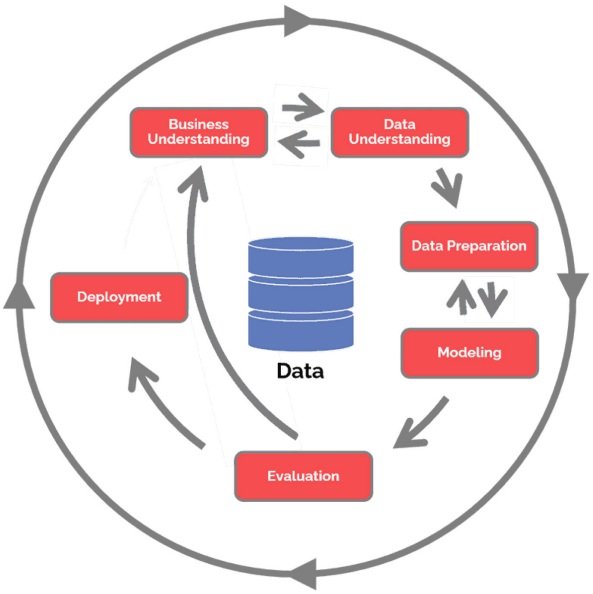
  Podstatou každého IT řešení je vyšší efektivita. Pokud jsou DLM a ILM správně implementovány v symbióze, jsou užitečná data čistá, přesná a snadno dostupná uživatelům. Tento proces pomáhá řídit automatizace. To vše pomáhá podnikům dosáhnout větší agility a efektivity.

Význam správné správy životního cyklu dat a sledování všech fází životního cyklu dat je zásadní pro velké množství činností, které společnosti denně provádějí.

Zdroje: [11][13][15][16]

2.2 CRISP-DM

Procesní model CRISP-DM je zkratka pro CRoss Industry Standard Process for Data Mining. CRISP-DM, známější pod samotnou zkratkou, je osvědčený, otestovaný a robustní standardní procesní model používaný pro projekty dolování dat a analýzy. CRISP-DM jasně znázorňuje nezbytné kroky, procesy a pracovní postupy pro realizaci jakéhokoli projektu od formalizace obchodních požadavků až po testování a nasazení řešení pro přeměnu dat na poznatky. Data Science, Data Mining a Machine Learning se snaží o spuštění několika opakujících se procesů, které mají z dat získat poznatky a informace. Proto můžeme říci, že analýza dat je skutečně uměním i vědou, protože nejde vždy o bezdůvodné spouštění algoritmů; mnoho hlavního úsilí zahrnuje pochopení podnikání, skutečné hodnoty investovaného úsilí a správných metod pro vyjádření konečných výsledků a poznatků. Model CRISP-DM nám říká, že pro vybudování komplexního řešení jakéhokoli analytického projektu nebo systému existuje celkem šest hlavních kroků nebo fází, přičemž některé z nich jsou iterativní. Stejně jako máme u projektu vývoje softwaru životní cyklus s několika hlavními fázemi nebo kroky, máme v tomto scénáři životní cyklus dolování dat nebo analýzy. Obrázek 4 znázorňuje životní cyklus dolování dat pomocí modelu CRISP-DM.



Obrázek 4 - Fáze referenčního modelu CRISP-DM [17]

Obrázek 4 přehledně ukazuje, že v životním cyklu dolování dat existuje celkem šest hlavních fází a směr postupu je znázorněn šipkami. Tento model není striktně vyžadovaný, ale spíše představuje rámec, který pomůže zajistit, abychom se při průchodu životním cyklem

jakéhokoli analytického projektu vydali správným směrem. V některých scénářích, jako je detekce anomálií nebo analýza trendů, nás může více zajímat porozumění datům, jejich průzkum a vizualizace než intenzivní modelování. Každá ze šesti fází je podrobně popsána níže.

Porozumění business problematiky

Jedná se o počáteční fázi před spuštěním jakéhokoli projektu a zároveň však o jednu z nejdůležitějších fází životního cyklu. Hlavní cíl zde začíná pochopením obchodních souvislostí a požadavků na daný problém, který má být vyřešen. Definice obchodních požadavků je klíčová pro převedení obchodního problému na problém dolování dat nebo analýzy a pro stanovení očekávání a kritérií úspěchu jak pro zákazníka, tak pro dodavatele řešení. Konečným výstupem této fáze by měl být podrobný plán s hlavními milníky projektu a očekávanými časovými harmonogramy spolu s kritérii úspěchu, předpoklady, omezeními, výhradami a možnými problémy.

Definice obchodního problému

Prvním úkolem v této fázi by bylo začít pochopením obchodního cíle, který má být řešen, a vytvoření formální definici problému. Následující body jsou klíčové pro jasné vyjádření a definování obchodního problému.

- zjištění obchodního kontextu řešeného problému, zhodnocení problému s pomocí odborníků na danou oblast (domain expert) a odborníků na danou problematiku (SME)
- popsání hlavních klíčových bodů nebo cílové oblasti obchodního cíle, který má být vyřešen
- pochopení která řešení jsou v současné době k dispozici, co jim chybí a co je třeba zlepšit
- definice obchodního cíle spolu s vhodnými výsledky a kritérii úspěchu na základě vstupů od business expertů, datových vědců či analytiků

Posouzení a analýza scénářů

Jakmile je obchodní problém jasně definován, hlavními úkoly, které s tím souvisejí, by měla být analýza a posouzení současného scénáře s ohledem na definici obchodního problému. To zahrnuje pohled na to, co je v současné době k dispozici, a zaznamenání různých

potřebných položek od zdrojů, personálu až po data. Kromě toho je třeba projednat řádné posouzení možných rizik. Hlavní kroky zahrnuté do fáze posouzení jsou zde uvedeny následovně.

- posouzení a analýza toho, co je v současné době k dispozici pro řešení problému z různých hledisek, včetně údajů, personálu, času zdrojů a rizik
- sestavení stručné zprávy o potřebných klíčových zdrojích (hardwaru i softwaru) a zapojeném personálu, v případě jakýchkoli nedostatků je třeba je zmínit a nezapomenout na ně
- ověření předpokladů a omezení na základě dostupných údajů
- zdokumentování a nahlášení možných rizik spojených s projektem, včetně časového harmonogramu, zdrojů, personálu, dat a obav založených na zdrojích financování
- nastavení kritérií úspěšnosti a v případě potřeby zdokumentování srovnávací analýzu návratnosti investic nebo nákladů oproti zhodnocení

Definice problému dataminingu

Tuto fázi lze definovat jako fázi před analýzou, která začíná, jakmile jsou definována kritéria úspěchu a obchodní problém a jsou zdokumentována všechna rizika, předpoklady a omezení. Tato fáze zahrnuje podrobné technické diskuse s analytiky, datovými vědci a vývojáři a nezbytnou koordinací a synchronizací s obchodně zainteresovanými stranami (stakeholders). Níže jsou uvedeny klíčové úkoly, které je třeba v této fázi provést.

- projednání a zdokumentování možných metod dolování dat (příp. strojového učení) vhodných pro řešení posouzením možných nástrojů, algoritmů a technik
- vypracování high-level návrhů architektury komplexního řešení
- definice, jaký bude konečný výstup z řešení a jak bude integrován se stávajícími podnikovými či IT komponentami společnosti
- definice kritéria hodnocení úspěšnosti z hlediska data science tzn. např. určení přesnosti modelu v %

Plán projektu

Jedná se o závěrečnou fázi v rámci fáze porozumění byznysu. Obvykle se vytváří plán projektu, který se skládá z celých hlavních šesti fází modelu CRISP-DM, odhadovaných

časových harmonogramů, přidělených zdrojů a personálu a možných rizik a nouzových plánů. Dbá se na to, aby pro každou fázi byly definovány konkrétní high-level výsledky a kritéria úspěchu, a pro iterativní fáze, jako je modelování, jsou doplněny poznámkami, jako je zpětná vazba pro případ, kdy bude třeba modely před nasazením přepracovat a doladit.

Dokud nejsou pokryty následující body, není možné přemýšlet na následujícími fázemi CRISP-DM.

- definice obchodních cílů problému
- kritéria úspěšnosti
- zajištění rozpočtu a plánování zdrojů
- jasně definované metodiky dolování dat případně strojového učení, které je třeba dodržovat, včetně principiálních pracovních postupů od analýzy až po nasazení
- podrobný projektový plán se všemi šesti fázemi projektu

Porozumění datům

Druhá fáze procesu CRISP-DM zahrnuje zanoření se do dostupných dat a jejich podrobnější pochopení před zahájením procesu analýzy. To zahrnuje shromáždění dat, popis různých atributů, provedení určité průzkumné analýzy dat a sledování kvality dat. Tato fáze by neměla být zanedbávána, protože špatná data nebo nedostatečné znalosti o dostupných datech mohou mít kaskádovité nepříznivé dopady v pozdějších fázích tohoto procesu.

Sběr dat

Tento úkol se provádí za účelem získání a shromáždění všech potřebných dat, která jsou nutná pro dosažení obchodního cíle. Obvykle se jedná o využití historických datových skladů (data warehouse) organizace, data skladišť (data marts) či datových jezer (data lakes) a datových sítí (data mesh) apod. Provádí se posouzení na základě stávajících dat, která jsou v organizaci k dispozici, a zda jsou zapotřebí další data. Ta lze získat z webu, tj. z otevřených zdrojů dat (open data), nebo je lze získat z jiných zdrojů, kanálů či metod, jako jsou průzkumy, nákupy, experimenty a simulace. Podrobné dokumenty by měly sledovat všechny datové soubory, které budou použity pro analýzu, a případné další zdroje dat. Tento dokument lze kombinovat s dalšími kroky této fáze.

Popis dat

Popis dat zahrnuje provedení počáteční analýzy dat s cílem lépe porozumět datům, jejich zdroji, objemu, atributům a vztahům. Jakmile jsou tyto údaje zdokumentovány, je třeba informovat příslušné pracovníky či majitele dat v podniku o případných nedostatcích, jsou-li zjištěny. Pro vytvoření správného popisu dat jsou rozhodující následující faktory.

- zdroj dat (ERP, CRM, DWH, Data Mart, Data Lakes, Data Mesh)
- formát zdroje dat (SQL, NoSQL, Big Data)
- objem dat (velikost, počet záznamů, celkový počet databází, tabulek)
- atributy dat a jejich popis (proměnné, datové typy)
- vztahy a mapovací schémata
- základní popisná statistika (průměr, medián, rozptyl, histogram)

Průzkumná analýza dat

Průzkumná analýza dat, známá také jako EDA, je jednou z prvních hlavních fází analýzy v životním cyklu. Zde je hlavním cílem podrobně prozkoumat a pochopit data. Můžeme využít popisnou statistiku, grafy, diagramy a vizualizace, abychom se podívali na různé atributy dat, našli asociace a korelace a zaznamenali případné problémy s kvalitou dat. Následují některé z hlavních úkolů v této fázi.

- prozkoumání, popsání a vizualizace dat
- výběr podmnožiny dat a atributů, které se zdají být pro daný problém nejdůležitější
- rozsáhlá analýza za účelem nalezení korelací a asociací a testování hypotéz

Analýza kvality dat

Analýza kvality dat je závěrečnou fází fáze porozumění datům, kdy analyzujeme kvalitu dat v našich souborech dat a dokumentujeme potenciální chyby, nedostatky a problémy, které je třeba vyřešit před další analýzou dat nebo zahájením modelovacích prací. Analýza kvality dat se zaměřuje především na následující.

- chybějící hodnoty
- nekonzistentní hodnoty
- chybné informace v důsledku chyb v datech (manuálních/automatických)

- chybné informace v metadatech

Příprava dat

Třetí fáze procesu CRISP-DM probíhá po získání dostatečných znalostí o obchodním problému a příslušné datové sadě. Příprava dat je především soubor úkolů, které se provádějí za účelem čištění, zpracování, úpravy a přípravy dat před spuštěním jakýchkoli analytických metod nebo metod strojového učení a sestavením modelů. V tomto oddíle stručně probereme některé hlavní úkoly spadající do fáze přípravy dat. Důležité je zde připomenout, že příprava dat je obvykle časově nejnáročnější fází životního cyklu dolování dat a často zabere 60 až 70 % času celého projektu. Tuto fázi je však třeba brát velmi vážně, protože, jak jsme již několikrát diskutovali, špatná data povedou ke špatným modelům a špatnému výkonu a výsledkům.

Integrace dat

Proces integrace dat se provádí hlavně tehdy, když máme více datových sad, které bychom mohli chtít integrovat nebo sloučit. To lze provést dvěma způsoby. Připojení několika datových sad jejich spojením, což se obvykle provádí u datových sad se stejnými atributy. Sloučení několika datových sad, které mají různé atributy nebo sloupce, pomocí společných polí jako primárních a cizích klíčů.

Předzpracování dat

Proces předzpracování dat (data wrangling nebo též data munging) zahrnuje zpracování, čištění, normalizaci a formátování dat. Data v surové podobě jsou jen zřídkakdy využitelná metodami dataminingu nebo strojového učení k vytváření modelů. Proto musíme data zpracovat na základě jejich formy, vyčistit základní chyby a nekonzistence a naformátovat je do formátů lépe využitelných pro algoritmy DM/ML. Následují hlavní úkoly související s úpravou dat.

- zpracování chybějících hodnot (odstranění či nahrazení řádků a chybějících hodnot)
- zpracování nekonzistencí dat (odstranění či nahrazení řádků a atributů, oprava nekonzistencí)

- oprava nesprávných metadat a anotací
- zpracování nejednoznačných hodnot atributů
- úprava a formátování dat do potřebných formátů (csv, json, xml)

Generování a výběr atributů

Data se skládají z pozorování nebo vzorků (řádky / rows) a atributů nebo rysů (sloupce / features). Generování atributů je v podstatě vytváření nových atributů nebo proměnných z existujících atributů na základě určitých pravidel, logiky nebo hypotéz. Jednoduchým příkladem může být vytvoření nové číselné proměnné s názvem věk na základě dvou polí s datem a časem – aktuální_datum a datum_narození – pro soubor dat zaměstnanců organizace.

Výběr atributů je v podstatě výběr podmnožiny atributů ze souboru dat na základě parametrů, jako je důležitost atributu, kvalita, relevance, předpoklady a omezení. Někdy se k výběru relevantních atributů na základě dat používají i metody strojového učení (např. PCA). V terminologii dataminingu a strojového učení je tento postup lidově označován jako výběr příznaků (feature extraction).

Modelování

Čtvrtá fáze procesu CRISP-DM je základní fází procesu, ve které probíhá většina analýz s ohledem na použití vyčištěných, zformátovaných dat a jejich atributů k vytvoření modelů pro řešení obchodních problémů. Jedná se o iterační proces, jak je znázorněno na předchozím obrázku 4, spolu s vyhodnocením modelu a všemi předchozími kroky vedoucími k modelování. Základní myšlenkou je iterativně vytvořit více modelů a snažit se dospět k nejlepšímu modelu, který splňuje naše kritéria úspěšnosti, cíle dataminingu a obchodní cíle. V této části stručně pohovoříme o některých hlavních fázích důležitých pro modelování.

Výběr technik modelování

V této fázi vybíráme seznam relevantních nástrojů dolování dat a strojového učení, frameworků, technik a algoritmů uvedených ve fázi "Porozumění business problematiky". Techniky se obvykle vybírají na základě vstupů a poznatků od datových analytiků a

datových vědců. Rozhodují o nich především aktuální dostupná data, obchodní cíle, cíle dolování dat, požadavky na algoritmus a případné omezení.

Tvorba modelu

Proces sestavování modelu se také nazývá *trénování modelu* pomocí dat a funkcí z datové sady.

Kombinace dat a algoritmů strojového učení nám dohromady dává model, který se snažíme zobecnit na trénovacích datech a poskytnout potřebné výsledky v podobě poznatků a/nebo předpovědí.

Obecně se používají různé algoritmy, které na stejných datech zkoušejí více modelovacích přístupů k řešení stejného problému, aby se získal nejlepší model, který funguje a poskytuje výstupy, které jsou nejblíže kritériím úspěšnosti projektu. Klíčovými věcmi, které je zde třeba sledovat, jsou vytvořené modely, použité (hyper-)parametry modelu a jejich výsledky.

Vyhodnocení a ladění modelu

V této fázi vyhodnocujeme každý model na základě několika metrik, jako je přesnost modelu (accuracy), přesnost (precision), odvolání (recall), F1 skóre atd. Rovněž ladíme parametry modelu na základě technik, jako je prohledávání mřížky (grid search) a křížová validace (cross validation), abychom se dostali k modelu, který nám poskytuje nejlepší výsledky. Vyladěné modely také porovnáme s cíli dolování dat, abychom zjistili, zda jsme schopni dosáhnout požadovaných výsledků i výkonu. Ladění modelu se ve světě strojového učení označuje také jako optimalizace hyperparametrů.

Posouzení modelu

Jakmile máme modely, které poskytují žádoucí a relevantní výsledky, provede se podrobné posouzení modelu na základě následujících parametrů.

- výkonnost modelu je v souladu s definovanými kritérii úspěšnosti
- reprodukovatelné a konzistentní výsledky modelů
- škálovatelnost, robustnost a snadné nasazení

- budoucí rozšiřitelnost modelu
- hodnocení modelu poskytuje uspokojivé výsledky

Hodnocení

Pátá fáze procesu CRISP-DM probíhá, jakmile máme k dispozici finální modely z fáze modelování, které splňují potřebná kritéria úspěšnosti s ohledem na naše cíle dolování dat a mají požadovaný výkon a výsledky s ohledem na metriky hodnocení modelu, jako je přesnost. Fáze hodnocení zahrnuje provedení podrobného posouzení a přezkoumání konečných modelů a výsledků, které z nich byly získány. Některé z hlavních bodů, které jsou v této části hodnoceny, jsou následující.

- hodnocení konečných modelů na základě kvality výsledků a jejich relevance na základě souladu s obchodními cíli
- náklady na nasazení celé pipeline od extrakce a zpracování dat až po modelování a předpovědi
- závěrečné návrhy, zpětná vazba a doporučení od týmu řešitelů jako bolestivá místa v celém procesu, čemu je třeba se vyhnout, na co si dát pozor

Na základě zprávy vytvořené na základě těchto bodů se tým po diskusi může rozhodnout, zda chce pokračovat v další fázi nasazení modelu, nebo je nutné úplné opakování, počínaje pochopením byznysu a dat až po modelování.

Nasazení

Závěrečná fáze procesu CRISP-DM spočívá v nasazení vybraných modelů do produkce a v zajištění bezproblémového přechodu z vývoje do produkce. Většina organizací se obvykle řídí standardní metodikou cesty do produkce. Správný plán nasazení je sestaven na základě potřebných zdrojů, serverů, hardwaru, softwaru atd. Modely jsou ověřeny, uloženy a nasazeny na potřebné systémy a servery. Zavádí se také plán pravidelného monitorování a údržby modelů, aby bylo možné průběžně vyhodnocovat jejich výkonnost, kontrolovat výsledky a jejich platnost a podle potřeby modely vyřazovat, nahrazovat a aktualizovat. Tato závěrečná fáze má čtyři části.

- plán nasazení – vypracování a zdokumentování plánu nasazení modelu/projektu
- plán monitoringu a údržby – podrobný plán monitorování a údržby, aby se předešlo problémům v produkčním prostředí
- vypracování závěrečné zprávy – shrnutí projektu, které zahrnuje závěrečnou prezentaci výsledků projektu
- přezkoumání projektu – retrospektivní zhodnocení projektu s cílem zjistit, co se povedlo, co mohlo být lepší a jak se v budoucnu zlepšit

Zdroje: [17][18][19][20][21]

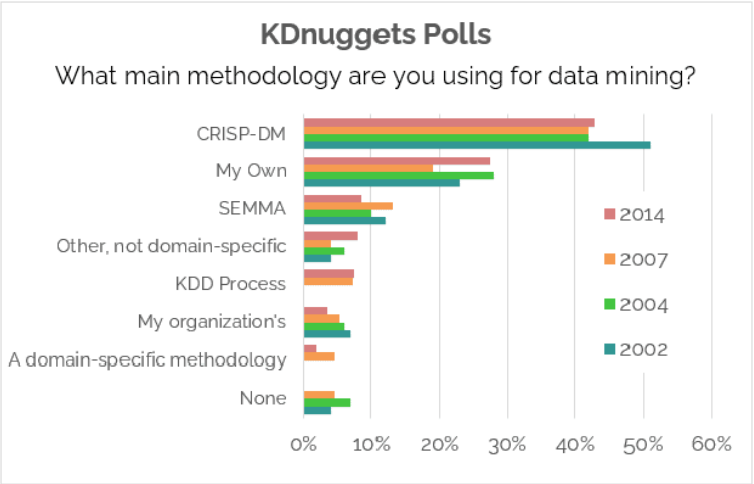
Je CRISP-DM Agile nebo Waterfall?

Je i není, záleží, jak se na metodiku díváme. Na jedné straně nám nabízí všechny vymoženosti waterfallu a na druhou stranu nám nic nenařizuje, ve své podstatě je dost iterativní. Pokud totiž budeme postupovat přesně podle CRISP-DM (na začátku projektu definujeme podrobné plány pro každou fázi a zahrneme do nich každou zprávu) a rozhodneme se, že nebudeme často iterovat, pak používáme spíše vodopádový proces. Na druhou stranu CRISP-DM nepřímo obhajuje agilní principy a postupy, když uvádí: "Pořadí fází není rigidní. Vždy je nutné přecházet mezi jednotlivými fázemi tam a zpět. Výsledek každé fáze určuje, která fáze nebo konkrétní úkol fáze musí být proveden jako další." Pokud tedy budeme postupovat podle CRISP-DM pružněji, rychle iterovat a vrstvit další agilní procesy, skončíme s agilním přístupem. Pro ilustraci toho, jak lze CRISP-DM implementovat agilním nebo vodopádovým způsobem, si představme projekt analýzy zákazníků se třemi výstupy kdy v případě agilního přístupu můžeme implementovat a dodávat jednotlivé zadání po sprintech anebo v případě vodopádového řízení jít striktně podle metodiky a nejdřív kompletně projít analýzou, následně preprocessingem dat pro všechny výstupy projektu a následně přistoupit k dalším fázím přesně podle vodopádové metodiky.

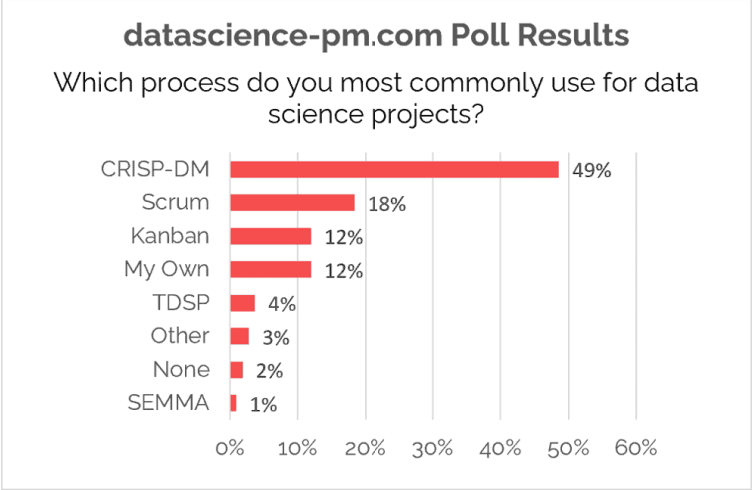
Jak je CRISP-DM populární

Neexistuje jednoznačný výzkum o tom, jak často Data Science týmy používají různé přístupy k řízení. Proto využijeme jednu z nejpopulárnějších stránek pro Data Science a Machine Learning – KDnuggets.com, a to jejich hlasování probíhající v letech 2002-2014, dále hlasování na stránce Data Science Process Alliance a na závěr Google Keyword Planner,

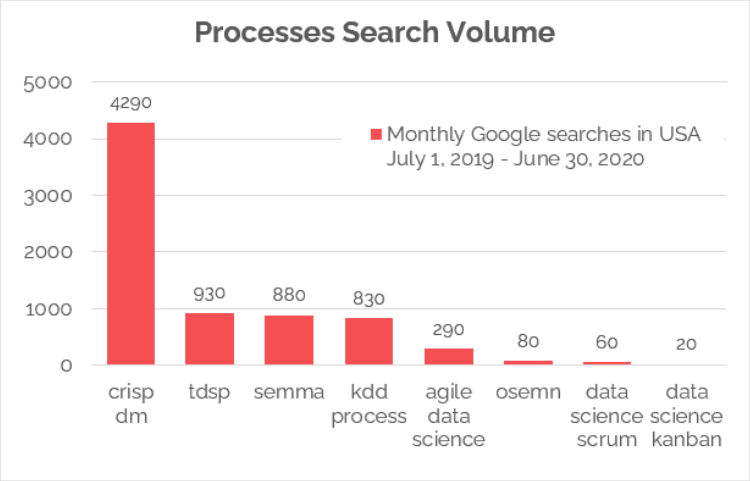
který poskytl průměrné měsíční objemy vyhledávání v USA pro vybrané klíčové výrazy a související výrazy (např. "crispdm" nebo "crisp dm data science") v letech 2019-2020.



Obrázek 5 - Hlasování o používaných metodologiích pro Datamining - kdnuggets.com (2002-2014) [17]



Obrázek 6 - Hlasování o používaných metodologiích pro Data Science - datascience-pm.com (2020) [17]



Obrázek 7 - Hlasování o používaných metodologiích pro Data Science - google.com (2019-2020) [17]

CRISP-DM pro datovou vědu?

CRISP-DM je tedy podle dotazníků stále populární a nad ostatními metodikami vede o koňskou délku. Podívejme se na jednotlivé výhody a nevýhody pro toto tvrzení.

Výhody

- Možnost zobecnění: William Vorhies, jeden z tvůrců CRISP-DM, tvrdí, že vzhledem k tomu, že všechny projekty datové vědy začínají s porozuměním podnikání, mají data, která je třeba shromáždit a vyčistit, a používají algoritmy datové vědy, "CRISP-DM poskytuje silný návod i pro ty nejpokročilejší z dnešních aktivit datové vědy" (Vorhies, 2016) [22].
- Zdravý rozum: Když byli studenti požádáni, aby provedli Data Science projekt bez vedení projektu, "přiklonili se k metodice podobné CRISPu, určili fáze a provedli několik iterací". Navíc týmy, které byly vyškoleny a bylo jim výslovně řečeno, aby implementovaly CRISP-DM, dosáhly lepších výsledků než týmy využívající jiné přístupy (Saltz, Shamshurin a Crowston, 2017) [23].
- Možnost přijetí: Stejně jako Kanban lze i CRISP-DM zavést bez velkého školení, změn organizačních rolí nebo kontroverzí.
- Jasný začátek: Počáteční zaměření na porozumění byznysu je užitečné pro sladění technické práce s obchodními potřebami a pro odvedení datových vědců od toho, aby se vrhli do problému bez řádného pochopení obchodních cílů.

- Flexibilní: Volná implementace CRISP-DM může být flexibilní a poskytovat mnoho výhod agilních principů a postupů. Tím, že uživatel akceptuje, že projekt začíná se značnými neznámými, může cyklicky procházet jednotlivými kroky a pokaždé hlouběji porozumět datům a problému. Znalosti získané v předchozích cyklech pak mohou být využity v cyklech následujících.

Slabé stránky a výzvy

- Rigidní: Na druhou stranu, CRISP-DM trpí stejnými nedostatky jako Waterfall a zatěžuje rychlé iterace.
- Objemná dokumentace: Téměř každý úkol obsahuje dokumentační krok. Ačkoli dokumentování vlastní práce je ve vyspělém procesu klíčové, požadavky na dokumentaci v CRISP-DM mohou zbytečně zpomalovat tým od skutečného poskytování přírůstků.
- Není moderní: V rozporu s Vorheisovým argumentem o trvalé relevanci CRISP-DM jiní tvrdí, že CRISP-DM jako proces, který předchází big data, "nemusí být vhodný pro projekty big data kvůli svým V" (Saltz & Shamshurin, 2016) [24].
- Nejedná se o přístup projektového řízení: Snad nejvýznamnější je, že CRISP-DM není skutečnou metodikou projektového řízení, protože implicitně předpokládá, že jejím uživatelem je jeden člověk nebo malý, úzký tým, a ignoruje koordinaci týmové práce, která je nezbytná pro větší projekty (Saltz, Shamshurin a Connors, 2017) [25].

Závěr

CRISP-DM je skvělým výchozím bodem pro ty, kteří chtějí porozumět obecnému procesu datové vědy. Doporučení, jak tyto nedostatky překonat:

- Rychlejší iterace: Důležité je nespadnout do chapadel vodopádu tím, že budeme důkladně pracovat napříč vrstvami projektu. Třeba uvažovat vertikálně a dodávat MVP[5]. První výsledek nemusí být příliš užitečný, iterace je cesta z vodopádu.

[5] Minimální životaschopný produkt (MVP) je verze produktu s dostatečným množstvím funkcí, aby jej mohli používat první zákazníci, kteří pak mohou poskytnout zpětnou vazbu pro budoucí vývoj produktu. Zaměření na vydání MVP znamená, že se vývojáři potenciálně vyhnou zdlouhavé a (v konečném důsledku) zbytečné práci [80].

- Dostatečná dokumentace ... ale ne přehnaná: Postupovat přesně podle CRISP-DM může vést ke strávení více času dokumentováním než samotnou prací na projektu. Dokumentujme to, co je rozumné a vhodné, ale s rozumem.
- Nezapomínat na moderní technologie: Pokud je to vhodné, využíváme moderních cloudových architektur a softwarových postupů, jako je správa verzí git a CI/CD[6].
- Nastavení cílů: CRISP-DM postrádá komunikační strategie se zúčastněnými stranami. Proto nezapomínejme nastavit cíle a často o nich komunikovat.
- Kombinace různých přístupů k řízení projektů: Jako obecnější tvrzení z předchozího bodu, CRISP-DM není skutečným přístupem k řízení projektů. Kombinace s jinými agilními přístupy je na snadě, např. Kanban, Scrum, Data Driven Scrum [17].

Další populární životní cykly datové vědy

Výše uvedený životní cyklus je jedním z desítek (a možná stovek), které můžeme najít on-line. Prozkoumáme některé z těch dalších populárnějších.

Životní cykly dolování dat

Následující tři klasické procesy dolování dat byly zahrnuty pod obecnou hlavičku životních cyklů datové vědy. Všechny pocházejí z devadesátých let. Jsou označovány spíše jako krátkozraké. Konkrétně proces KDD a SEMMA se zaměřují na datový problém, nikoliv na problém obchodní. Pouze zmiňovaný CRISP-DM má fázi nasazení. Žádný z nich se nezaobírá provozní fází.

- Proces KDD (Knowledge Discovery in Database): Jedná se o obecný proces objevování znalostí v datech prostřednictvím dataminingu neboli získávání vzorů a informací z velkých souborů dat pomocí strojového učení, statistiky a databázových systémů.

[6] V softwarovém inženýrství je **CI/CD** nebo CICD kombinací postupů kontinuální integrace a kontinuálního doručování nebo kontinuálního nasazování.CI/CD překlenuje rozdíly mezi vývojovými a provozními činnostmi a týmy tím, že prosazuje automatizaci při vytváření, testování a nasazování aplikací. Moderní postupy DevOps zahrnují kontinuální vývoj, kontinuální testování, kontinuální integraci, kontinuální nasazení a kontinuální monitorování softwarových aplikací v průběhu jejich životního cyklu. Praxe CI/CD neboli CI/CD pipeline tvoří páteř moderních operací DevOps [81].

- SEMMA: SAS vyvinul metodu Sample, Explore, Modify, Model, and Assess (SEMMA), která má uživatelům pomoci s orientací v nástrojích SAS Enterprise Miner pro řešení problémů dolování dat.
- CRISP-DM: CRoss Industry Structured Process for Data Mining je nejoblíbenější metodika pro projekty datové vědy a pokročilé analytiky. Má šest kroků: Pochopení byznysu, Pochopení dat, Příprava dat, Modelování, Ověření a Nasazení. Je zaměřena šířeji než SEMMA a KDD Process, ale stejně tak postrádá provozní aspekty životního cyklu produktu datové vědy.

Moderní životní cykly datové vědy

Níže uvedené životní cykly představují modernější přístupy, které jsou specifické pro datovou vědu. Stejně jako procesy dolování dat je OSEMN více zaměřen na základní problém dat. Většina ostatních, zejména Domino, se zaměřuje spíše na úplnější řešení.

- OSEMN: Zkratka OSEMN znamená Obtain, Scrub, Explore, Model, and iNterpret, tedy pětifázový životní cyklus [26].
- Microsoft TDSP: Týmový proces Data Science kombinuje mnoho moderních agilních postupů s životním cyklem podobným CRISP-DM. Má pět kroků: Pochopení byznysu, Získání a pochopení dat, Modelování, Nasazení a Přijetí zákazníkem [27].
- Životní cyklus Domino Data Labs: Jeho šest kroků je následujících: Nápad, Získávání a zkoumání dat, Výzkum a vývoj, Ověřování, Dodávka a Monitorování [28].

2.3 Data Science Maturity Models

Modely vyspělosti jsou založeny na konceptu úrovní nebo stádiích, kterými společnosti procházejí v průběhu svého vývoje. Definování úrovní je způsob, jak proměnit spektrum organizačního vývoje na jasně definované a srozumitelné kategorie.

Model vyspělosti v oblasti datové vědy (Data Science Maturity Model, DSMM) je pokusem zmapovat vývoj pokročilé analytiky v rámci organizace. Jedná se o adaptaci Capability Maturity Model (CMM)[7]. Již v roce 70. letech 20. století byl CMM pro IT organizace způsobem, jak posoudit, jak jsou vyspělé, a vytvořit plán pro postupné zlepšování svého fungování v dané oblasti.

Cílem DSMM je měřit, jak spolehlivě a udržitelně může organizace produkovat požadované výsledky z oblasti datové vědy. Koncepty spolehlivosti a udržitelnosti jsou důležité, protože začínající organizace mohou mít občas úspěch, a naopak vyspělé organizace mohou někdy selhat.

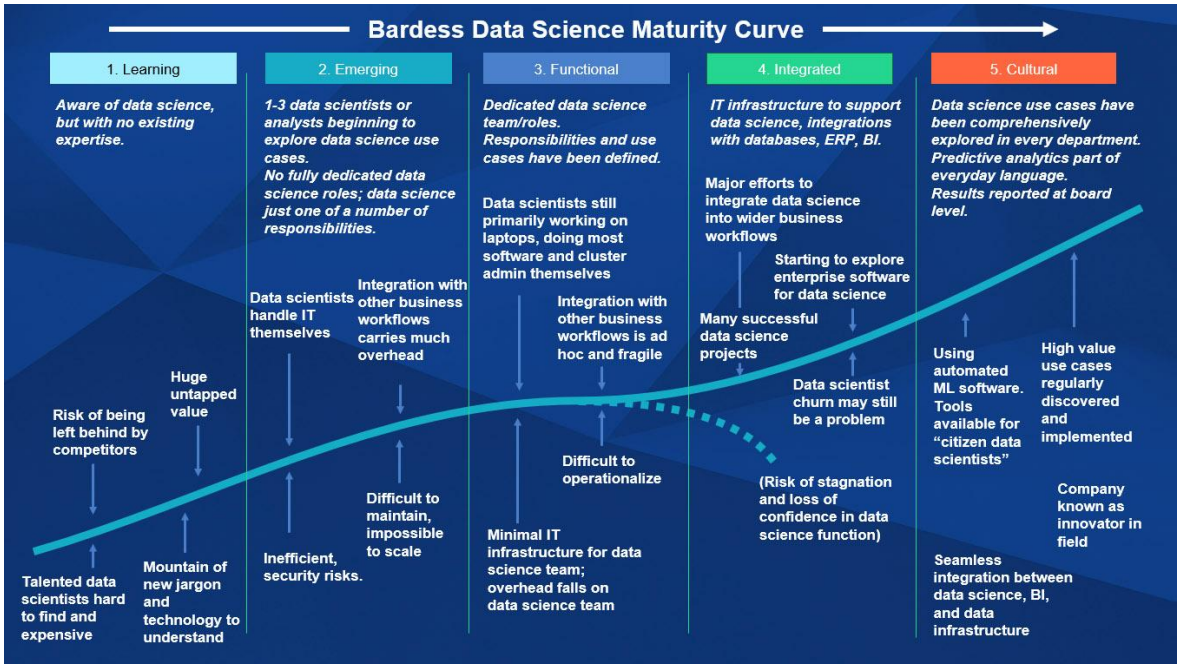
DSMM hodnotí, jak spolehlivě a udržitelně může tým pro datové vědy přinést hodnotu své organizaci. Model se skládá ze čtyř úrovní dospělosti a je rozdělen do pěti dimenzí, které platí pro všechny organizace zabývajícími se datovými či analytickými projekty. Podle návrhu není model specifický pro žádné konkrétní odvětví, tzn. je záměrně průmyslově agnostický – platí stejně tak pro banku jako pro výrobce kosmetiky. Níže uvedená matice na obrázku 8 popisuje rozměry DSMM na jednotlivých úrovních vyspělosti podle společnosti Domino Data Lab Inc [29][30].

[7] Capability Maturity Model byl původně vyvinut jako nástroj pro objektivní posouzení schopnosti procesů vládních dodavatelů implementovat smluvní softwarový projekt [82].

Level	Structured Processes	Discoverability & Compounding	Analytical Speed & Agility	Breadth & Depth of Impact	Organizational Cohesion
1 Ad Hoc Exploration	Practitioners operate autonomously in a black box	Assets stored locally, emailed around	Limited talent and tools	Ivory tower, no tangible value	Analytics island, purely transactional
2 Repeating, but Limited	Recurring workflows discussed, no enforcement	Assets stored centrally, but lack metadata / permissions	Some tools and talent investment	Static reports in a few business areas	Some collaboration with line managers
3 Defined and Controlled	Formalized process, manually enforced	Assets stored and tagged centrally with metadata and permissions	Rapidly test ideas with novel methods / tools	Results translated into multiple operational workflows	Analytics are key stakeholders in strategic decisions
4 Optimized and Automated	Best practices codified into infrastructure, transparency for all	All asset versions stored/tagged, searchable, reproducible	Cutting-edge tools, comfortable at the analytical frontier	Data products drive org with robust safeguards	Analytics enmeshed in business and proactively anticipates needs

Obrázek 8 - Data Science Maturity Model (DSMM) - Domino Data Labs [29]

DSMM není žádným průmyslovým standardem a možná proto existuje několik návrhů tohoto modelu od různých společností, spolků či zájmových sdružení. Dalším příkladem může být DSMM od společnosti Bardess [31] z roku 2019 znázorněný v obrázku 9.



Obrázek 9 - Data Science Maturity Model (DSMM) - Bardes [31]

Mezi další všeobecně známé DSMM můžeme považovat model od společnosti Oracle [32], který definuje několik dimenzí vyspělosti s pěti úrovněmi pro každou z dimenzí. Zde jsou uvedeny důležité dimenze modelu vyspělosti s cílem poskytnout nástroj pro hodnocení i potenciální plán [33]:

- Strategie – jaká je podniková strategie pro datovou vědu
- Role – jaké role jsou v podniku definovány a vytvořeny pro podporu aktivit v oblasti datové vědy
- Spolupráce – jak datoví vědci spolupracují s ostatními v podniku při vývoji a předávání pracovních produktů datové vědy
- Metodika – jaký je přístup nebo metodika podniku k projektům datové vědy
- Informovanost o datech – jak snadno se mohou datoví vědci dozvědět o datových zdrojích podniku
- Přístup k datům – jak datoví analytici a datoví vědci žádají o data a jak k nim přistupují, jak se k datům přistupuje, jak se kontrolují, spravují a monitorují
- Škálovatelnost – jak dobře se nástroje používané pro datovou vědu škálují a jak dobře fungují při zkoumání, přípravě, modelování, vyhodnocování, nasazování a spolupráci s daty
- Správa prostředků – jak jsou prostředky datové vědy spravovány a kontrolovány

- Nástroje – jaké nástroje, včetně open source, se v podniku používají pro cíle datové vědy
- Nasazení – jak snadno lze pracovní produkty datové vědy umístit do produkce, aby byly včas splněny obchodní cíle

Jak je vidět z následující tabulky 3, v poslední dekádě postupně docházelo, ať už z důvodu měnící se vyspělosti v oblasti implementace Big data v organizacích anebo prostě Data Science znělo lépe, k zřetelnému přesunu od označení Big Data/Analytics Maturity Model přes Data-driven k námi představenému Data Science Maturity Model. Klíčovým rokem v tomto posunu se stal rok 2016.

Tabulka 3 - Přehled prací v oblasti BDMM a DSMM [32]

Rok	Vydavatel	Název práce, článku, publikace
2012	Data Science Central	Big Data Analytics Maturity Model Source: IDC Business Analytics Practice (2011) https://www.datasciencecentral.com/profiles/blogs/big-data-analytics-maturity-model
2013	Data Science Central	Big Data Maturity Tool http://csc.bigdatamaturity.com
2013	IDC	Big Data maturity assessment tool http://www.infotech.com/research/it-big-data-maturity-assessment-tool
2013	Infotech	Big Data in Small Steps: Assessing the Value of Data
2014	Veenstra	Big Data maturity: An action plan for policymakers and executives.
2014	El-Darwiche	TDWI Analytics Maturity Model Guide.
2014	Halper, Stodder	Big Data Maturity Assessment https://bigdatamaturity.knowledgent.com
2014	Knowledgent	Roadmaps and Maturity Models: Pathways Toward Adopting Big Data
2014	Nott	Big Data & Analytics Maturity Model. IBM Blog. https://www.ibmbigdatahub.com/blog/big-data-analytics-maturity-model
2014	Radcliffe	Leverage a Big Data maturity model to build your Big Data roadmap
2015	Braun	Evaluation of Big Data Maturity Models: A Benchmarking Study to Support Big Data Maturity Assessment in Organizations. Tampere University of Technology.
2016	Comuzzi, Patel	How Organizations Leverage Big Data: A Maturity Model. In Industrial Management and Data Systems. 2016.
2016	Halper, Stodder	A Guide to Achieving Big Data Analytics Maturity (TDWI)
2016	Hortonworks	Big Data Maturity Model. A Hortonworks White Paper http://hortonworks.com/wp-content/uploads/2016/04/Hortonworks-Big-Data-Maturity-Assessment.pdf
2016	McKinsey Global Institute	The Age of Analytics: Competing in a Data-Driven World

Rok	Vydavatel	Název práce, článku, publikace
2016	Onis	The Four Stages of the Data Maturity Model (CIO Digital Magazine)
2016	Steele	Introducing the Data Science Maturity Model (Domino Data Labs) https://blog.dominodatalab.com/introducing-the-data-science-maturity-model/
2016	Guerra, Borne	10 Signs of Data Science Maturity https://www.oreilly.com/content/10-signs-of-data-science-maturity/
2017	Luersen	Data Warehouse in the Age of AI Maturity https://www.memsql.com/blog/memsql-maturity-framework/
2018	Hornick	A Data Science Maturity Model for Enterprise Assessment (Oracle Blog) https://blogs.oracle.com/r/a-data-science-maturity-model-for-enterprise-assessment-part-1 https://blogs.oracle.com/r/data-science-maturity-model-summary-table-for-enterprise-assessment-part-12
2018	Hornick	Data Science Maturity Model V1.3 XLSX Summary Table for Enterprise Assessment (Oracle Blog) https://app.compendium.com/api/post_attachments/9a4f0341-e293-489c-a3fe-6a5bad665029/view
2018	Physioc	How to Diagnose Your SEO Client's Search Maturity. MOZ. https://moz.com/blog/seo-client-maturity
2019	Bardess	The Bardess Data Science Maturity Curve. Bardess Blog. https://www.bardess.com/the-bardess-data-science-maturity-curve/
2019	Malchi	Six Stages of Transforming into a More Data-Driven Organization. World Wide Technology https://www.wwt.com/article/data-maturity-curve

Zajisté za jednu z nejlepších prací v této oblasti se dá považovat práce Mike Stratta z července 2020, který zahrnuje různé přístupy v DSMMs za posledních několik let do následujícího modelu [34].

Úrovně: Většina zkoumaných modelů vyspělosti datové vědy (DSMM) pokrývá 4-5 úrovní, přičemž odlehlé případy dosahují 6 nebo 7 úrovní díky tomu, že jsou úrovně rozděleny diskrétněji nebo zahrnují úroveň před uvědoměním si dat. Mnoho modelů vyspělosti pro základní datovou vědu (a přilehlé oblasti, jako jsou datové sklady, umělá inteligence, velká data atd.) používá variace nebo synonyma termínů úrovní z dřívějších modelů, a to dokonce až k modelu CMMI (Capability Maturity Model Integration), který vznikl v Carnegie Mellon v polovině 80. let 20. století.

Domény: Většina domén byla ve všech zkoumaných modelech podobná nebo se jednalo o jejich dílčí součásti. Jejich podobnost je patrná při porovnání více DSMM za účelem sladění domén. Zahrnutý model "Consensus Composite" tyto domény využívá:

- *Organizace* (povědomí, lidé, sponzoring, role, segmentace, strategie, podnik, kultura).
- *Infrastruktura* (architektura, technologie, platformy, nástroje)
- *Správa dat* ("ta" data, velikost, zdroje, složitost, metodika atd.)

- *Analytika* (co se s daty dělá, postupy, automatizované, integrované atd.)
- *Správa* (kdo kontroluje nebo zpřístupňuje přístup, bezpečnost, ochrana osobních údajů, správa aktiv atd.)
- *Osvědčené postupy* (představující nejlepší opatření dostupná na dané úrovni a/nebo opatření kritická pro růst schopností)

Atributy: Mnoho společných atributů existuje napříč přezkoumávanými modely (implementace strategie správy dat, datové projekty na úrovni oddělení atd.). Existuje však jen málo společných znaků pro to, kdy se tyto atributy vyskytují. Vzhledem k tomu, že úrovně nejsou definovány kalendářní dobou trvání, je delta mezi atributy v jednotlivých modelech nejasná. Níže uvedené atributy byly označeny za klíčové z důvodu společné povahy napříč modely a/nebo kritičnosti pro pokračování budování schopností na další úrovni.

Použití modelu: DSMM lze použít k zařazení organizace do spektra DSM prostým porovnáním nebo vyplněním jednoho z dostupných hodnocení DSMM. DSMM může pomoci identifikovat klíčové problémy v rámci každé oblasti nebo úrovně a určit zásadní kroky při budování organizačních schopností.

Tabulka 4 - Data Science Maturity Model (DSMM) – Arcalea [32]

	LEVEL 1 Ad Hoc	LEVEL 2 Foundational	LEVEL 3 Integrated	LEVEL 4 Cultural	LEVEL 5 Transformational
ORGANIZACE	Žádná strategie, která by upravovala použití datové vědy. Vedoucí pracovníci pravděpodobně nechápou sílu analytiky a zájem o ni je nízký.	Specializované týmy jsou nasetupovány, ale datová věda je stále považována za novinku. Některé experimentování s interními a externími daty s cílem zlepšit části podnikání.	Integrace probíhá napříč podnikem a širší platformou datové vědy. Vedoucí pracovníci si uvědomují, že sledování dat je klíčem k dosažení budoucích cílů.	Podnik přijímá rozhodování založené na datech. Zdroje datové vědy jsou plně podporovány.	Kulminace schopností z všech předešlých úrovní při využití datové vědy. Obvykle jde o vedoucí pracovníky, kteří definují svůj obor podnikání a jsou hybnou silou změn ve svém odvětví.
INFRASTRUKTUR	Platformy a nástroje používané pro práci s daty jsou převzaty z jiných oblastí, než je BI nebo analytika. Obvykle se jedná o malá a neúplné objemy dat.	Podpora je minimální, ale potřeba podnikové datové infrastruktury je zřejmá. V podniku existují různé zdroje dat a různé nástroje.	Postupná automatizace pracovních postupů na celopodnikové úrovni. Rozšířená skupina odborníků se snadno dostupnými datovými nástroji.	CDO zodpovědný za dohled nad daty jako firemní aktivum. Jsou zavedeny osvědčené postupy pro produkty datové vědy, a nástroje pro metadata.	Zdroje jsou neustále přehodnocovány podle případů použití, technickými inovátory a myšlenkovými lídry. Většinou fungující jako datová služba pro klienty a partnery.
SPRÁVA DAT	Neřízené strategií. Primární interní údaje zdroje jsou ve vlastnictví IT oddělení, se správou dat strategie se omezuje na vlastnictví několika databází.	Lokalizované datové sklady jsou pravděpodobné, ale omezené na stávající systémy a zdroje. Některé větší objemy dat se schopností spravovat nestrukturovaná data.	Podniková datová strategie zahrnuje identifikaci a vyhodnocení datových aktiv, je rozpoznána potřeba řešit velké objemy dat.	Sady datových nástrojů existují, které umožňují spolupráci, modelování, a sledování datových projektů. Kvantifikovatelné metriky jsou zavedeny pro hodnocení projektů.	Všechny zdroje dat zkontrolovány pro případy použití s vysokou hodnotou. Nástroje pro správu metadat jsou integrovány, formalizovány a používány.
ANALYTIKA	Omezeno na finanční, regulační a compliance údaje. Vytvoření souboru dat je pomalý proces, při kterém se používá ad hoc metodika a izolovaného pracovního úsilí.	Malé pokroky dosažené pomocí začlenění ML a prediktivní analýzy pro řešení obchodních problémů. Některé složitější reporty vyvinuty nad rámec požadavků.	Datové projekty skrze více podniky odděleními. Prediktivní analýza slouží k předvídání pravděpodobnosti konkrétních výsledků obchodních procesů.	Podniková analytika už je možná. Projekty nyní zahrnují funkce a obchodní činnosti s možností kvantifikovat hodnotu poznatků aplikované na různé obchodní činnosti.	Nejlepší metodologie a postupy pro datovou vědu integrované do všech projektů napříč organizací. Bezproblémová spolupráce a sdílení analytických dat napříč organizace.
SPRÁVA	Vlastnictví dat v kompetenci IT oddělení stejně jako vlastnictví hardwaru a technologií. Nové byznys data často přesahují možnosti IT.	V raných fázích, vedení organizace nejeví zájem. Jednotlivá oddělení prosazují své zájmy a využívají pouze svá datová sila.	Uvědomění si potřeby komplexní správa dat. Získávání nových zdrojů dat spolu s implementací zabezpečení a přístupových politik řídí inovace.	Řízení správy na úrovní celé organizace. Přístup ke zdrojům dat spravován rychle a efektivně. Datově řízené rozhodování napříč všemi obchodními jednotkami.	Integrace do všech podnikových procesů, z nichž většina je automatizovaná. Přístup na vyžádání k datovým zdrojům dostupným na místě s bezproblémovým řízením.

3 DATA SCIENCE

3.1 Definice

Datová věda umožňuje podnikům zpracovávat obrovské množství strukturovaných i nestrukturovaných velkých dat a odhalovat tak vztahy mezi daty. To následně umožňuje společnostem zvýšit efektivitu, řídit náklady, identifikovat nové tržní příležitosti a zvýšit svou tržní výhodu. Požádat osobního asistenta, jako je Alexa nebo Siri, o doporučení vyžaduje datovou vědu. Stejně tak ovládání samořiditelného automobilu, používání vyhledávače, který poskytuje užitečné výsledky, nebo rozhovor s chatbotem pro obsluhu zákazníků. To všechno jsou aplikace datové vědy v reálném životě.

Datová věda se zabývá vytěžováním rozsáhlých souborů nestrukturovaných i strukturovaných dat s cílem identifikovat vzory a získat z nich užitečné informace. Jedná se o interdisciplinární obor a základy datové vědy zahrnují statistiku, odvozování, informatiku, prediktivní analýzu, vývoj algoritmů strojového učení a nové technologie pro získávání poznatků z velkých dat.

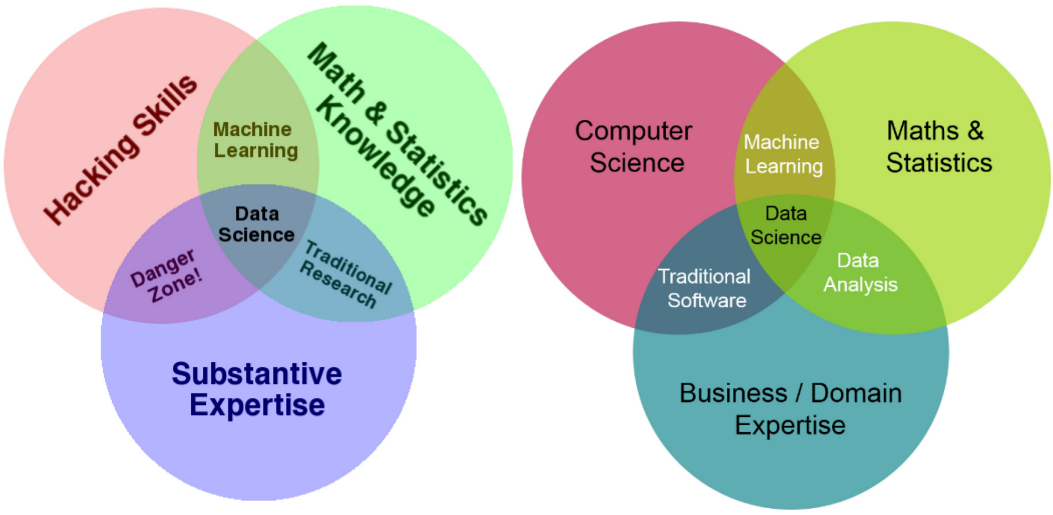
Podle wikipedie můžeme definovat datovou vědu velmi podobně: *„Datová věda je interdisciplinární obor, který využívá vědecké metody, postupy, algoritmy a systémy k získávání znalostí a poznatků ze strukturovaných i nestrukturovaných dat a k aplikaci znalostí a využitelných poznatků z dat v široké škále aplikačních oblastí. Datová věda souvisí s dolováním dat, strojovým učením a velkými daty.“* [35][36]

Datová věda je "koncept sjednocující statistiku, analýzu dat, informatiku a s nimi související metody" s cílem "pochopit a analyzovat aktuální jevy" pomocí dat. Využívá techniky a teorie čerpané z mnoha oborů v kontextu matematiky, statistiky, informatiky, informační vědy a doménových znalostí [37].

Chceme-li definovat datovou vědu a zlepšit řízení projektů datové vědy, zopakujme si její životní cyklus z minulé kapitoly. První fáze pracovního postupu datové vědy zahrnuje zachycení: získání dat, někdy jejich extrakci a zadání do systému. Další fází je údržba, která zahrnuje datové sklady, čištění dat, zpracování dat, staging dat a datovou architekturu. Následuje zpracování dat, které představuje jeden ze základů datové vědy. Právě při zkoumání a zpracování dat se datoví vědci odlišují od datových inženýrů. Tato fáze zahrnuje dolování dat, klasifikaci a shlukování dat, modelování dat a shrnutí poznatků získaných z dat – procesy, které vytvářejí efektivní data. Poté následuje analýza dat, což je stejně důležitá fáze.

Zde datoví vědci provádějí průzkumné a potvrzující práce, regresi, prediktivní analýzu, kvalitativní analýzu a vytěžování textů. V závěrečné fázi datový vědec sděluje své poznatky. To zahrnuje vizualizaci dat, jejich reportování, využívání různých nástrojů business intelligence a pomoc podnikům a dalším osobám při chytřejším rozhodování [38].

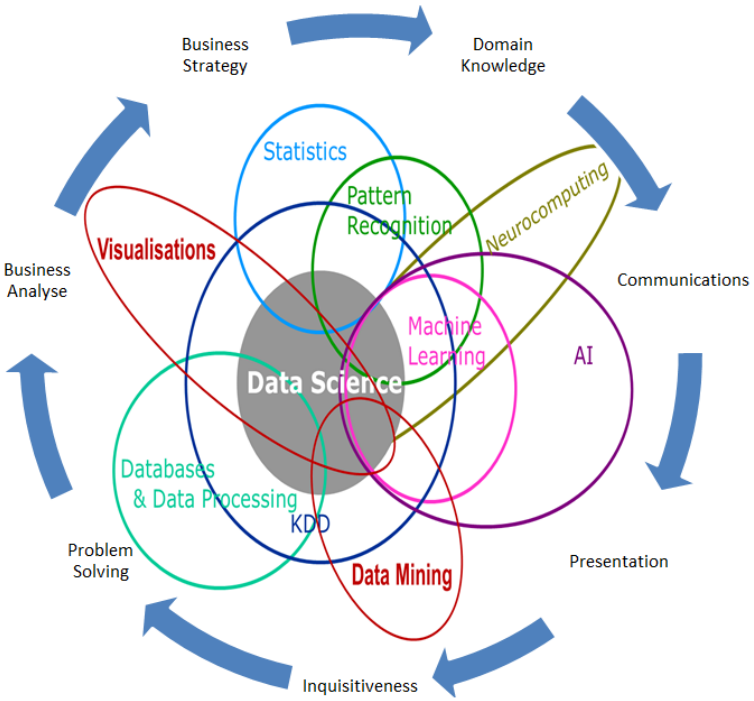
Příprava a analýza dat jsou nejdůležitějšími dovednostmi v oblasti datové vědy, ale samotná příprava dat obvykle zabere 70 až 85 procent času datového vědce. Málokdy jsou data generována v opravené, strukturované a bezšumové podobě. V tomto kroku jsou data transformována a připravena k dalšímu použití. Tato část procesu zahrnuje transformaci a vzorkování dat, kontrolu rysů i pozorování a použití statistických technik k odstranění šumu. Tento krok také osvětluje, zda jsou jednotlivé rysy v souboru dat na sobě nezávislé a zda v datech mohou chybět hodnoty. Tento krok zkoumání je také zásadním rozdílem mezi datovou vědou a datovou analytikou. Datová věda má makro pohled a jejím cílem je formulovat lepší otázky o datech, aby z nich bylo možné získat více poznatků a znalostí [38].



Obrázek 10 - Venn diagram - Data Science [39][40]

Samotná exaktní definice pro pojem Data Science / Datová věda neexistuje. V podstatě se dá ze samotných sub-definicí říct, že je to kombinace matematiky, statistiky, počítačové vědy, algoritmů a byznys doménové znalosti toho konkrétního oboru. Abychom ilustrovali složitost, komplexnost a vzájemnou provázanost jednotlivých dílčích oblasti, dovolil jsem si připojit ještě jeden ilustrativní pohled znázorněný v obrázku 11.

Abychom byli objektivní, je potřeba se na nový fenomén Data Science podívat i trochu kriticky a odpovědět si na otázku: Není to příliš povyku pro nic, anebo nejedná se o další hype v oblasti IT?



Obrázek 11 - Data Science je multidisciplinátní [41]

V říjnu 2012 vyšel v Harvard Business Review článek [42] kde Thomas Davenport označil „*datového vědce za nejvíc sexy zaměstnání 21. století*". Tento článek určitě nebyl důvodem, proč je datová věda nyní tak populární, ale jsem si jistý, že některé lidi motivoval k tomu, aby se stali datovými vědci. Je potřeba být trochu soudný a přiznat si, že IT průmysl po kriminálním a porno průmyslu patří k nejvíce vydělávajícím odvětví s nejvyššími maržemi a možností rychle z nuly vydělat velké peníze. Toto s sebou přináší situaci, že se do oboru dostávají i ti nejlepší marketéři, kteří ještě v minulém století pracovali pro bankovnictví či FMCG a v té době megaúspěšné odvětví. Stejně jako tehdy, i dnes dokážou svými marketingovými triky prodat „neprodejné" anebo přesvědčit konzumenty. že právě ta jejich technologie nebo výrobek je absolutně "must". Podíváme-li se trochu do minulosti někdy na začátek 90. let minulého století a zkusíme-li se podívat co se dělo s daty (kromě samotné informativní a uchovávací vlastnosti), tak vše začalo pojmem „sestavy" – ale žádný software s označením „Generátor sestav" není dostatečně zajímavý, aby podniky za něj utrácely peníze. Proto se alespoň v České republice začalo moderně sestavám říkat „reporty" – JZD měly své reporty, ale v kancelářích větších měst už se „sjížděli" reporty. Jak se měnily názvy

„vedoucích pracovníků“ na „manažery“, přišel na trh MIS – Manažerský informační systém, který kromě starých dobrých sestav pardon reportů, uměl přidat i grafy. Kolem roku 1997-98, kdy PC už bylo mezi společnostmi penetrováno jako něco nepostradatelného a každá firma začala shromažďovat či generovat množství dat, objevil se na trhu další, nazvěme ho, buzzword – Datamining – opět to znělo sexy, firmy chtěly využít své data a najít to, co nebylo v repotech ani MISu. No problém byl ten, že firmy nashromažďovaly cíleně data, které navíc byla nekoncepčně roztroušena po celé firmě, tak zde přišel další velký boom – tentokrát datových skladů, společnosti začínají investovat nemalé peníze do budování DWH či EDW. Zjišťuje se, že zpracování trvá příliš dlouho, a manažeři chtějí vidět výstupy okamžitě – další příležitost pro ODS[8] (Operational Data Store), která se snaží aktualizovat data v menších řádech než dny. Po obrovské explozi dat po roce 2000 způsobené fenoménem internetu se objevuje oblast Big Data – opět tu máme výzvu s ukládáním obrovského množství dat na což se následně váže jejich zpracování. A pomalinku a jistě se dostáváme k našemu pojmu Data Science. Podíváme-li se na začátek, kdy jsme měli „sestavy“ a nyní si hrajeme s Data Science – není to sexy? Je to sexy, a to se vyplatí, jak Horst z reklamy na různé serepetičky[9]. Jak jsem se pokusil vysvětlit na této analogii, vždy se jednalo o určitý typ technologického pokroku, ale než vždy vše bylo tak horké, jak se nám někdy snažily vesměs americké společnosti prezentovat. V dost velkém případě se jedná o geniální marketing, kdy reklama na prací prášek buď prezentuje, že zrovna ten jejich prášek je kvalitnější, než konkurenční anebo je více šetrný k přírodě, v případě IT šlo spíš přesvědčit o pocitu, že společnosti ujíždí technologický vlak a pokud neudělá upgrade anebo nezačne používat zrovna jejich technologii, musí zákonitě zkrachovat. Upřímně, pracoval jsem na tolika projektech, kde bylo jasné, odkud vítr vane.

V následujících kapitolách se blíže podíváme na některé tyto složky datové vědy trochu blíže, i když předem upozorňuji, že tato oblast je natolik komplexní, že bude velmi těžké vybrat to nejdůležitější a neopomenout některou oblast. Stejně tak jako v jiných oborech, i zde bude docházet k tomu, že někdo může považovat tu kterou část za méně či více důležitou

[8] ODS je databáze určená k integraci dat z různých zdrojů pro další operace s daty, pro reporting, kontroly a podporu operativního rozhodování. Na rozdíl od produkčního úložiště hlavních dat se data nepředávají zpět do provozních systémů. Mohou být předávána pro další operace a do datového skladu pro reporting.
[9] Omlouvám se, ale opravdu jsem nenašel vhodnější pojmenování. #hiddengem

a mít oprávněné výhrady, že tato práce se tím nezabývá anebo danou oblast záměrně ignoruje. Rád bych podotkl, že tomu tak záměrně není a je opravdu těžké vybrat oblasti, které je nezbytné zmínit, a navíc najít vhodnou hloubku, jak dané téma v dostupném rozsahu popsat.

3.2 Matematika a Statistika

V roce 1892 britský matematik a statistik Karl Pearson vydal svou práci „The Grammer of Science“ [43], a zanedlouho nato prohlásil, že “Statistika je gramatika vědy“, což platí zejména pro počítačové a informační vědy, fyzikální vědy a biologické vědy. Když začínáme svou cestu v oblasti Data Science nebo datové analýzy, znalosti statistiky nám pomohou lépe využívat poznatky z dat.

Statistika je obor aplikované matematiky, který se zabývá sběrem, popisem, analýzou a vyvozováním závěrů z kvantitativních dat a hromadných jevů. Matematické teorie, které stojí za statistikou, se do značné míry opírají o diferenciální a integrální počet, lineární algebru a teorii pravděpodobnosti. Statistici – lidé zaobírající se statistikou, se zabývají zejména určováním způsobů, jak vyvozovat spolehlivé závěry o velkých skupinách dat a obecných jevech z pozorovatelných charakteristik malých vzorků, které představují pouze malou část velké skupiny nebo omezený počet případů obecného jevu.

Dvě hlavní oblasti statistiky se nazývají:

- **deskriptivní statistika** (nebo též statistika popisná), která popisuje vlastnosti výběrových a populačních dat, a pomáhá nám k tomu, abychom se lépe vyznali v datech. Je běžné, že hodnoty, které jsme naměřili, mají vysokou komplexitu či složitost a je tak těžké data správně pochopit. Úkolem deskriptivní statistiky je právě tuto komplexitu zásadně zjednodušit, tak aby bylo jednodušší tyto data správně interpretovat.

  Základní dělení deskriptivní statistiky [44]:

  - Metody popisování polohy – zahrnuje pojmy jako modus, medián, průměr, které popisují, kde se zhruba hodnoty nacházejí vzhledem k celému datasetu.
  - Metody popisování rozptylu – zahrnuje pojmy jako rozptyl, směrodatná odchylka, průměrná odchylka, které popisují disperzi hodnot (jak moc se od sebe hodnoty liší).
  - Metody popisování tvaru – zahrnuje pojmy jako šikmost, špičatost, tedy jaký má distribuce hodnot tvar.
- **inferenční statistika**, která tyto vlastnosti využívá k testování hypotéz a vyvozování závěrů za pomoci výběrů (vzorků z celkové populace) s využitím teorie pravděpodobnosti a výsledek inferenční statistiky má vždy pravděpodobnostní charakter.
  Statistické postupy či metody lze zhruba rozdělit na [45]:

- exploračních analýzy (EA) - např. shluková analýza, explorační faktorová analýza, metoda hlavních komponent (PCA), metoda GUHA (kombinační analýza dat), explorační analýza dat aj
- konfirmační analýzy (KA) - např. intervaly spolehlivosti, regresní analýza, analýza rozptylu (ANOVA)

Základní rozdíl těchto dvou přístupů lze charakterizovat takto:

- V EA máme k dispozici množství dat a od EA požadujeme, aby nám z nich vygenerovala nějaké hypotézy.
- V KA formulujeme hypotézu a metody KA použijeme k tomu, abychom ji potvrdili či vyvrátili.

Zkráceně a zjednodušeně – Popisná statistika popisuje data (například graf nebo tabulka) a inferenční statistika umožňuje z těchto dat vyvozovat závěry. Při použití inferenční statistiky se vychází ze vzorků dat a provádí se zobecnění nad celou populací.

Dřív, než opustíme inferenční statistiku, pojďme si ještě jednou podívat na to, jak se od sebe liší explorační a konfirmační analýza. Využijeme zde příklad od Shelby Blitz [46].

Jak detektiv řeší případ? Ve většině případů dává dohromady všechny důkazy, které má k dispozici a všechny údaje a data, které má k dispozici, a hledá stopy a vzorce v nich ukryté. Zároveň se důkladně podívá na jednotlivé důkazy. Co podporuje jeho hypotézu? Co se vymyká trendu? Které faktory jsou v rozporu s jeho verzí? Jaké otázky ještě potřebuje zodpovědět... a co musí udělat dál, aby na ně odpověděl? Poté, když k tomu přidá své bohaté zkušenosti a intuici, vytvoří si obrázek o tom, co se skutečně stalo – a možná dokonce předpoví, co by se mohlo stát dál. Tím však příběh nekončí. Nevěříme jen detektivovu slovu, že zločin vyřešil. Jeho zjištění ho necháme přednést soudu a donutíme ho, aby to dokázala. To je ve zkratce rozdíl mezi explorativní a konfirmační analýzou. Analýza dat je široký sbor a úspěšné zvládnutí tohoto procesu zahrnuje několik kol testování, experimentování, vytváření hypotéz, ověřování a dotazování jak vašich dat, tak přístupu. Dát si dohromady případ a pak rozcupovat to, o čem si myslíte, že jsme si jisti, abychom zpochybnili své vlastní předpoklady, je pro analýzu alfou a omegou.

Průzkumná analýza dat (EDA – **Exploratory Data Analysis**) je první částí procesu analýzy dat. V této fázi je třeba udělat několik důležitých věcí, ale její podstatou je následující: zjistit, co z dat udělat, stanovit otázky, které si chceme položit, a způsob, jak je formulovat, a vymyslet nejlepší způsob prezentace a manipulace s daty, které máme k dispozici, abychom získali tyto důležité poznatky.

Jak už název napovídá, zkoumáme – hledáme stopy. Pomocí kvantitativních a vizuálních metod odhalujeme trendy a vzorce, stejně jako odchylky od modelu, odlehlé hodnoty a neočekávané výsledky. To, co nyní zjistíme, nám pomůže rozhodnout, jaké otázky položit, jaké oblasti výzkumu prozkoumat a obecně jaké další kroky podniknout.

EDA zahrnuje například: zjištění základní struktury dat, identifikaci chyb a chybějících údajů, stanovení klíčových proměnných, odhalení anomálií, kontrolu předpokladů a testování hypotéz ve vztahu ke konkrétnímu modelu, odhad parametrů, stanovení intervalů spolehlivosti a rozpětí chyb a vymyšlení "úsporného modelu" - tj. takového, který můžeme použít k vysvětlení dat s co nejmenším počtem predikčních proměnných.

Tímto způsobem se dá považovat průzkumná analýza dat za detektivní prací. Abychom ji však dokázali prosadit, potřebujeme konfirmační analýzu dat, stejně tak jako když jdeme k soudci přesvědčit ho o svých zjištěních.

Konfirmační analýza dat (CDA – **Confirmation Data Analysis**) je část, ve které vyhodnocujeme své důkazy pomocí tradičních statistických nástrojů, jako je významnost, inference a spolehlivost. V této fázi skutečně zpochybňujeme své předpoklady. Důležitou součástí konfirmační analýzy dat je kvantifikace takových věcí, jako je míra, do jaké se jakákoli odchylka od modelu, který jsme vytvořili, mohla stát náhodou, a v jakém okamžiku musíme začít svůj model zpochybňovat. CDA zahrnuje takové věci, jako je testování hypotéz, vytváření odhadů se stanovenou úrovní přesnosti, regresní analýza a analýza rozptylu. Potvrzující analýza dat je tak místem, kde podrobujeme svá zjištění a argumenty zkoušce.

Ve skutečnosti se průzkumná a konfirmační analýza dat neprovádějí jedna po druhé, ale neustále se prolínají, aby nám pomohly vytvořit co nejlepší model analýzy.

Opět si zkusme demonstrovat rozdíl mezi EDA a CDA na příkladu. Představme si, že jsme v posledních měsících zaznamenali prudký nárůst počtu uživatelů, kteří zrušili předplatné svého produktu. Chceme zjistit, proč tomu tak je, abychom mohli řešit příčinu a tento

trend zvrátit. Začneme tedy průzkumnou analýzou dat. Vzali bychom všechna data, která máme o přeběhlících (ztracených zákaznících), i o spokojených zákaznících našeho produktu, a začali bychom je procházet a hledat indicie. Po spoustě času stráveného manipulací s daty a pohledem z různých úhlů si všimneme, že naprostá většina lidí, kteří přeběhli, se zaregistrovala během stejného měsíce. Při bližším zkoumání zjistíme, že během daného měsíce náš marketingový tým přecházel na nový systém správy zákazníků a v důsledku toho ne vždy byl odeslán uvítací balíček, který obvykle posíláme novým zákazníkům. Toto by mohlo vysvětlit mnoho počátečních problémů, se kterými se noví uživatelé obvykle potýkají. Nyní máme hypotézu: lidé odcházejí, protože nedostali uvítací balíček a snadným řešením je tedy zajistit, aby uvítací balíček dostávali vždy. Nejdříve si však musíme být jisti, že jsme se v této úvaze nemýlili. Na základě analýzy průzkumných dat nyní sestavíme nový prediktivní model, který nám umožní porovnat míru přeběhlictví mezi těmi, kteří uvítací balíček obdrželi, a těmi, kteří jej neobdrželi. A to má kořeny v konfirmační analýze dat.

3.2.1 Statistika pro datovou vědu

Význam statistiky v datové vědě a datové analytice nelze podceňovat. Statistika poskytuje nástroje a metody, které umožňují nalézt strukturu a poskytnout hlubší vhled do dat. Statistika i matematika milují fakta a nesnášejí odhady. Znalost základů těchto dvou důležitých předmětů nám umožní kriticky myslet a být kreativní při využívání dat k řešení obchodních problémů a přijímání rozhodnutí založených na datech. V této kapitole se budeme zabývat tématy z oblasti statistiky jako nezbytné minimum pro datové vědce.

Náhodná proměnná

Pojem náhodných veličin je základem mnoha statistických pojmů. Jeho formální matematickou definici je možná těžké pochopit, ale jednoduše řečeno, náhodná veličina je způsob, jak výsledky náhodných procesů, jako je házení mincí nebo kostkou, vyjádřit čísly. Například náhodný proces házení mincí můžeme definovat pomocí náhodné proměnné X, která nabývá hodnoty 1, pokud je výsledek panna, a 0, pokud je výsledek orel.

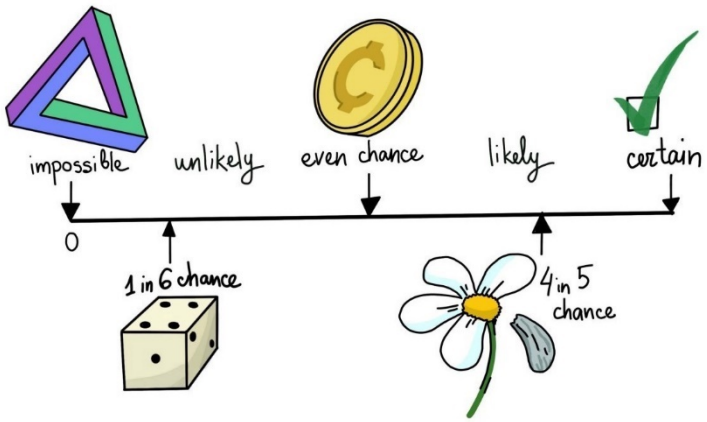
$$X = \begin{cases} 1 & pokud\ padne\ 'panna' \\ 0 & pokud\ padne\ 'orel' \end{cases}$$

V tomto příkladu máme náhodný proces házení mincí, kdy tento pokus může dát dva možné výsledky: $\{0,1\}$. Tato množina všech možných výsledků se nazývá výběrový prostor

experimentu. Každé opakování náhodného procesu se označuje jako událost. V tomto příkladu je házení mincí a získání ‚orla' jako výsledku událostí. Šance nebo pravděpodobnost výskytu této události s určitým výsledkem se nazývá pravděpodobnost této události. Pravděpodobnost události je pravděpodobnost, že náhodná veličina nabývá určité hodnoty x, kterou lze popsat vztahem $P(x)$. V příkladu házení mincí je pravděpodobnost, že padne ‚panna' nebo ‚orel', stejná, tedy 0.5 nebo 50 %. Máme tedy následující zadání:

$$P(X = panna) = 0.5$$
$$P(X = orel) = 0.5$$

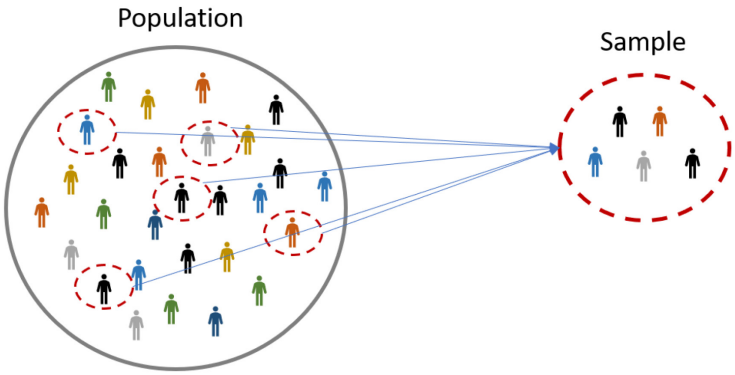
kde pravděpodobnost události může v tomto příkladu nabývat hodnot pouze v rozsahu $[0,1]$.



Obrázek 12 - Pravděpodobnost [47]

Průměr, rozptyl, směrodatná odchylka

Pro pochopení pojmů průměr, rozptyl a mnoha dalších statistických témat je důležité naučit se pojmy populace a vzorek. Populace je soubor všech pozorování (jedinců, objektů, událostí nebo postupů) a je obvykle velmi rozsáhlá a různorodá, zatímco vzorek je podmnožina pozorování z populace, která je v ideálním případě věrným obrazem populace.



Obrázek 13 - Populace vs. vzorek [48]

Vzhledem k tomu, že experimentovat s celou populací je buď nemožné, nebo prostě příliš nákladné, používají statistici nebo analytici při svých experimentech nebo pokusech spíše vzorky než celou populaci. Aby bylo jisté, že výsledky experimentu jsou spolehlivé a platí pro celou populaci, musí být vzorek věrným reprezentantem populace. To znamená, že vzorek musí být objektivní.

Průměr (Mean)

Průměr, je centrální hodnota konečného souboru čísel. Předpokládejme, že náhodná veličina X v datech má následující hodnoty:

$$x_1, x_2, x_3, \dots x_N$$

kde N je počet pozorování nebo datových bodů ve výběrovém souboru nebo jednoduše četnost dat. Pak výběrový průměr definovaný μ, který se velmi často používá k aproximaci populačního průměru, lze vyjádřit takto:

$$\mu = \frac{\sum_{i=1}^{N} x_i}{N}$$

Střední hodnota se také označuje jako *očekávání*, které je často definováno pomocí E nebo $\bar{E}$. Například *očekávání* náhodných veličin X a Y, tedy E(X), respektive E(Y), lze vyjádřit takto:

$$\bar{X} = \frac{\sum_{i=1}^{N} X_i}{N}, \bar{Y} = \frac{\sum_{i=1}^{N} Y_i}{N}$$

Rozptyl (Variance)

Rozptyl měří, jak daleko jsou datové body rozptýleny od průměrné hodnoty, a je roven součtu čtverců rozdílů mezi hodnotami dat a průměrem (střední hodnotou). Dále lze výběrový rozptyl definovaný sigma kvadrátem, který lze použít k aproximaci populačního rozptylu, vyjádřit takto:

$$\sigma^2 = \frac{\sum_{i=1}^{N}(x_i - \mu)^2}{N}$$

Směrodatná odchylka

Směrodatná odchylka je jednoduše druhou odmocninou rozptylu a měří, do jaké míry se data liší od svého průměru. Směrodatnou odchylku definovanou sigma lze vyjádřit takto:

$$\sigma = \sqrt{\sigma^2} = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \mu)^2}{N}}$$

Směrodatná odchylka se často upřednostňuje před rozptylem, protože má stejnou jednotku jako datové body, což znamená, že ji lze snadněji interpretovat.

Kovariance

Kovariance je mírou společné variability dvou náhodných veličin a popisuje vztah mezi těmito dvěma veličinami. Je definována jako očekávaná hodnota součinu odchylek obou náhodných veličin od jejich středních hodnot. Kovarianci mezi dvěma náhodnými veličinami X a Y lze popsat následujícím výrazem, kde $\bar{X}$ a $\bar{Y}$ představují střední hodnoty X a Y.

$$Cov(X, Y) = E[(X - \bar{X})(Y - \bar{Y})]$$

Kovariance může nabývat záporných nebo kladných hodnot a také hodnoty 0. Kladná hodnota kovariance naznačuje, že dvě náhodné veličiny mají tendenci měnit se stejným směrem, zatímco záporná hodnota naznačuje, že se tyto veličiny mění opačným směrem. A konečně hodnota 0 znamená, že se nemění společně.

Korelace

Korelace je také mírou vztahu a měří jak sílu, tak směr lineárního vztahu mezi dvěma proměnnými. Pokud je zjištěna korelace, znamená to, že mezi hodnotami dvou cílových proměnných existuje vztah nebo vzorec. Korelace mezi dvěma náhodnými proměnnými X a Z

se rovná kovarianci mezi těmito dvěma proměnnými vydělené součinem směrodatných odchylek těchto proměnných, což lze popsat následujícím výrazem.

$$Cor = \frac{Cov(X, Z)}{\sigma_x \sigma_z}$$

Hodnoty korelačních koeficientů se pohybují v rozmezí -1 až 1. Mějme na paměti, že korelace proměnné se sebou samou je vždy rovna 1, tedy $Cor(X, X) = 1$. Další věcí, kterou je třeba mít při interpretaci korelace na paměti, je nezaměňovat ji s příčinnou souvislostí, vzhledem k tomu, že korelace není příčinná souvislost. I když mezi dvěma proměnnými existuje korelace, nelze z toho vyvozovat, že jedna proměnná způsobuje změnu druhé. Tento vztah může být náhodný nebo může změnu obou proměnných způsobovat třetí faktor.

Hustota pravděpodobnosti

Hustota pravděpodobnosti (hustota rozdělení pravděpodobnosti, anglicky Probability Density Function, PDF) v teorii pravděpodobnosti je funkce jejíž integrací na kterémkoli vzorku (podmnožině prostoru elementárních jevů) vyjde relativní pravděpodobnost, že hodnota náhodné proměnné by se rovnala tomuto vzorku. [49]

Funkce, která popisuje všechny možné hodnoty, výběrový prostor a odpovídající pravděpodobnosti, kterých může náhodná veličina nabývat v daném rozsahu, ohraničeném mezi minimální a maximální možnou hodnotou, se nazývá distribuční funkce pravděpodobnosti nebo hustota pravděpodobnosti. Každé funkce musí splňovat následující dvě kritéria:

$$0 \leq P(X) \leq 1$$

$$\sum_{i=1}^{n} P(X_i) = 1$$

kde první kritérium říká, že všechny pravděpodobnosti by měly být čísla v rozsahu $[0,1]$ a druhé kritérium říká, že součet všech n možných pravděpodobností by měl být roven 1.

Pravděpodobnostní funkce se obvykle dělí do dvou kategorií: *diskrétní* a *spojité*. Diskrétní distribuční funkce popisuje náhodný proces s počitatelným výběrovým prostorem, jako je tomu v případě příkladu hodu mincí, který má pouze dva možné výsledky. Spojitá distribuční funkce popisuje náhodný proces se spojitým výběrovým prostorem. Příklady diskrétních distribučních funkcí jsou Bernoulliho, binomická, Poissonova, diskrétní

rovnoměrná. Příklady spojitých distribučních funkcí jsou normální, spojitá rovnoměrná, Cauchyho.

Binomické rozdělení

Binomické rozdělení je diskrétní rozdělení pravděpodobnosti počtu úspěchů v posloupnosti n nezávislých experimentů, z nichž každý má logický výsledek: úspěch (s pravděpodobností p) nebo neúspěch (s pravděpodobností $q = 1 - p$). Předpokládejme, že náhodná veličina X se řídí binomickým rozdělením, pak pravděpodobnost pozorování k úspěchů v n nezávislých pokusech lze vyjádřit následující funkcí hustoty pravděpodobnosti:

$$P_x = \binom{n}{x} p^x q^{n-x}$$

Binomické rozdělení je užitečné při analýze výsledků opakovaných nezávislých pokusů, zejména pokud nás zajímá pravděpodobnost dosažení určité prahové hodnoty při určité chybovosti.

Poissonovo rozdělení

Poissonovo rozdělení je diskrétní rozdělení pravděpodobnosti počtu událostí, které se vyskytnou v určitém časovém období, vzhledem k průměrnému počtu výskytů události v tomto časovém období. Předpokládejme, že náhodná veličina X se řídí Poissonovým rozdělením, pak pravděpodobnost pozorování k událostí za časové období lze vyjádřit následující pravděpodobnostní funkcí:

$$P_x = \frac{\lambda^x}{x!} e^{-\lambda}$$

kde e je Eulerovo číslo a λ lambda, parametr rychlosti příchodu, je očekávaná hodnota X. Poissonova distribuční funkce je velmi oblíbená pro své použití při modelování počitatelných událostí, které se vyskytnou v daném časovém intervalu.

Poissonovo rozdělení lze například použít k modelování počtu zákazníků přicházejících do obchodu mezi 16. a 18. hodinou nebo počtu pacientů přicházejících na pohotovost mezi 22. a 24. hodinou.

Normální rozdělení

Normální rozdělení pravděpodobnosti je spojité rozdělení pravděpodobnosti pro náhodnou veličinu s reálnou hodnotou. Normální rozdělení, nazývané také Gaussovo rozdělení, je pravděpodobně jednou z nejoblíbenějších distribučních funkcí, které se běžně používají ve

společenských a přírodních vědách pro účely modelování, například se používá k modelování výšky lidí nebo výsledků testů. Předpokládejme, že náhodná veličina X se řídí normálním rozdělením, pak její funkci hustoty pravděpodobnosti lze vyjádřit takto.

$$P_x = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

kde parametr μ je střední hodnota rozdělení označovaná také jako parametr polohy, parametr σ je směrodatná odchylka rozdělení označovaná také jako parametr měřítka. Číslo π je matematická konstanta.

Bayesova věta

Bayesova věta nebo často nazývaná Bayesův zákon či teorém je pravděpodobně nejmocnější pravidlo pravděpodobnosti a statistiky, pojmenované po slavném anglickém statistikovi a filozofovi Thomasi Bayesovi.

Bayesova věta je zákon pravděpodobnosti, který vnáší pojem subjektivity do světa statistiky a matematiky, kde se vše týká faktů. Popisuje pravděpodobnost události na základě předchozích informací o podmínkách, které by s touto událostí mohly souviset. Je-li například známo, že riziko onemocnění koronavirem nebo Covid-19 se zvyšuje s věkem, pak Bayesova věta umožňuje určit riziko pro jedince známého věku přesněji tím, že ho podmíníme věkem než prostým předpokladem, že je toto riziko společné pro celou populaci.

Pojem podmíněné pravděpodobnosti, který hraje v Bayesově teorii ústřední roli, je mírou pravděpodobnosti, že nastane určitá událost za předpokladu, že již nastala jiná událost. Bayesovu větu lze popsat následujícím výrazem, kde X a Y znamenají události X resp. Y:

$$P(X|Y) = \frac{P(Y|X)P(X)}{P(Y)}$$

$P(X|Y)$: pravděpodobnost, že nastane událost X za předpokladu, že nastala nebo platí událost nebo podmínka Y.

$P(Y|X)$: pravděpodobnost, že nastane událost Y za předpokladu, že nastala nebo je pravdivá událost nebo podmínka X.

$P(X)$ a $P(Y)$: pravděpodobnosti pozorování událostí X, resp. Y.

V případě předchozího příkladu je pravděpodobnost onemocnění koronavirem (událost X) podmíněná dosažením určitého věku $P(X|Y)$, která se rovná pravděpodobnosti dosažení určitého věku za předpokladu, že člověk dostal koronavirus, $P(Y|X)$, vynásobené pravděpodobností onemocnění koronavirem, $P(X)$, vydělené pravděpodobností dosažení určitého věku, $P(Y)$.

Lineární regrese

Dříve byl zaveden pojem příčinné souvislosti mezi proměnnými, která nastává, když má proměnná přímý vliv na jinou proměnnou. Je-li vztah mezi dvěma proměnnými lineární, pak je lineární regrese statistickou metodou, která může pomoci modelovat dopad jednotkové změny proměnné, nezávislé proměnné, na hodnoty jiné proměnné – závislé proměnné.

Závislé proměnné se často označují jako proměnné odezvy nebo vysvětlované proměnné, zatímco nezávislé proměnné se často označují jako represory nebo vysvětlující proměnné. Pokud je model lineární regrese založen na jediné nezávislé proměnné, pak se model nazývá jednoduchá lineární regrese, a pokud je model založen na více nezávislých proměnných, označuje se jako vícenásobná lineární regrese. Jednoduchou lineární regresi lze popsat následujícím výrazem:

$$Y_i = \beta_0 + \beta_1 X_i + u_i$$

kde Y je závislá proměnná, X je nezávislá proměnná, která je součástí dat, β_0 je intercept, který je neznámý a konstantní, β_1 je koeficient sklonu nebo parametr odpovídající proměnné X, který je rovněž neznámý a konstantní. A konečně u je chybový člen, který model vytváří při odhadu hodnot Y. Hlavní myšlenkou lineární regrese je najít nejlépe odpovídající přímku, regresní přímku, prostřednictvím souboru párových dat (X, Y).

Metoda nejmenších čtverců

Metoda nejmenších čtverců (MNČ) je metoda pro odhad neznámých parametrů, jako jsou β_0 a β_1 v lineárním regresním modelu. Model je založen na principu nejmenších čtverců, který minimalizuje součet čtverců rozdílů mezi pozorovanou závislou proměnnou a jejími hodnotami předpovídanými lineární funkcí nezávislé proměnné, často označovanými jako fitované hodnoty. Tento rozdíl mezi skutečnými a předpovídanými hodnotami závislé proměnné Y se označuje jako reziduum a to, co MNČ dělá, je minimalizace součtu čtverců

reziduí. Výsledkem tohoto optimalizačního problému jsou následující MNČ odhady neznámých parametrů β_0 a β_1, které se také nazývají odhady koeficientů.

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{N}(X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^{N}(X_i - \bar{X})^2}$$

$$\hat{\beta}_0 = \bar{Y} - \hat{\beta}_1 \bar{X}$$

Jakmile jsou tyto parametry modelu jednoduché lineární regrese odhadnuty, lze vypočítat přizpůsobené hodnoty proměnné odezvy následujícím způsobem:

$$\hat{Y}_i = \hat{\beta}_0 + \hat{\beta}_1 X_i$$

Testování statistických hypotéz

Testování hypotézy ve statistice je způsob, jak otestovat výsledky experimentu nebo průzkumu, aby se zjistilo, nakolik jsou výsledky smysluplné. V podstatě se testuje, zda jsou získané výsledky platné, a to tak, že se zjišťuje pravděpodobnost, že výsledky vznikly náhodou.

Nulová a alternativní hypotéza

Nejprve je třeba určit tezi, kterou chceme testovat, poté je třeba formulovat nulovou a alternativní hypotézu. Test může mít dva možné výsledky a na základě statistických výsledků můžeme stanovenou hypotézu buď zamítnout, nebo přijmout. Zpravidla mají statistici tendenci uvádět pod nulovou hypotézou verzi nebo formulaci hypotézy, kterou je třeba zamítnout, zatímco přijatelná a požadovaná verze je uvedena pod alternativní hypotézou.

Statistická významnost

O statistické významnosti hovoříme tehdy, když nastane taková odchylka od teoretického očekávaní, která by za platnosti předem daného předpokladu měla velmi malou pravděpodobnost. V takovém případě se má za to, že předpoklad není správný [50].

$$\begin{cases} H_0: muži\ ve\ Zlínském\ kraji\ jsou\ šťastnější\ než\ Pražáci \\ H_1: muži\ ve\ Zlínském\ kraji\ nejsou\ šťastnější\ než\ Pražáci \end{cases}$$

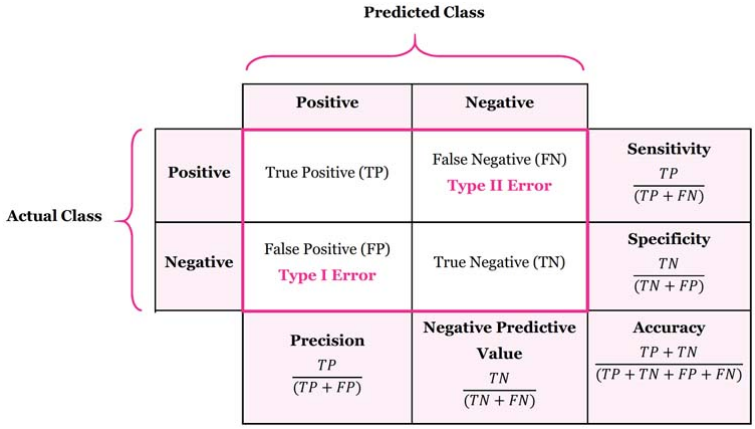
kde H_0 představuje nulovou hypotézu a H_1 alternativní hypotézu. Zamítnutí nulové hypotézy by znamenalo, že muži ve Zlínském kraji jsou šťastnější než muži v Praze. Vzhledem k tomu, že odhad parametru β_1 popisuje tento vliv nezávislé proměnné 'Vypité množství slivovice' na závislou proměnnou 'Štastnost'. Tuto hypotézu lze přeformulovat následovně:

$$\begin{cases} H_0: \beta_1 = 0 \\ H_1: \beta_1 \neq 0 \end{cases}$$

kde H_0 říká, že odhad parametru β_1 je roven 0, tj. vliv vypitého množství alkoholu na šťastný život je statisticky nevýznamný, zatímco H_0 říká, že odhad parametru β_1 není roven 0, což naznačuje, že vliv množství vypitého alkoholu na šťastný život je statisticky významný.

Chyby typu I a typu II

Při provádění statistického testování hypotéz je třeba vzít v úvahu dva koncepční typy chyb: Chybu typu I a chybu typu II. K chybě typu I dochází, když je nulová hypotéza chybně zamítnuta, zatímco k chybě typu II dochází, když nulová hypotéza chybně zamítnuta není. Matice záměn může pomoci jasně si představit závažnost těchto dvou typů chyb.



Obrázek 14 - Chybová matice [51]

- Pravdivě pozitivní (TP - True Positive) je počet správných předpovědí, že příklad je pozitivní, což znamená, že pozitivní třída byla správně identifikována jako pozitivní.
  Příklad: Daná třída je spam a klasifikátor ji správně předpověděl jako spam.
- Falešně negativní (FN – False Negative) je počet nesprávných předpovědí, že příklad je negativní, což znamená, že pozitivní třída byla nesprávně identifikována jako negativní.
  Příklad: Daná třída je spam, klasifikátor ji však nesprávně předpověděl jako nespam.
- Falešně pozitivní (FP – False Positive) je počet nesprávných předpovědí, že příklad je pozitivní, což znamená, že negativní třída byla nesprávně identifikována jako pozitivní.
  Příklad: Daná třída není spam, klasifikátor ji však nesprávně předpověděl jako spam.
- Pravdivě negativní (TN - True Negative) je počet správných předpovědí, že příklad je negativní, což znamená, že negativní třída byla správně identifikována jako negativní.
  Příklad: Daná třída je spam a klasifikátor ji správně předpověděl jako negativní.

Nyní se podíváme na některé pokročilé klasifikační metriky založené na matici záměn.

Citlivost se také označuje jako True Positive Rate nebo Recall. Je to míra pozitivních příkladů označených klasifikátorem jako pozitivní. Měla by být vyšší. Například podíl e-mailů, které jsou spam, mezi všemi spamovými e-maily.

Specifičnost je také známa jako True Negative Rate. Je to míra negativních příkladů označených klasifikátorem jako negativní. Specifičnost by měla být vysoká. Například podíl e-mailů, které nejsou spamem, mezi všemi nespamovými e-maily.

Přesnost je poměr celkového počtu správně klasifikovaných pozitivních příkladů a celkového počtu předpovězených pozitivních příkladů. Ukazuje správnost dosaženou při pozitivní predikci.

V průběhu minulého století navíc ještě několik vědců, matematiků a statistiků definovalo tzv. chybu typu III:

- „položení špatné otázky a použití chybné nulové hypotézy“ [52]
- "správné odmítnutí nulové hypotézy z chybného důvodu" (Mosteller, 1948)
- "chybu způsobenou podáním správné odpovědi na nesprávný problém" (Allyn W. Kimball, 1957)
- "chybu vyřešení nesprávného problému, zatímco bychom měli řešit správný problém" nebo jako "chybu zvolení špatné prezentace problému, zatímco bychom měli zvolit správnou prezentaci problému" (Ian Mitroff a Tom Featheringham, 1974)

V roce 1969 harvardský ekonom Howard Raiffa v žertu navrhl "kandidáta na chybu čtvrtého druhu: vyřešení správného problému příliš pozdě"). V roce 1970 L. A. Marascuilo a J. R. Levin předložili "čtvrtý typ chyby" – "chybu typu IV" – kterou definovali po Mostellerově způsobu jako omyl v "nesprávné interpretaci správně odmítnuté hypotézy"; která, jak navrhli, je ekvivalentem k "lékařově správné diagnóze nemoci, následované receptem na nesprávný lék" [53].

Statistické testy

Po stanovení nulové a alternativní hypotézy a definování testovacích předpokladů je dalším krokem určení vhodného statistického testu a výpočet testovací statistiky. Zda zamítnout či nezamítnout nulovou hypotézu, lze určit porovnáním testové statistiky s kritickou

hodnotou. Toto porovnání ukazuje, zda pozorovaná testová statistika je či není extrémnější než definovaná kritická hodnota, a může mít dva možné výsledky:

- testová statistika je extrémnější než kritická hodnota → nulovou hypotézu lze zamítnout
- testová statistika není tak extrémní jako kritická hodnota → nulovou hypotézu nelze zamítnout.

Kritická hodnota vychází z předem stanovené hladiny významnosti α (obvykle se volí rovna 5 %) a z typu rozdělení pravděpodobnosti, kterým se testovací statistika řídí. Kritická hodnota rozděluje plochu pod křivkou tohoto rozdělení pravděpodobnosti na oblast (oblasti) zamítnutí a oblast nezamítnutí. K testování různých hypotéz se používá řada statistických testů. Příklady statistických testů jsou Studentův t-test, F-test, Chí-kvadrát test, Durbinův-Hausmanův-Wuův test endogenity, Whiteův test heteroskedasticity.

Inferenční statistika

Inferenční statistika využívá výběrová data k tomu, aby bylo možné učinit přiměřené závěry o populaci, z níž výběrová data pocházejí. Používá se ke zkoumání vztahů mezi proměnnými v rámci vzorku a k vytváření předpovědí o tom, jak se tyto proměnné budou vztahovat k větší populaci.

Zákon velkých čísel (ZVČ) i Centrální limitní věta (CLM) hrají v inferenční statistice významnou roli, protože ukazují, že výsledky experimentu platí bez ohledu na to, jaký tvar mělo původní populační rozdělení, pokud jsou data dostatečně velká. Čím více dat je shromážděno, tím přesnější jsou statistické závěry, a tedy tím přesnější jsou i odhady parametrů.

Zákon velkých čísel

Předpokládejme, že $X_1, X_2, \ldots, X_n$ jsou všechny nezávislé náhodné veličiny se stejným základním rozdělením, nazývané také nezávislé identicky rozdělené, kde všechny X mají stejný průměr μ a směrodatnou odchylku σ. S rostoucí velikostí vzorku je pravděpodobnost, že průměr všech X je roven průměru μ, rovna 1.

Centrální limitní věta

Předpokládejme, že $X_1, X_2, \ldots, X_n$ jsou všechny nezávislé náhodné veličiny se stejným základním rozdělením, nazývané také nezávislé identicky rozdělené, kde všechny X mají stejný průměr μ a směrodatnou odchylku σ. S rostoucí velikostí vzorku pravděpodobnostní

rozdělení X konverguje k rozdělení v normálním rozdělení se střední hodnotou μ a rozptylem σ^2, jinak řečeno, pokud máme populaci se střední hodnotou μ a směrodatnou odchylkou σ a z této populace odebereme dostatečně velké náhodné vzorky, pak rozdělení výběrových průměrů bude blížit k normálnímu rozdělení.

Techniky snižování dimenzionality

Redukce dimenzionality je transformace dat z vysoko dimenzionálního prostoru do nízko rozměrného prostoru tak, aby tato nízko rozměrná reprezentace dat stále obsahovala co nejvíce významných vlastností původních dat.

S nárůstem popularity velkých dat vzrostla i poptávka po těchto technikách redukce dimenzionality, které snižují množství nepotřebných dat a funkcí. Příklady populárních technik redukce dimenzionality jsou Analýza hlavních komponent (PCA), Faktorová analýza (FA), Kanonická korelace, Náhodný les (RF).

Analýza hlavních komponent (PCA)

Analýza hlavních komponent neboli PCA je technika redukce dimenzionality, která se velmi často používá ke snížení dimenzionality velkých souborů dat, a to transformací velkého souboru proměnných na menší soubor, který stále obsahuje většinu informací nebo variací v původním velkém souboru dat.

Předpokládejme, že máme data X s p proměnnými: $X_1, X_2, \ldots, X_p$ s vlastními vektory $e_1, \ldots, e_p$ a vlastními hodnotami $\lambda_1, \ldots, \lambda_p$. Vlastní čísla udávají rozptyl vysvětlený určitým datovým polem z celkového rozptylu. Podstatou PCA je vytvoření nových (nezávislých) proměnných, tzv. hlavních komponent, které jsou lineární kombinací stávajících proměnných. i-tou hlavní komponentu lze vyjádřit následujícím způsobem:

$$Y_i = e_{i1}X_1 + e_{i2}X_2 + \cdots + e_{ip}X_p$$

Pak lze pomocí Elbow nebo Kaiserova pravidla určit počet hlavních komponent, které optimálně shrnují data, aniž by se ztratilo příliš mnoho informací. Důležité je také sledovat podíl celkové variability (PRTV), který je vysvětlen každou hlavní komponentou, aby bylo možné rozhodnout, zda je výhodné ji zahrnout, nebo vyloučit. PRTV pro i-tou hlavní komponentu lze vypočítat pomocí vlastních čísel takto:

$$PRTV_i = \frac{\lambda_i}{\sum_{i=1}^{p} \lambda_k}$$

Elbow pravidlo

Pravidlo lokte nebo metoda lokte je heuristický přístup, který se používá k určení počtu optimálních hlavních komponent z výsledků PCA. Podstatou této metody je vykreslit vysvětlenou variabilitu jako funkci počtu komponent a vybrat zlom (loket) křivky jako počet optimálních hlavních komponent.

Faktorová analýza (FA)

Faktorová analýza je další statistická metoda pro redukci dimenzionality. Je to jedna z nejčastěji používaných technik vzájemné závislosti a používá se v případech, kdy příslušný soubor proměnných vykazuje systematickou vzájemnou závislost a cílem je zjistit latentní faktory, které vytvářejí společnou závislost. Předpokládejme, že máme data X s p proměnnými: $X_1, X_2, \ldots, X_p$. FA model lze vyjádřit následovně:

$$X - \mu = AF + u$$

kde X je $[p\ x\ N]$ matice p proměnných a N pozorování, μ je $[p\ x\ N]$ matice populačního průměru, A je $[p\ x\ k]$ matice společných faktorových zátěží, F $[k\ x\ N]$ je matice společných faktorů a u $[p\ x\ N]$ je matice specifických faktorů. Jinak řečeno, faktorový model je tedy jako řada vícenásobných regresí, které předpovídají každou z proměnných X_i z hodnot nepozorovatelných společných faktorů f_i:

$$\begin{array}{rcl} X_1 & = & \mu_1 + a_{11}f_1 + a_{12}f_2 + \cdots + a_{1m}f_m + u_1 \\ X_2 & = & \mu_2 + a_{21}f_1 + a_{22}f_2 + \cdots + a_{2m}f_m + u_2 \\ \vdots & = & \vdots \\ X_p & = & \mu_p + a_{p1}f_1 + a_{p2}f_2 + \cdots + a_{pm}f_m + u_p \end{array}$$

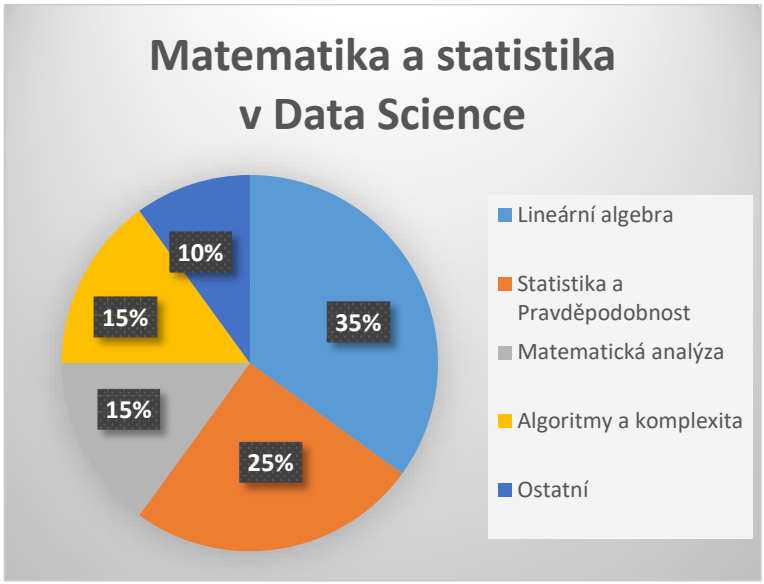
Každá proměnná má k vlastních společných faktorů a ty jsou s pozorováním spojeny prostřednictvím matice faktorového zatížení pro jedno pozorování následujícím způsobem: Při faktorové analýze se faktory počítají tak, aby se maximalizoval rozptyl mezi skupinami a zároveň minimalizoval rozptyl ve skupině. Jsou to faktory, protože seskupují základní proměnné. Na rozdíl od PCA je při FA třeba data normalizovat, vzhledem k tomu, že FA předpokládá, že soubor dat se řídí normálním rozdělením.

Závěr

V předcházejících odstavcích jsem vybral a představil opravdu jen to nezbytné minimum, které je třeba znát při práci jako datový vědec. Jeden by mohl sice namítnout, že všechny hlavní frameworky, které se používají při datových analýzách, již toto mají implementované, ale tyto frameworky slouží na zjednodušení práce a bez znalosti střev jsou danému uživateli v běžné praxi málo užitečné.

3.2.2 Matematika pro datovou vědu

Matematika je velmi důležitou součástí našeho každodenního života. Od narození až do smrti používáme matematiku vědomě či nevědomě v různých typech aplikací, které používáme, a v mnoha scénářích. V dětství se začínáme učit počítat čísla, rovnice, dělat tabulky atd. až po výpočet složitých rovnic při řešení konkrétního matematického problému.



Obrázek 15 - Matematika a statistika v Data Science [54]

Matematika získala větší význam v oblasti nejnovějších technologií, jako je strojové učení, umělá inteligence, datová věda, hluboké učení a mnoho dalších technologií. Každá nová technologie v dnešním světě přímo či nepřímo souvisí s matematikou, aby bylo možné vyvinout chytrá a jednoduchá řešení problémů. V současné době matematika vládne světu jako panovník díky svým aplikacím v rozmanitých oblastech. Matematika pomáhá v mnoha průmyslových odvětvích, například v softwarovém, lékařském, automobilovém,

designérském a robotickém průmyslu, a v mnoha dalších oblastech díky svým různým matematickým funkcím, novým technikám, tvrzením, algoritmům apod.

Matematika a statistika je jako spojena nádoba, jedna se bez druhé neobejde, a jak jsme si ukázali v předcházející kapitole jaký je význam statistiky pro data science, to same můžeme prohlásit o matematice. V krátkosti se podíváme na oblasti matematiky, které by měli patřit do základní výbavy datového vědce.

Datová věda a zejména strojové učení je o matematice, která pomáhá vytvořit model, jenž se dokáže učit z dat a přesně předpovídat. Předpověď může být tak jednoduchá, jako je klasifikace psů nebo koček z dané sady obrázků nebo jaký druh výrobků doporučit zákazníkovi na základě předchozích nákupů. Proto je velmi důležité správně pochopit matematické koncepty, které stojí za každým centrálním algoritmem strojového učení. Pomůže při výběru těch správných algoritmů pro váš projekt v oblasti datové vědy a strojového učení.

Strojové učení je primárně postaveno na matematických předpokladech, takže pokud dokážeme pochopit, proč se matematika používá, bude to pro nás zajímavější. Díky tomu pochopíme, proč vybíráme jeden algoritmus strojového učení místo druhého a jak to ovlivňuje výkonnost modelu strojového učení.

Strojové učení je založeno na čtyřech klíčových konceptech, kterými jsou statistika, pravděpodobnost, lineární algebra, a výpočet. Zatímco statistické koncepty jsou základní součástí každého modelu, kalkul nám pomáhá učit se a optimalizovat model. Lineární algebra se mimořádně hodí, když pracujeme s obrovským souborem dat, a pravděpodobnost pomáhá při předpovídání živosti událostí, které nastanou. Statistiku a pravděpodobnost jsme si představili v minulé kapitole, tak teď se velmi krátce podívejme na tu matematickou část.

Lineární algebra

Lineární algebra je odvětví matematiky, které se zabývá vektory, vektorovými prostory, soustavami lineárních rovnic a lineárními transformacemi [55]. Lineární algebra se uplatňuje v algoritmech strojového učení v oblasti ztrátových funkcí, regularizace, kovariančních matic, rozkladu singulárních hodnot (SVD), maticových operací a klasifikace pomocí SVM. Uplatňuje se také v algoritmech strojového učení, jako je lineární regrese. Vše od návrhů přátel na Facebooku, přes doporučení zboží na Amazonu až po vyhlazení selfie pomocí

hloubkového učení přenosu zahrnuje matice a maticovou algebru. Zde jsou základní témata, která je třeba pochopit:

- Základní vlastnosti matic a vektorů: skalární násobení, lineární transformace, transpozice, konjugát, hodnost, determinant.
- Vnitřní a vnější součin, pravidlo násobení matic a různé algoritmy, inverzní matice.
- Speciální matice: čtvercová matice, matice identity, trojúhelníková matice, představa o řídké a husté matici, jednotkové vektory, symetrická matice, hermitovská, šikmohermitovská a unitární matice.
- Pojem faktorizace matice/LU rozklad, Gaussova/Jordanova eliminace, řešení lineární soustavy rovnic Ax=b
- Vektorový prostor, báze, rozpětí, ortogonalita, ortonormálnost, lineární nejmenší čtverec
- Vlastní čísla, vlastní vektory, diagonalizace, rozklad na singulární hodnoty

Obrázek 16 - Skalár, Vektor, Matice, Tensor [56]

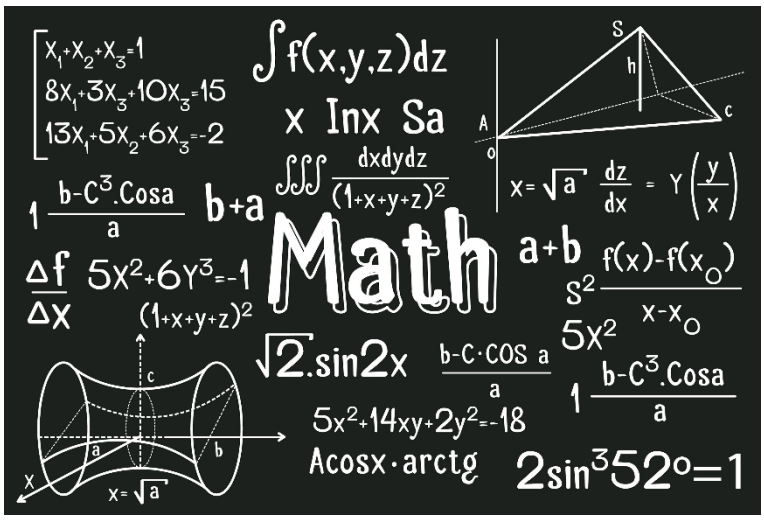
Ve strojovém učení je většina dat nejčastěji reprezentována jako vektory, matice nebo tenzory. Strojové učení se proto do značné míry opírá o lineární algebru.

- Vektor je 1D pole. Například bod v prostoru lze definovat jako vektor tří souřadnic (x, y, z). Obvykle je definován tak, že má velikost i směr.
- Matice je dvourozměrné pole čísel, které má pevný počet řádků a sloupců. Obsahuje číslo v průsečíku každého řádku a každého sloupce. Matice se obvykle označuje hranatými závorkami.
- Tenzor je zobecněním vektorů a matic. Například tenzor dimenze jedna je vektor. Kromě toho můžeme mít také tenzor dvou rozměrů, který je maticí. Pak můžeme mít trojrozměrný tenzor, například obrázek s barvami RGB. To se dále rozšiřuje na čtyřrozměrné tenzory a tak dále.

Počet resp. analýza

Kalkul(-us) neboli počet, správněji nazývaný matematická analýza, je odvětví matematiky, které se zabývá studiem rychlosti změn veličin (které lze interpretovat jako sklony křivek) a délky, plochy a objemu objektů. Kalkul se dělí na diferenciální[10] a integrální počet[11]. Slovo Calculus pochází z latiny a znamená "malý kámen", protože je to jako porozumět něčemu pohledem na malé kousky.

Počet je neodmyslitelnou součástí matematiky a zejména mnoha algoritmů strojového učení, takže není potřeba přemýšlet nad tím, zda tuto oblast studovat anebo se ji širokým obloukem vyhnout.



Obrázek 17 - Calculus [47]

Datoví vědci používají kalkul téměř pro každý model, základním, ale velmi vynikajícím příkladem kalkulu ve strojovém učení je Gradientní sestup[12].

---

[10] Diferenciální počet je matematická disciplína, která zkoumá změny funkčních hodnot v závislosti na změně nezávislé proměnné. Základním pojmem diferenciálního počtu je derivace. Pokud je derivace spojité funkce v daném bodě kladná, resp. záporná, je zde funkce rostoucí, resp. klesající. Lokální extrém může nastat pouze v bodě, ve kterém je derivace rovna nule nebo derivace neexistuje. Diferenciální počet tedy umožňuje vyšetřovat průběh funkce. Mezi další důležité pojmy diferenciálního počtu patří např. limita, diferenciál nebo spojitost [83].

[11] Integrální počet je část matematiky, která se zabývá především integrací, což je inverzní proces k derivaci, a integrály. Základním pojmem integrálního počtu je integrál. Integrály se využívají pro hledání ploch, objemů a délek křivek. Mezi další důležité pojmy integrálního počtu patří např. limita [84].

[12] Gradientní sestup (anglicky gradient descent) je iterativní optimalizační algoritmus prvního řádu pro nalezení lokálního minima diferencovatelné funkce. Myšlenkou metody je posouvat se z výchozího bodu po krocích vždy v opačném směru gradientu (nebo přibližného gradientu) funkce v daném bodě, protože to je směr nejstrmějšího klesání její hodnoty. Naopak krokování ve směru gradientu povede k lokálnímu maximu této funkce; postup je pak známý jako gradientní výstup [85].

3.3 Počítačová věda a Znalost business domény

Slušná znalost statistiky nám usnadní pochopení algoritmů strojového učení. Bez statistických pojmů bychom nedokázali vysvětlit, proč je či není lineární regrese pro danou úlohu vhodná. Do jisté míry také potřebujeme pokrýt některá témata lineární algebry a matematiky. Výpočty prováděné modely strojového učení nebo hlubokého učení zahrnují násobení matic. Abychom pochopili, jak fungují optimalizační algoritmy používané v modelech, jsou nutné určité základní znalosti matematiky.

Nestačí jen znát tato témata. Je zapotřebí je umět **implementovat**. Je tedy nevyhnutelné osvojit si programátorské dovednosti. Nemusíme být softwarovým vývojářem, ale všechny tyto algoritmy a nástroje pro analýzu dat se používají prostřednictvím programovacího jazyka. Existuje mnoho alternativ, ale nejčastěji používané programovací jazyky v datové vědě jsou zajisté Python a R. Existuje mnoho frameworků, které urychlují proces analýzy dat a strojového učení, ale k jejich používání je nutná základní úroveň programovacích dovedností.

Řekněme, že identifikujeme problém a navrhneme jeho řešení, které zahrnuje data. Data shromažďujete, čistíme a udržujeme. Vznikne užitečný a přesný model. Dalším krokem je nasazení modelu. Pokud vaše práce zůstane v notebooku jupyteru, je k ničemu. Nemůže vytvořit žádnou hodnotu. Zde přichází na řadu MLOps[13], který nám mimojiné řeší nasazování do produkčního prostředí a celkově pokrývá tuto oblast. Pokud pracujeme na středně rozsáhlém nebo velkém projektu, pravděpodobně používáme systém pro správu verzí, například Git. Navíc nám usnadní život, pokud budeme umět pohodlně pracovat v prostředí Linuxu. Samotná práce s daty vyžaduje znalost SQL a pro datovou analytiku dokonce velmi hlubokou znalost tohoto jazyka, abychom byli schopni využít různé analytické schopnosti daného jazyka – popravdě SQL by měl být v této kapitole na prvním místě. Stále více společností začíná udržovat své systémy a data v cloudu a poběží-li náš systém nad těmito daty, budou se hodit i praktické zkušenosti s cloud computingem. Oblast Data Science je velice širokou oblastí a teoreticky čím více dovedností máme, tím jsme pro firmy atraktivnější.

[13] MLOps je soubor postupů, jejichž cílem je spolehlivě a efektivně nasadit a udržovat modely strojového učení ve produkci [86].

Co se týče byznys znalostí, tady platí, že znalost dané problematiky je nanejvýš ceněnou devizou. Pokud se datový vědec orientuje např. v oblasti money-laundry, loan fraud anebo customer churn, bude přesně vědět, které data potřebuje získat pro svou práci, bude lépe a přesněji definovat hypotézy a jeho model bude zajisté přesnější.

V oblasti datové vědy nelze od sebe oddělit technické a obchodní dovednosti. Právě schopnost datových vědců vysvětlit složité algoritmy běžným lidem je to, co z nich dělá "práci desetiletí", o které se nejvíce mluví. Najít skvělého datového vědce je proto výzva, protože musí být tím nejlepším z obou světů. Jak již bylo řečeno, datový vědec by měl být schopen zvládnout zpracování dat, vytvářet užitečné modely, intuitivně chápat obchodní problémy, rozumět nuancím dat, tomu, jak model funguje, a hlavně prezentovat to zbytku světa či společnosti.

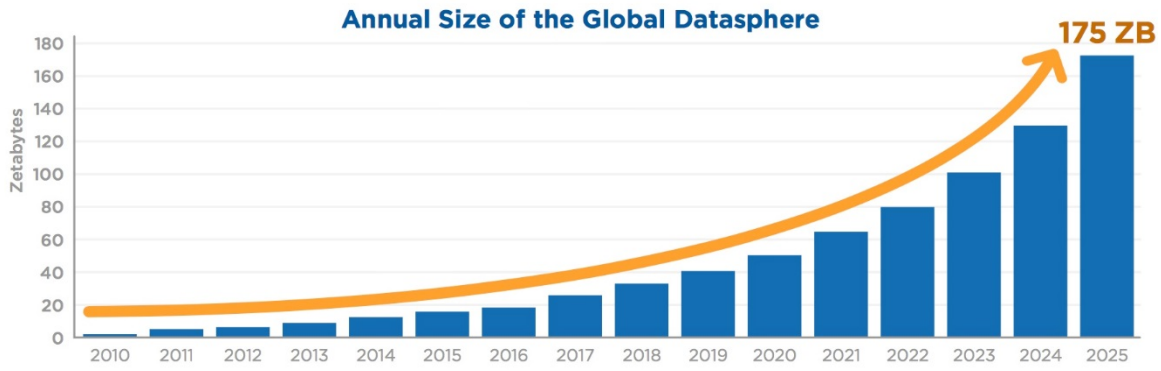
3.4 Big Data

V roce 2013 Dan Ariely přirovnal Big Data k "*Big data jsou jako sex teenagerů: všichni o tom mluví, nikdo neví, jak to dělat, všichni si myslí, že to dělají všichni ostatní, a tak všichni tvrdí, že to dělají.*" Od té doby se mnoho lidí pokoušelo toto nové slovní spojení definovat nebo zaškatulkovat, ale trh je tak rychlý a mění se, že přesnou definici dodnes nedokážeme vyjádřit. Big Data chápu jako oblast v odvětví informačních technologií, která probíhala za účelem analýzy, systematického získávání informací nebo jiného nakládání se soubory dat, které jsou příliš velké nebo příliš složité na to, aby je zvládl tradiční aplikační software pro zpracování dat ve smyslu objemu a/nebo času. Na počátku roku 2010 se také vedla velká diskuse o tom, zda jsou Big Data dalším velkým trendem a zda by se jimi firmy měly zabývat. Nyní, téměř o 10 let později můžeme s klidem prohlásit, je Big Data tu jsou, většina úspěšných společností je v určitém stupni implementace (maturity – viz Data Science Maturity Models).

Abychom měli alespoň hrubou představu, jak velká jsou Big data, níže je uvedeno přirovnání, kde jsou příklady objemu dat související s metrickými předponami.

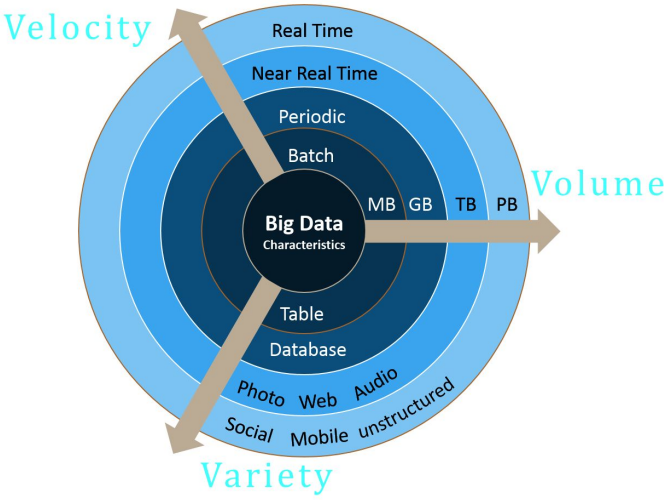
- kilo 10^3 odstavec textového dokumentu
- mega 10^6 malý román
- giga 10^9 Beethovenova 5. symfonie
- tera 10^{12} všechny rentgenové snímky ve velké nemocnici
- peta 10^{15} polovina všech akademických výzkumných knihoven v USA
- exa 10^{18} 20 % slov, která kdy člověk pronesl
- zetta 10^{21} zrnek písku na všech plážích světa

Většina z nás si pravděpodobně dokáže představit velikost mega nebo giga, možná někteří z nás dokonce pracovali s daty o objemu terabajtů, ale petabajty jsou pro většinu lidí něco jako 4. rozměr v euklidovské geometrii. Pro lepší představu je třeba podívat se na objem dat z jiného úhlu pohledu nebo jeden z rozměrů – čas – zmenšit na menší měřítko. Abychom měli představu o rychlosti nárůstu dat z pohledu času, podívejme se na obrázku 18, který nám demonstruje nárůst objemu dat v dlouhodobém časovém horizontu, kde je zobrazen objem dat za posledních 10 let s předpovědí na dalších 5 let. Jakákoli další předpověď, zejména v odvětví IT, by byla pouhým věštěním z křišťálové koule.



Obrázek 18 - Nárůst objemu dat v časovém období 2010 - 2025 [57]

V souvislosti s velkým množstvím dat se často hovoří o tzv. 3V, které jsou obecně uznávány jako charakteristiky Big data (grafické znázornění těchto charakteristik viz obrázek 19). Tato charakteristická 3 V definoval v roce 2001 Dough Laney, analytik společnosti Gartner [11].



Obrázek 19 - Big Data 3V [58]

Volume - Objem - velikost dat

Prvním "V" je objem, tj. velikost zpracovávaných dat. Se stále rostoucím objemem dat generovaných moderními technologiemi vyvstala otázka, jak tato data zpracovat, tak, aby výsledkem analýzy těchto dat byly co nejpřesnější výsledky v co nejkratším čase. Takový objem dat nelze zpracovat tradičními databázovými nástroji. Jaká velikost dat již znamená "velká", není nikde řečeno, ale tato hranice se s vývojem technologií neustále posouvá.

Velocity - Rychlost - rychlost přenosu dat

Druhým "V" je rychlost, s jakou data vznikají. S rostoucí velikostí dat se zvyšuje rychlost dat, která organizace produkuje, zpracovává nebo jen přijímá. Schopnost rychle přijímat data a reagovat na ně může být významnou konkurenční výhodou nebo dokonce nutností. Aby však organizace mohla na data rychle reagovat, je nutné rychlé zpracování dat. Pro tradiční databázové nástroje je velmi obtížné zpracovávat nestrukturovaná data nebo strukturovaná data velkých objemů, náklady na takové zpracování a ukládání by byly neúnosné. Naproti tomu technologie pro zpracování velkých objemů dat takové ukládání a zpracování umožňují a jsou k tomu určeny.

Variety - Různorodost - rozmanitost dat

Třetí "V" představuje rozmanitost dat. Data málokdy pocházejí z jednoho zdroje a často nejsou v jednotném formátu. Tato rozmanitost dat je zásadním rozdílem oproti tradičním metodám ukládání dat, kde se používají data s pevnou strukturou – strukturovaná data. Strukturovaná data jsou popsána svými metadaty a jejich analýzu lze snadno provádět pomocí tradičních databázových systémů. Opačným příkladem jsou nestrukturovaná data, která nejsou popsána ani uspořádána, takže je není snadné zpracovávat. Tento typ dat se vyskytuje například v e-mailech, dokumentech nebo sociálních sítích. V případě velkých dat je však možné analyzovat i taková data z různých zdrojů a s různou strukturou, což může být pro organizace velkým přínosem.

K těmto základním 3 V charakteristikám Big Data někteří autoři později přidali další V, tzv. 4 V / 5 V, a tím začal závod v hledání dalších V. Pro hlubší poznání bych si dovolil odkázat na práci „Big Data Ecosystem“ [6], kde je možnost se dozvědět o dalších V a mnoho dalších zajímavých informací o Big Data.

3.5 Data Analytics

Ačkoli se práce datových vědců a datových analytiků někdy spojuje, tyto obory nejsou totožné. Termín datový analytik ve skutečnosti znamená pouze jedno nebo druhé.

Datový vědec přichází do hry dříve než datový analytik, zkoumá obrovský soubor dat, zkoumá jeho potenciál, identifikuje trendy a poznatky a vizualizuje je pro ostatní. Datový analytik vidí data v pozdější fázi. Podává zprávy o tom, co mu říkají, na základě své analýzy vytváří předpisy pro lepší výkonnost a optimalizuje veškeré nástroje související s daty.

Datový analytik pravděpodobně analyzuje konkrétní soubor strukturovaných nebo číselných dat na základě zadané otázky nebo otázek. Datový vědec se spíše zabývá většími masami strukturovaných i nestrukturovaných dat. Bude také formulovat, testovat a hodnotit výkonnost položených otázek v kontextu celkové strategie.

Datový analytik má více co do činění se zasazením historických dat do kontextu a méně s prediktivním modelováním a strojovým učením. Analýza dat není otevřené hledání správné otázky; spočívá v tom, že od začátku máme připraveny správné otázky. Navíc na rozdíl od datových vědců datoví analytici obvykle nevytvářejí statistické modely ani netrénují nástroje strojového učení.

Namísto toho se datoví analytici zaměřují na strategii podniků a porovnávají datová aktiva s různými organizačními hypotézami nebo plány. Datoví analytici také častěji pracují s lokalizovanými daty, která již byla zpracována. Naopak pro zpracování surových dat i jejich analýzu jsou nezbytné technické i netechnické dovednosti v oblasti datové vědy. Obě role samozřejmě vyžadují matematické, analytické a statistické dovednosti.

Datoví analytici mají při své každodenní práci menší potřebu širšího přístupu k obchodní kultuře. Namísto toho mají při analýze dat tendenci zaujmout více odměřený postoj. Jejich rozsah a účel bude téměř jistě omezenější než u datového vědce.

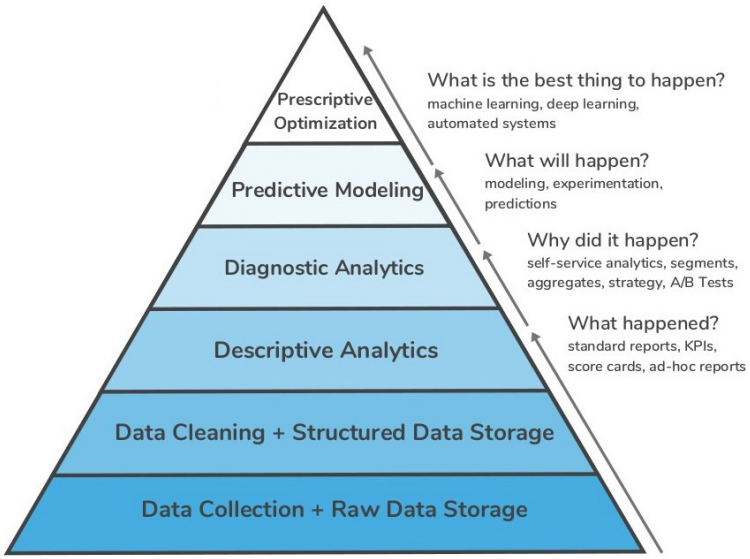
Souhrnně řečeno, datový vědec se při pohledu na data spíše dívá dopředu, předpovídá nebo prognózuje. Vztah mezi datovým analytikem a daty je retrospektivní. Datový analytik se pravděpodobněji zaměří na konkrétní otázky, na které bude hledat odpovědi při hloubání v existujících souborech dat, které již byly zpracovány za účelem získání poznatků.

Datová analytika vs Analýza dat

Častým omylem je, že analýza dat a datová analytika jsou totéž. Obecně přijímané rozlišení je následující kdy, *Datová analytika* je široká oblast využívání dat a nástrojů k přijímání

obchodních rozhodnutí. *Analýza dat* na druhou stranu je podmnožina datové analytiky, se týká konkrétních činností.

Datová analytika je široký pojem, který definuje koncept a praxi (nebo možná vědu a umění) všech činností souvisejících s daty. Hlavním cílem je, aby datoví experti, včetně datových vědců, inženýrů a analytiků, usnadnili zbytku podniku přístup k těmto zjištěním a jejich pochopení. Data, která leží nezpracovaná, tak jak jsou, nemají žádnou hodnotu. Hodnotu naopak přináší to, co s těmito daty děláte. Datová analytika zahrnuje všechny kroky, které podniknete, a to jak s lidskou, tak strojovou pomocí, abychom objevili, interpretovali, vizualizovali a vyprávěli příběh vzorců ve vašich datech s cílem řídit obchodní strategii a výsledky. Úspěch datové analytiky tví v poskytnutí lepší strategii, kam se podnikání organizace může ubírat. Pokud je datová analytika prováděna dobře, může nám pomoci např. v následujících oblastech: najít trendy, odhalit příležitosti, předvídat události anebo pomoci udělat kvalitnější rozhodnutí managementu na různých úrovních řízeni organizace.



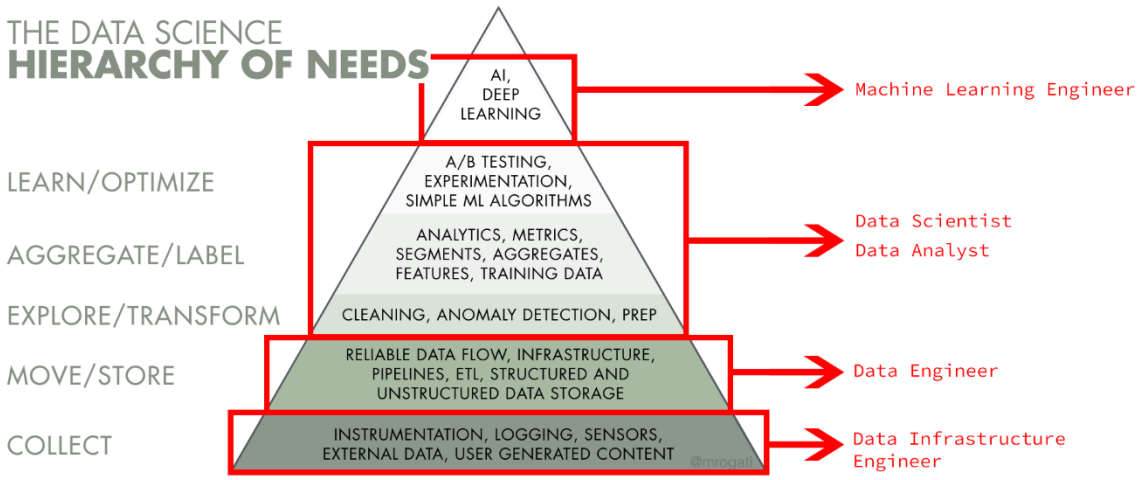
Obrázek 20 - Analytická pyramida potřeb [59]

Analýzu dat považujeme za jeden z kroků datové analytiky. Analýza dat spočívá v čištění, transformaci, modelování a dotazování dat s cílem najít užitečné informace. Analýza dat se obvykle omezuje na jednu, již připravenou sadu dat. Data se prohlédnou, uspořádají a zpochybní ve smyslu hypotézy. Existuje mnoho typů technik analýzy dat. Zde jsou uvedeny ty nejznámější:

- Deskriptivní (statistická) analýza. Tato analýza odpovídá na otázku "Co se stalo?" s využitím historických dat. Statistická analýza zahrnuje sběr, analýzu, interpretaci, prezentaci a modelování dat.
- Diagnostická analýza. Tato analýza odpovídá na otázku "Proč se to stalo?" hledáním příčiny na základě poznatků zjištěných během statistické analýzy. Tento typ analýzy je přínosný pro identifikaci vzorců chování dat.
- Prediktivní analýza. Tato analýza navrhuje, co se pravděpodobně stane, s využitím předchozích dat. Prediktivní analýza na základě dat vytváří předpovědi budoucích výsledků.
- Preskriptivní analýza. Tento typ analýzy kombinuje poznatky ze statistické, diagnostické a prediktivní analýzy s cílem určit opatření, která je třeba přijmout k vyřešení aktuálního problému.

3.6 Data Engineering

Podíváme-li se na hierarchii potřeb na obrázek 21 pro data science projekt, zjistíme, že dalším krokem po shromáždění dat pro analýzu je *Datové inženýrství* (Data Engineering). Tuto disciplínu není radno podceňovat, protože umožňuje efektivní ukládání dat a spolehlivý tok dat a zároveň se stará o infrastrukturu.

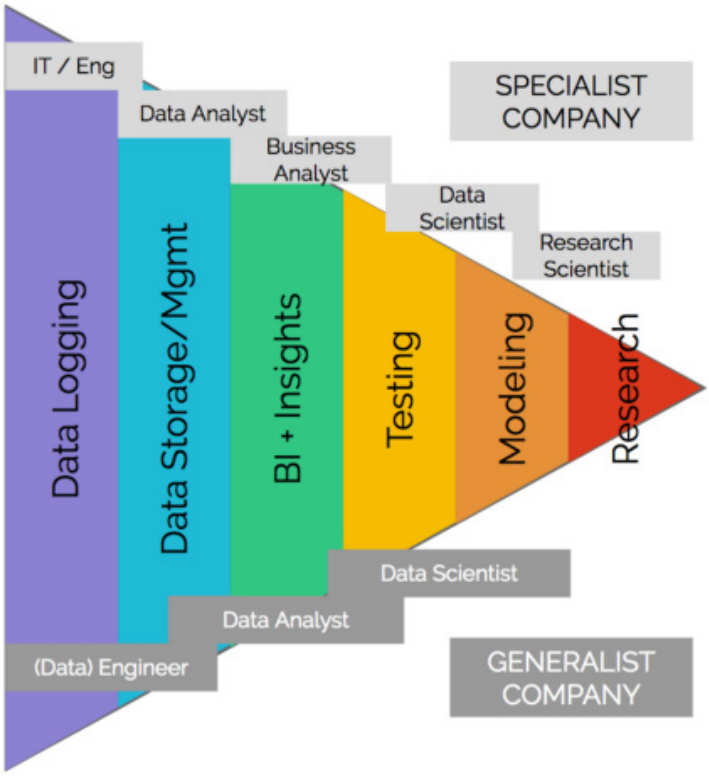


Obrázek 21 - Pyramida potřeb pro Data science [60]

Datové inženýrství je soubor operací zaměřených na vytváření rozhraní a mechanismů pro tok a přístup k informacím. Je zapotřebí specializovaných odborníků – datových inženýrů – kteří udržují data tak, aby zůstala dostupná a použitelná pro ostatní. Stručně řečeno, datoví inženýři nastavují a provozují datovou infrastrukturu organizace a připravují ji pro další analýzu datovými analytiky a vědci. Podíváme-li se na pyramidu potřeb z trochu jiného úhlu než v obrázku 21, můžeme pozorovat lehký překryv pozic v oblasti práce z daty (viz obrázek 22) ale je jasné, že na začátku procesu stojí Datový inženýr a na konci je Datový vědec.

S různými potřebami firem se liší i jejich personální strategie. Některé dávají přednost specialistům (pro každou vrstvu pyramidy využívají různé lidi), zatímco jiné dávají přednost generalistům (lidé, kteří mají na starosti velké části projektu a nespecializují se na konkrétní jednu vrstvu či krok v procesu). Model specialistů dosahuje kvalitnějších výsledků v každé fázi, ale vyžaduje obrovskou komunikační režii pro zajištění správného budování napříč celým týmem. Nízké překrývání odpovědností určuje bezproblémové vlastnictví, ale také ztěžuje dorozumění se stejným jazykem v rámci celé pyramidy, resp. týmu. Zatímco model

generalisty umožňuje lidem vytvářet produkty (tj. dashboardy, modely atd.) rychleji, na druhou stranu často může skončit na chybějících potřebných hlubokých znalostech dané vrstvy. Upřímně řečeno, dosáhnout kvalitního nastavení týmu je v praxi opravdu oříšek, zvlášť když se podíváme na dostupnost daných profesí na pracovním trhu a momentální dostupnosti daných specialistu [61].



Obrázek 22 - Pozice pro práci s daty v organizaci [61]

Abychom pochopili datové inženýrství jednoduše, podívejme se na databáze. V rámci velké organizace obvykle existuje mnoho různých typů softwaru pro řízení provozu: ERP, CRM, výrobní systémy a další. A také existuje i mnoho různých databází. Jak se počet zdrojů dat násobí, rozptýlená data v různých formátech brání organizaci vidět úplný a jasný obraz o stavu jejího podnikání. Je třeba vyřešit, jak získat údaje o prodeji z jeho specializované databáze, například se skladovými záznamy uloženými na serveru SQL. Vzniká tak nutnost integrovat data do jednotného úložného systému, kde jsou data shromážděna, přeformátována a připravena k použití – tzv. datového skladu DWH (*Data Warehouse)* anebo EDW (Enterprise Data Warehouse) . Nyní se mohou datoví vědci a inženýři zabývající se business

intelligence (BI) připojit k datovému skladu, získat přístup k potřebným datům v potřebném formátu a začít z nich získávat cenné poznatky.

Proces přesunu dat z jednoho systému či datového úložiště do druhého zajišťují datoví inženýři. Technologicky nadřízená pozice se nazývá *Datový architekt* a je zodpovědný za budování DW – navrhuje jeho strukturu, definuje zdroje dat, volí jednotný formát dat a zásadním způsobem nastavuje proces získávaní dat neboli datovou pipelinu (Data Integration, Data Acquisition, Data Ingestion) pomocí technik ETL/ELT[14] či jejich moderních derivátů. Existují dva hlavní typy: dávkově řízené (batch) a v reálném čase (real-time, stream). Dávkově řízené zpracování zpracovává data pouze s určitou frekvencí a často je řízen systémem pro orchestraci dat, například Apache Airflow nebo unixový Cron. Obvykle se zpracovává velká dávka historických dat najednou, proto to trvá dlouho než se zpracování dokončí, a způsobují větší zpoždění dat v koncovém systému. Běžně se zpracovávají data ve 24h cyklu za předchozí den, samotné zpracování také může trvat i několik hodin a tak se naprosto běžné, že než se všechna data objeví v datovém skladu, může nastat prodleva i 48h. Opakem k tomuto zpracování je systém zpracování v reálném čase, který zpracovává nová data, jakmile jsou k dispozici, a mezi zdrojovým a koncovým systémem nedochází téměř k žádnému zpoždění. Architektura pro zpracování dat v reálném čase se velmi liší od architektury dávkových pipeline, protože data jsou zpracovávána jako proud událostí, nikoli jako bloky záznamů. Jeden z nejznámějších systému na streamové zpracování v reálném čase je systém Apache Kafka anebo IBM MQ. Zpoždění mezi zdrojovým rozhraním API a cílovým tématem Kafka může být v řádu max vteřin.

Datový sklad je centrální úložiště, kde se surová data transformují a ukládají do podoby vhodné pro dotazování. Bez DW musí datoví vědci čerpat data přímo z produkční databáze

---

[14] Operace ETL:
1. **E**xtrakce dat ze zdrojových databází - Na začátku pipeline máme co do činění se surovými daty z mnoha různých zdrojů. Datoví inženýři píší části kódu (úlohy), které běží podle plánu a extrahují všechna data shromážděná během určitého období.
2. **T**ransformace dat - Data z různých zdrojů jsou často nekonzistentní. Pro efektivní dotazování a analýzu je tedy nutné je upravit. Po extrakci dat inženýři provedou další sadu úloh, které je transformují tak, aby splňovala požadavky na formát (např. měrné jednotky, data, atributy jako barva nebo velikost.) Transformace dat je kritickou funkcí, protože výrazně zlepšuje zjistitelnost a použitelnost dat.
3. **L**oadování přeformátovaných dat do datového skladu - Po uvedení dat do použitelného stavu je mohou datoví inženýři nahrát do cílového umístění, kterým je obvykle relační datový sklad.
4. Udržování změn. Přestože je datový pipelina automatizovaná, datoví inženýři jej musí neustále udržovat: opravují poruchy, aktualizují systém přidáváním/odstraňováním polí nebo přizpůsobují schéma měnícím se potřebám podniku.

a může se stát, že na stejný dotaz budou vykazovat odlišné výsledky nebo dojde ke zpoždění či dokonce k výpadkům. Datový sklad, který slouží podniku jako jediný zdroj pravdy, zjednodušuje organizaci reportování a analýzu, rozhodování a předpovídání metrik. Z technického hlediska je datový sklad relační databáze optimalizovaná pro čtení, agregaci a dotazování velkých objemů dat. Překvapivě DW není běžná databáze především se liší z hlediska struktury dat. Běžná databáze normalizuje data s vyloučením jakýchkoli nadbytečných dat a rozděluje související data do tabulek. To zabírá mnoho výpočetních prostředků, protože jeden dotaz kombinuje data z mnoha tabulek. Naproti tomu DW používá jednoduché dotazy s několika málo tabulkami, které zlepšují výkon a analytiku. Za druhé, zaměřené na každodenní transakce, databáze obvykle neukládají historická data, zatímco pro datové sklady je to jejich hlavní účel, protože shromažďují data z více období. DW zjednodušuje práci datového analytika, protože umožňuje manipulovat se všemi daty z jediného rozhraní a odvozovat analýzy, vizualizace a statistiky.

Big Data Engineering

Když mluvíme o datovém inženýrství, nemůžeme opomenout koncept velkých dat. Velká data, která jsou založena na třech V – objem, rychlost a rozmanitost. Big Data inženýrství spočívá v budování masivních úložišť a vysoce škálovatelných a vůči chybám odolných distribuovaných systémů, které jsou schopny data ukládat a zpracovávat.

Architektura velkých dat se liší od běžného zpracování dat, protože zde hovoříme o tak obrovských objemech rychle se měnících informačních toků, které datový sklad není schopen pojmout. Architekturou, která dokáže takové množství dat zpracovat, je datové jezero.

Datové jezero je rozsáhlý datové úložiště pro ukládání dat v jejich původní, nezpracované podobě. Datové jezero vyniká vysokou agilitou, protože není omezeno pevnou konfigurací datového skladu. Zatímco datové sklady uložené v RDBMS používají koncept Schema-On-Write, pro nestrukturovaná data v datovém jezeře se využívá konceptu Schema-On-Read. Koncept *Schema-On-Read* je protiváhou konceptu *Schema-On-Write*. Schéma databáze se vytváří teprve až při čtení dat. Datové struktury nejsou aplikovány nebo iniciovány před načtením dat do databáze; vytvářejí se až během procesu ETL. To umožňuje ukládat do datového jezera nestrukturovaná data. Hlavním důvodem pro vytvoření tohoto nového konceptu je prudký nárůst objemů nestrukturovaných dat a vysoké režijní náklady spojené s původním konceptem využívaným u datových skladů.

Zdroje: [62][63][64][65]

3.7 Vzdělání a uplatnění

3.7.1 Vzdělání

Akademická scéna v České republice

Snad všechny vysoké školy v České republice technického směru se zaměřením na IT nabízí jednotlivé předměty, kterými pokrývá drtivou většinu oblasti Data science. Většina těchto předmětů je součástí programů jako Softwarové inženýrství, Informatika apod. V následujícím seznamu uvedu programy jednotlivých univerzit, kde název programu souvisí s datovou vědou:

VŠE Praha

- *Data Analytics (Bc.)*
- *Data a analytika pro business (Mgr.)*
- *Data & Analytics for Business Management (MBA)* [66]

Univerzita Karlova Praha

- *Softwarové a datové inženýrství / Software and Data Engineering (EN) (Mgr.)*
- *Master in Finance and Data Analytics (EN) (Mgr.)*

Unicorn University Praha

- *Softwarové inženýrství a big data / Software Engineering and Big Data (EN)* (Mgr.)
- *Aplikovaná ekonomie a analýza dat / Applied Economics and Data Analysis (EN)* (Mgr.)

ČVUT Praha

- *Znalostní inženýrství (Mgr.)*

Vysoká škola chemicko-technologická v Praze

- *Datové inženýrství v chemii / Data Engineering in Chemistry (EN) (Mgr.)*

Česká zemědělská univerzita v Praze

- *Environmental Data Science* (EN) (Bc.)

Univerzita Hradec Králové

- *Datová věda (Mgr.)*
- *Informační a znalostní management (Dr.)*

Univerzita Palackého Olomouc

- *Aplikovaná matematika – specializace Data Science* (Bc.)

Masarykova univerzita Brno

- *Analytika byznysových dat* (Bc.)
- *Umělá inteligence a zpracování dat* (Mgr.)

Technická univerzita Ostrava

- *Informační a znalostní management* (Mgr.)

MOOC[15]

V současné době existuje nepřeberné množství společností poskytující online kurzy v oblasti Data science. Oproti on-site kurzům jsou v podstatě za minimální kurzovné, nabízí možnost konzultací s lektorem po chatu nebo emailu, na konci kurzu si účastník kurzu vyzkouší své znalosti na Capstone projektu. Na závěr většina poskytovatelů těchto kurzů nabízí certifikát ať už o absolvování kurzu anebo závěrečných zkoušek. Některé platformy nabízí kurzy za předem stanovenou cenu za daný kurz, jiné nabízí členství na měsíční bázi. Je třeba zmínit, že některé společnosti nabízí kurzy úplně zdarma, pouze není možné využít konzultaci s lektorem a závěrečný certifikát též není součástí.

Coursera.com

Založena jedním z nejznámějším odborníkem na strojové učení a umělou inteligenci, profesorem ze Stanfordu – Adrew Ng.

[15] Masivní otevřený online kurz (MOOC) je online kurz zaměřený na neomezenou účast a otevřený přístup přes web. Kromě tradičních studijních materiálů, jako jsou natočené přednášky, četba a soubory problémů, mnoho MOOC poskytuje interaktivní kurzy s uživatelskými fóry nebo diskusemi na sociálních sítích, které podporují komunitní interakce mezi studenty, profesory a asistenty pedagoga, stejně jako okamžitou zpětnou vazbu k rychlým kvízům a úkolům. MOOC byl poprvé představen v roce 2008 a který nabyl popularity v letech 2010-2012.

- Data Science (Johns Hopkins University)
  - skládající se z 10 samostaných kurzů
    - The Data Scientist's Toolbox (18h)
    - R Programming (57h)
    - Getting and Cleaning Data (20h)
    - Exploratory Data Analysis (55h)
    - Reproducible Research (8h)
    - Statistical Inference (54h)
    - Regression Models (54h)
    - Practical Machine Learning (9h)
    - Developing Data Products (10h)
    - Data Science Capstone (6h)
- Data Science Professional Certificate (IBM)
  - 11 měsíční kurz v rozsahu cca 150h pokrývající oblast Data science a strojového učení
- Introduction to Data Science Specialization (IBM)
  - zkrácená verze předešlého kurzu v rozsahu 4 měsíců a 52h
    - What is Data Science? (9h)
    - Tools for Data Science (17h)
    - Data Science Methodology (8h)
    - Databases and SQL for Data Science with Python (18h)
- Data Engineering Professional Certificate (IBM)
  - 9 měsíční kurz v rozsahu cca 110h
- Data Engineering Foundations Specialization (IBM)
  - zkrácená verze předešlého kurzu v rozsahu 5 měsíců a 45h
    - Introduction to Data Engineering (10h)
    - Python for Data Science, AI & Development (17h)
    - Python Project for Data Engineering (5h)
    - Introduction to Relational Databases (13h)

Udemy.com

- Machine Learning A-Z™: Hands-On Python & R In Data Science - 44h videí, 73 článků, 38 zdrojů: Data preprocessing, Regression, Classification, Clustering, Association rule,

Reinforcement Learning, NLP, Deep Learning, Dimensionality Reduction, Model Selection & Boosting

- The Data Science Course 2021: Complete Data Science Bootcamp - 28.5h videí, 90 článků, 501 zdrojů: kompletní Data Science: Mathematics, Statistics, Python, Advanced Statistics in Python, Machine & Deep Learning
- Python for Data Science and Machine Learning Bootcamp - 25h videí, 13 článků, 5 zdrojů: NumPy, Pandas, Seaborn, Matplotlib, Plotly, Scikit-Learn, Machine Learning, Tensorflow

SimplyLearn.com

- Post Graduate Program in Data Science (Purdue University & IBM) – 12 měs/4200€
- Post Graduate Program in Data Analytics (Purdue University & IBM) – 8 měs/3000€
- Post Graduate Program in Data Engineering (Purdue University & IBM) – 8 měs/2800€
- Data Scientist (IBM)) – 12 měs/1499€

Mezi další MOOC platformy, které jsou mimo jiné též podporovány různými prestižními univerzitami patří:

- edx.org - za kterou stojí klub top univerzit jako Harvard University, Massachusetts Institute of Technology a University of California, Berkeley
- udacity.com
- kaggle.com/learn
- futurelearn.com (univerzity z VB)
- swayam.gov.in (univerzity z Indie)

Dále je možno studovat online přímo na jednotlivých amerických univerzitách:

- Stanford University
- UC Berkeley
- UCLA
- University of Michigan [dostupné na Coursera]
- Johns Hopkins
- University of Illinois – Urbana Champagne [dostupné na Coursera]

- Northwestern
- Imperial College London [dostupné na Coursera]
- CUNY
- University of Wisconsin
- Indiana University
- UC Riverside
- University of Colorado Boulder [dostupné na Coursera]
- Colorado State University

Vzdělání zaměřené na ženy

Tak jak se většina společností snaží přitáhnout více žen do IT, tak vznikají různé ziskové i neziskové iniciativy, které tomu mají napomoci. Jedna z nich je česká czechitas.cz. **Czechitas** se snaží ženy motivovat, vzdělávat a pomáhat jim najít vysněnou práci v oboru. Propojuje je s partnerskými firmami, které vítají různorodost v pracovních týmech. V oblasti datových věd nabízí kurzy:

- Úvod do datové analýzy (7h) - seznámení se základními pojmy ze světa dat a SQL: Úvod do datové analýzy a databázových systémů, zpracování a čištění dat pomocí SQL, vizualizace dat
- Power BI / Tableua (7h) – dva kurzy zaměřené na BI prostředí: Základy práce s BI prostředím (přihlášení, napojení zdrojových dat, práce s daty v nástroji, prezentace výstupů na webu); vytvoření dashboardů s vizualizacemi
- SQL 1,2,3 (7h) – série 3 navazujících kurzů: Úvod do jazyka SQL, seznámení s online prostředím, SQL dotazy, DML, omezení, procedury, funkce, transakce
- SQL (18h) - prostředí Microsoft SQL Server a Azure Data Studio, výběr dat z tabulek databáze, filtrace výsledků, agregace data a upravovat je s využitím funkcí, import dat do databáze, vytváření nových tabulek, spojování a mazaní tabulek
- Python pro data 1,2 (7h) – série dvou navazujících kurzů: Knihovny Pandas a Matplotlib, čištění a filtrování dat, spojování, agregace a výpočty, export do HTML, vytváření grafů

Kromě jednodenních kurzů Czechitas nabízí i semestrální kurz v délce 150 h nazvaný *Digitální akademie*, kde se zájemci naučí pracovat s databázemi, čistit data, vizualizovat je a tvořit analýzy. Programování v jazyce Python je integrální součástí celého kurzu.

V průběhu kurzu je kladen důraz na analytické myšlení i prezentační a komunikační dovednosti. Jednotlivé oblasti a předměty jsou pravidelně aktualizovány podle současných potřeb na trhu práce a prezentovány lektory z mnohaletou praxí, kterými se stávají pro účastníky kurzu jejich mentormi. Cílem kurzu je vypracovat závěrečný projekt nad reálnými daty. Odměnou pro ty nejlepší je nabídka práce ve spolupracujících společnostech.

Obdobnou iniciativou tentokrát na Slovensku je **AjTyVIT**.sk, která pro ženy nabízí semestrální kurz *Women Data Academy*. V krátkém čase 2 měsíců a 16 lekcích, které jsou dostupné na webu pro offline studium účastník postupně získá celkový přehled v oblasti technologií a přístupů v datové analytice, základy jazyka Python potřebné pro datovou analytiku, nejpoužívanější koncepty a operace pro zpracovaní dat pomocí knihovny Pandas, související matematickou a statistickou terminologii, základní analytiku v Excelu, explorační analýzu, SQL, vizualizaci dat a grafy a úvod do strojového učení. Na závěr kurzu probíhá finální certifikace, která ověří znalosti nabyté v kurzu.

3.7.2 Pracovní pozice v oblasti Data Science

Jak už jsme si představili Data science v minulých kapitolách, je nám teď jasné, že nemůže existovat žádný Chuck Norris[16], který by pokryl celou oblast Data science svými znalostmi. Proto je dobré rozlišovat několik různých typů pracovních pozic a jejich náplní v oblasti Data science.

Data Scientist - Datový vědec

Začněme nejobecnější rolí, datovým vědcem. Být datovým vědcem znamená, že se budeme zabývat všemi aspekty projektu. Počínaje obchodní stránkou přes sběr a analýzu dat až po jejich vizualizaci a prezentaci. Datový vědec zná od všeho trochu; každý krok projektu, díky tomu může nabídnout lepší náhled na nejlepší řešení pro konkrétní projekt a odhalit vzorce a trendy. Kromě toho bude mít na starosti výzkum a vývoj nových algoritmů a

[16] Chuck Norris dokáže zamítnou jakoukoliv nultou hypotézu, i tu co ještě nebyla definovaná.
Pravděpodobnost, že na kostce padne 0 je 1.01, pokud hází Chuck Norris.

přístupů. Ve velkých společnostech mohou být často vedoucími týmů, kde mají na starosti řízení lidi se specializovanými dovednostmi. Jejich soubor dovedností jim umožňuje dohlížet na projekt a vést ho od začátku do konce.

Data Analyst - Datový analytik

Druhou nejznámější rolí je datový analytik. Datový vědec a datový analytik se občas ve společnostech překrývají, a ačkoliv pracují jako datoví analytici a provádí datové analýzy, mohou být nazýváni "datovými vědci". Datoví analytici mají na starosti různé úkoly, jako je vizualizace, transformace a manipulace s daty. Někdy jsou také zodpovědní za sledování webové analytiky a analýzu A/B testování. Vzhledem k tomu, že datoví analytici mají na starosti vizualizaci, mají často na starosti přípravu dat pro komunikaci s obchodní stranou projektu tím, že připravují reporty, které efektivně ukazují trendy a poznatky získané z jejich analýzy.

Data Engineer - Datový inženýr

Datoví inženýři jsou zodpovědní za návrh, budování a údržbu datových pipeline. Datoví inženýři mají na starosti dávkové i streamové zpracování shromážděných dat. Zkrátka zajišťují, aby byla data připravena ke zpracování a analýze ať už v datovém skladu, operačním skladu či v případě big data v datových jezerech či sítích.

Data Architect - Datový architekt

Datový architekt má některé společné povinnosti s datovými inženýry. Oba musí zajistit, aby data byla dobře formátovaná a dostupná pro datové vědce a analytiky, a zabezpečit bezchybných chod a výkonnost datových pipeline. Kromě toho datoví architekti navrhují a podílí se na vytváření nových databázových systémů. Tyto databázové systémy musí udržovat, a to jak z hlediska funkčnosti, tak z hlediska správy. Musí tedy sledovat data a rozhodovat o tom, kdo může prohlížet, používat a manipulovat s různými částmi dat, resp. se podílí na nastavování různých politik. Jejich překryv je i do Data Governance a Master Data Managementu.

Data Storyteller - Vypravěč dat

Jedná se pravděpodobně o nejnovější pracovní roli v tomto seznamu, docela významnou a kreativní. Často se vyprávění příběhů o datech zaměňuje s vizualizací dat. Ačkoli mají některé společné rysy, je mezi nimi zřetelný rozdíl. Vyprávění příběhů o datech není jen o vizualizaci dat a vytváření reportů a statistik, ale spíše o nalezení narativu, který data nejlépe

vystihuje, a jeho využití k jejich vyjádření. Vypravěč příběhů o datech musí převzít nějaká data, zjednodušit je, zaměřit se na určitý aspekt, analyzovat jejich chování a využít své poznatky k vytvoření poutavého příběhu, který lidem pomůže data lépe pochopit.

Machine Learning Scientist - Vědec zabývající se strojovým učením

Většinou, když se v pracovní pozici setkáme s pojmem "vědec", znamená to, že tato pracovní pozice vyžaduje provádět výzkum a přicházet s novými algoritmy a poznatky. Vědec zabývající se strojovým učením zkoumá nové přístupy k manipulaci s daty a navrhuje nové algoritmy, které se budou používat. Často jsou součástí oddělení výzkumu a vývoje a jejich práce obvykle vede ke vzniku vědeckých prací. Jejich práce má blíže k akademické sféře, přesto se pohybuje v průmyslovém prostředí. Názvy pracovních pozic, které lze použít pro označení vědců zabývajících se strojovým učením, jsou Research Scientist nebo Research Engineer.

Machine Learning Engineer - Inženýr strojového učení

Inženýři strojového učení jsou dnes velmi žádaní. Musí být velmi dobře obeznámeni s různými algoritmy strojového učení, jako je shlukování, kategorizace a klasifikace, a musí mít přehled o nejnovějších pokrocích ve výzkumu v této oblasti. Aby mohli inženýři strojového učení řádně vykonávat svou práci, musí mít kromě znalostí základů softwarového inženýrství také dobré znalosti statistiky a programování. Kromě navrhování a vytváření systémů strojového učení musí inženýři strojového učení provádět testy – například A/B testy - a sledovat výkon a funkčnost různých systémů.

Business Intelligence Developer - Vývojář Business Intelligence

Vývojáři Business Intelligence – nazývaní také BI vývojáři – mají na starosti navrhování a vývoj strategií, které umožňují podnikovým uživatelům rychle a efektivně vyhledávat informace potřebné k rozhodování. Kromě toho musí také velmi dobře používat nové nástroje BI nebo navrhovat vlastní nástroje, které poskytují analytické a obchodní informace pro lepší pochopení jejich systémů. Práce vývojářů BI je většinou orientována na byznys, proto musí mít alespoň základní znalosti základů obchodních modelů a jejich implementace.

Database Administrator - Správce databáze

Někdy se liší tým, který databázi navrhuje, a tým, který ji používá. V současné době může mnoho společností navrhnout databázový systém na základě konkrétních obchodních požadavků. Správu databáze však provádí společnost, která databázi kupuje nebo o její návrh

žádá. V takových případech si každá společnost najímá osobu, které mají správu databázového systému na starosti. Správce databáze bude mít na starosti monitorování databáze, zajišťování jejího správného fungování a vytváření záloh a obnovení. Má také na starosti udělování různých oprávnění různým zaměstnancům podle jejich pracovních požadavků a úrovně zaměstnání.

Data Manager – Data manažer

Datoví manažeři musí mít mnohem větší povědomí o obchodní stránce věci než datoví vědci. Jsou klíčoví pro dosažení důležitých obchodních cílů a jsou zodpovědní za tok dat, procesy a dost často i za koordinaci lidí.

Business Analyst - Byznys analytik

Obchodní analytik zkoumá a analyzuje obchodní procesy. Byznys analytik poskytuje potřebné technické informace pro podnikání. Informační technologie jsou nejčastějším odvětvím, kde se obchodní analytici vyskytují.

Software Engineer - Softwarový inženýr

Softwaroví inženýři se od datových vědců liší tím, že jejich teritorium se mnohem více soustřeďuje na funkce pro koncové uživatele a také na vývoj aplikací a tvorbu funkcí. Zaměřují se na navrhování a vývoj softwarových systémů. Softwaroví inženýři vytvářejí aplikace, které generují data, jež mohou využívat datoví vědci. Tato pozice vyžaduje silné programátorské dovednosti.

Statistician - Statistik

Statistik se od datového vědce liší tím, že se zaměřuje pouze na statistiku, nikoli na všechny ostatní obory, které jsou součástí datové vědy. Chcete-li se stát statistikem, musíme mít vysokoškolské vzdělání (nebo více než jeden titul) v oboru statistiky nebo matematiky. Jako statistik budeme vytvářet a využívat statistické techniky a teorie pro sběr, analýzu a interpretaci číselných údajů. To je nezbytné pro přijímání rozhodnutí a vytváření politiky v organizaci.

Data Modeller - Datový modelář

Práce datového modeláře je pro datové vědce nezbytná, aby mohli vykonávat svou práci. Navrhují strukturu a propojení tabulek v databázích, které jsou místem pro ukládání dat, která datoví vědci používají. Stejně jako datoví vědci jsou i datoví modeláři nezbytní pro to,

aby podnik získal užitečné informace ze surových dat a tyto informace pak využil pro obchodní rozhodnutí.

Big Data Engineer – Big Data inženýr

Nezbytnou dovedností je znalost různých programovacích a skriptovacích jazyků. Obvykle jsou na vstupní pozici preferováni lidé s 1 až 3 lety zkušeností s prací s databázemi a vývojem softwaru.

Business Intelligence Analyst - Business Intelligence analytik

Nejzákladnějším úkolem business intelligence analytika je hledat vzory a hodnoty v datech společnosti. Od analytiků BI se očekává, že budou umět analyzovat data, pracovat s jazykem SQL a provádět vizualizaci a modelování dat. Stejně jako většina datových rolí i tato práce vyžaduje silné komunikační dovednosti, abychom mohli své výsledky přesvědčivě sdělit ostatním ve firmě.

MLOps Engineer - Inženýr MLOps

Inženýr MLOps zajišťuje komunikaci mezi datovými vědci a provozním nebo produkčním týmem. Má hluboce kolaborativní charakter, jehož cílem je co nejvíce automatizovat a uvádět do produkce projekty data science a případně konzultovat s ostatními členy týmu problémy vzniklé v produkčním prostředí.

Zdroje: [67][68]

Jak je vidět, pole působnosti je opravdu široké každý zájemce o práci na projektech Data science si může vybrat tu oblast, která mu je bližší anebo ke které má vhodnější předchozí zkušenosti či vzdělání.

II. PRAKTICKÁ ČÁST

4 APACHE SPARK

V této části práce je prezentován produkt Apache Spark používaný v oblasti zpracování velkých objemů dat a jeho využití v oblasti Data Science. V úvodu je vysvětleno co Apache Spark je, k čemu slouží a jaké typy úloh nám pomáhá řešit.. Krátký náhled do historie Sparku umožňuje lépší pochopení v historickém kontextu. Představeny jsou základní možnosti instalace anebo využití předinstalovaných řešení v nejrozšířenějších cloudových systémech. Dále autor hovoří o abstrakci jádra Sparku a Spark RDD a následně se už zabývá všemi základními komponentami Sparku. Pro podrobnější pochopení jsou představeny také funkce a případné omezení Sparku. Velmi podstatnou částí této kapitoly jsou ukázky praktického použití Sparku na demonstrativních příkladech.

Definice

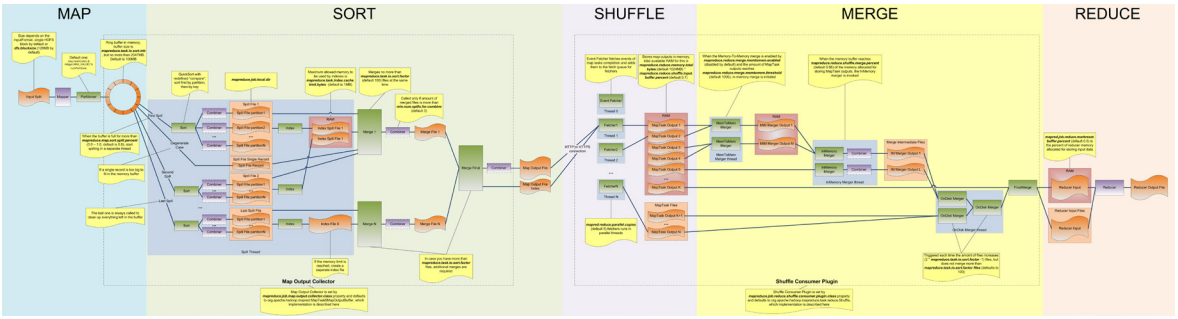
Apache Spark je framework pro zpracování dat, který dokáže rychle provádět úlohy zpracování velmi velkých datových souborů a také dokáže distribuovat úlohy zpracování dat mezi více počítačů v clusteru, a to buď samostatně, nebo ve spojení s dalšími nástroji pro distribuované výpočty. Tyto dvě vlastnosti jsou klíčové pro svět velkých objemů dat a strojového učení, které vyžadují využití obrovského výpočetního výkonu pro zpracování velkých datových úložišť. Spark také přebírá část programátorské zátěže těchto úloh z ramen vývojářů díky snadno použitelnému rozhraní API, které abstrahuje od většiny náročné práce při distribuovaných výpočtech a zpracování velkých objemů dat.

Model MapReduce

Dřív, než se vůbec budeme zabývat Apache Spark, musíme se podívat na základní paradigma pro zpracování velkých objemů dat a tím je MapReduce. MapReduce je framework neboli programovací paradigma[17,18], který umožňuje paralelní zpracování velkých objemů

[17] **Paradigma** (z řeckého παράδειγμα parádeigma = vzor, příklad, model) je obecně přijímané schéma, vzorec myšlení, či model [87].
[18] **Programovací paradigma** je základní programovací styl (na rozdíl od metodiky, která představuje způsob vývoje konkrétních aplikací v softwarovém inženýrství). Paradigmata se liší v pojmech a abstrakcích, které tvoří jednotlivé prvky programu (objekty, funkce, proměnné, omezení, aj.), a krocích, ze kterých se výpočet skládá (přiřazení, vyhodnocení, continuations (pokračování), datové toky, atd.) [88].

dat na komoditním hardwaru. Myšlenkou tohoto modelu je rozdělit komplexní úlohu nebo problém na menší úlohy. Snížení prostorové složitosti, resp. náročnosti zpracování menších úloh je také méně časově náročné. V případě, že se k dokončení takové úlohy použije paralelní zpracování dat, dosáhne uživatel, který úlohu iniciuje, mnohem vyšší rychlosti zpracování úlohy. Paralelismus má velkou výhodu v tom, že šetří čas potřebný k dokončení dané úlohy, ale při návrhu úlohy paralelního zpracování dat je třeba vzít v úvahu i některé problémy. Například nutnost rozdělit složitou úlohu na několik relativně stejně velkých jednodušších úloh. Takové rozdělení však nemusí být vždy jednoznačné. A pak dále, kombinace výsledků z menších úloh může kvůli složitosti vyžadovat další úlohy, speciálně určené k řešení tohoto problému a v konečném důsledku omezující výpočetní kapacitu jednotlivého serveru. Programovací model MapReduce je navržen tak, aby výše uvedené problémy byly při zpracování úloh zohledněny. Algoritmus MapReduce kombinuje dvě fáze: Map a Reduce. V části Map jsou vstupní data rozdělena na několik částí (daných počtem mapovačů nebo souborů), přičemž nad každou z nich je proveden výpočet. Může běžet na libovolném uzlu clusteru. Fáze Map transformuje data na klíč-hodnota a aplikuje logiku napsanou programátorem (např. součet stejných slov). Poté se spustí fáze Reduce, jejímiž vstupními parametry jsou dvojice vrácené fází Map z jednotlivých datových uzlů. Fáze Reduce agreguje výsledky mapování - sloučí všechny hodnoty se stejným klíčem a výsledek vrátí klientovi. Model MapReduce je však pro zjednodušení rozdělen do dvou základních fází, ve skutečnosti se skládá z více fází, které jsou schématicky znázorněny na obrázku 23.



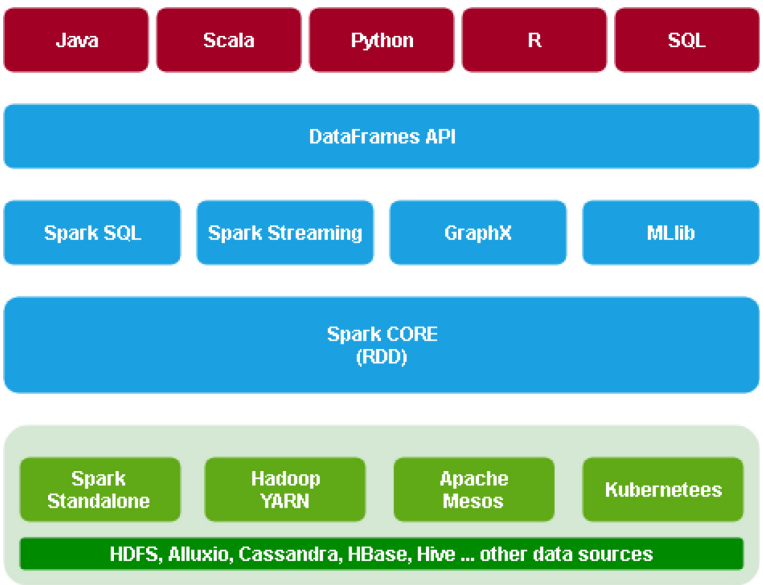
Obrázek 23 - Grafické znázornění MapReduce modelu [69]

Největší výhodou úloh MapReduce je jejich škálovatelnost díky paralelnímu zpracování, kdy je úloha rozdělena mezi více uzlů a každý uzel pracuje s částí úlohy současně. Jakmile je úloha napsána tak, aby běžela nad několika datovými uzly, může stejně tak běžet na tisících datových uzlech. Právě tato výhoda přilákala velké množství vývojářů, kteří využívají vlastnosti, že není nutné měnit kód v případě výrazného nárůstu počtu uzlů v clusteru.

Druhou největší výhodou je lokálnost dat – kdy se data nemusí přenášet do výpočetní jednotky, ale zpracování probíhá přímo v místě uložení dat. Na druhou stranu hlavní nevýhodou frameworku MapReduce je, že mezivýsledky z fáze Map se musí stále fyzicky ukládat do HDFS (bavíme-li se o Apache Hadoop), což má značný negativní dopad na rychlost zpracování úloh MapReduce. To je zásadní rozdíl v rámci Spark, jak bude vysvětleno dále [6].

Hlavní nevýhodou modelu MapReduce je, že mezivýsledky programu se zapisují na disk, což je zásadní problém zejména u aplikací s rychlou odezvou nebo iterativních aplikací, protože tento přístup je výrazně zpomaluje a stává se úzkým hrdlem celého systému. Odpovědí na nedostatky modelu Hadoop MapReduce je Apache Spark. Spark je jednotný analytický engine pro zpracování rozsáhlých dat, ale na rozdíl od MapReduce v Hadoopu dokáže minimalizovat přístup na disk ukládáním mezivýsledků do paměti, takže je vhodný i pro interaktivní, průzkumné analýzy a iterativní programy, které proti MapReduce zrychluje až o jeden či dva řády.

Apache Spark narozdíl od Apache Hadoop neposkytuje žádné úložiště (např. HDFS) ani žádné možnosti správy zdrojů. Je to pouze jednotný framework pro zpracování velkého množství dat téměř v reálném čase. Na obrázku 24 je framework Apache Spark uspořádán do tří hlavních vrstev.



Obrázek 24 - Architektura Apache Spark [6]

1) *Vrstva správce zdrojů* - stejně jako Apache Hadoop, ani Apache Spark nenabízí vlastní modul pro správu zdrojů jako je např. YARN. Pro režim distribuovaného clusteru může být integrován s externími moduly pro správu zdrojů, jako je YARN, Mesos nebo Kubernetess, a může komunikovat s mnoha různými zdroji dat.

2) *Vrstva Spark Core* - Spark Core je základní engine pro rozsáhlé paralelní a distribuované zpracování dat. Dále nabízí další knihovny, které jsou postaveny nad jádrem, umožňují různorodé zpracování od streamování, přes SQL až po strojové učení. Core modul je zodpovědný za správu paměti a obnovu po chybách, plánování, distribuci a monitorování úloh v clusteru & interakci se systémy ukládání dat.

3) *Vrstva ekosystému Spark* - skládá se ze čtyř klíčových zásobníků knihoven

- Spark SQL - modul, který integruje relační zpracování s rozhraním API pro funkční programování. Podporuje dotazování na data buď prostřednictvím jazyka SQL, nebo prostřednictvím jazyka HiveQL.
- Spark Streaming - komponenta, která slouží ke zpracování streamovaných dat v reálném čase.
- GraphX - je knihovna pro manipulaci s grafy[19] a provádění paralelních výpočtů nad grafovými daty.
- MLlib - slouží k provádění běžných funkcí strojového učení v Apache Spark. Modul MLlib poskytuje několik typů algoritmů strojového učení, včetně klasifikace, regrese, shlukování a kolaborativního filtrování nebo podpůrných funkcí, jako je vyhodnocování modelů a import dat.

4) Poslední, avšak nepovažovaná za základní vrstvu, je *API pro pět programovacích jazyků*, jako jsou Java, Scala, Python, R a SQL. Jazyková rozhraní API Sparku umožňují spouštět kód Sparku pomocí různých programovacích jazyků. Většinou Spark v každém jazyce představuje některé základní "koncepty"; tyto koncepty jsou pak přeloženy do kódu Sparku, který běží na clusteru strojů. Pokud používáme pouze strukturované rozhraní API, můžeme očekávat, že všechny jazyky budou mít podobné výkonnostní charakteristiky.

---

[19] V teorii grafů je **graf** abstraktní struktura, která představuje množinu objektů spolu s existujícími spojeními mezi těmito objekty. Matematické abstrakce objektů se nazývají *uzly* (také vrcholy) grafu. Párová spojení mezi uzly se nazývají *hrany*. Hrany mohou být směrované nebo nesměrované. Často se grafy kreslí popisně tak, že se uzly znázorňují pomocí bodů a hrany pomocí čar.

- Scala - Spark je primárně napsán v jazyce Scala, což z něj činí "výchozí" jazyk Sparku.
- Java - Přestože je Spark napsán v jazyce Scala, autoři Sparku dbali na to, abychom mohli kód Sparku psát i v jazyce Java.
- Python - Python podporuje téměř všechny konstrukce, které podporuje Scala.
- R - Spark má dvě běžně používané knihovny R: jednu jako součást jádra Spark (SparkR) a druhou jako komunitní balíček R (sparklyr).
- SQL - Spark podporuje podmnožinu standardu ANSI SQL 2003. Díky tomu mohou analytici i neprogramátoři snadno využívat možnosti Sparku v oblasti velkých dat.

Pro ilustraci faktu, jak moc jsou si jednotlivé API podporovaných jazyků podobné tu mamé příklad, kde si načteme data z JSON do DataFrame, k němu zaregistrujeme dočasnou tabulku, provedeme SQL příkaz se selekcí i projekcí.

```
Template pro všechny jazyky

%jazyk
// importy
// JSON dataset
// vizualizace schématu
// vytvoření dočasné tabulky pomocí DataFrame
// spuštění SQL
```

```
%scala
import spark.implicits._

val path = "/FileStore/tables/friends.json"
val friendsDF = spark.read.json(path)

friendsDF.printSchema()

friendsDF.createOrReplaceTempView("friends")

val teensDF = spark.sql("SELECT name FROM friends WHERE age BETWEEN 13 AND 19")
teensDF.show()
```

```
%java
import org.apache.spark.sql.Dataset;
import org.apache.spark.sql.Row;

Dataset<Row> friendsDF = spark.read().json("/FileStore/tables/friends.json");

friendsDF.printSchema();

friendsDF.createOrReplaceTempView("friends");

Dataset<Row> teen-
sDF = spark.sql("SELECT name FROM friends WHERE age BETWEEN 13 AND 19");
teensDF.show();
```

```
%python

friendsDF = spark.read.json("/FileStore/tables/friends.json")

friendsDF.printSchema()

friendsDF.createOrReplaceTempView("friends")

teensDF = spark.sql("SELECT name FROM friends WHERE age BETWEEN 13 AND 19")
teensDF.show()
```

```
%r
library(SparkR)

friendsDF <- read.df("/FileStore/tables/friends.json", "json")

printSchema(friendsDF)

createOrReplaceTempView(friendsDF, "friends")

teensDF <- sql("SELECT name FROM friends WHERE age BETWEEN 13 AND 19")
head(teensDF)
```

```
%sql
CREATE TEMPORARY VIEW friends
USING org.apache.spark.sql.json
OPTIONS (
  path "/FileStore/tables/friends.json"
);

SELECT name FROM friends WHERE age BETWEEN 13 AND 19
```

4.1 Historie

Apache Spark vznikl na Kalifornské univerzitě v Berkeley v roce 2009 jako výzkumný projekt Spark, který byl následující rok poprvé publikován v článku s názvem "Spark: Cluster Computing with Working Sets" [70], jehož autory byli Matei Zaharia a kolektiv z laboratoře UC Berkeley AMPlab. V té době byl Hadoop MapReduce dominantním paralelním programovacím enginem pro clustery a byl prvním open source systémem, který se zabýval paralelním zpracováním dat na clusterech o tisíci uzlech. Laboratoř AMPlab spolupracovala s několika prvními uživateli MapReduce, aby pochopila výhody a nevýhody tohoto nového programovacího modelu, a mohla proto syntetizovat seznam problémů v několika případech použití a začít navrhovat obecnější výpočetní platformy. Kromě toho Zaharia také spolupracoval s uživateli Hadoopu na Kalifornské univerzitě v Berkeley, aby pochopil jejich potřeby pro tuto platformu – konkrétně šlo o týmy, které prováděly rozsáhlé strojové učení pomocí iterativních algoritmů, které potřebují provést několik průchodů daty. Z diskuzí vzešlo, že clusterové výpočty v sobě skrývaly obrovský potenciál: v každé organizaci, která používala MapReduce, bylo možné vytvořit zcela nové aplikace s využitím stávajících dat a mnoho týmů začalo systém používat po prvním seznámení. Nad druhou stranu engine MapReduce ztěžoval a zároveň činil neefektivním vytváření velkých aplikací. Například typický algoritmus strojového učení mohl potřebovat provést 10 nebo 20 průchodů nad daty a v MapReduce musel být každý průchod napsán jako samostatná úloha MapReduce, kterou bylo nutné spustit zvlášť na clusteru a načíst data z disku vždy od začátku. Aby tento problém vyřešil, tým Sparku nejprve navrhl rozhraní API založené na funkčním programování, které by dokázalo stručně vyjádřit vícekrokové aplikace. Poté tým implementoval toto rozhraní API nad novým enginem, který dokázal provádět efektivní sdílení dat v paměti napříč výpočetními kroky.

První verze Sparku podporovala pouze dávkové aplikace, ale brzy se ukázal další přesvědčivý případ použití: interaktivní datová věda a ad hoc dotazy. Jednoduchým zapojením interpretu Scala do Sparku mohl projekt poskytnout vysoce použitelný interaktivní systém pro spouštění dotazů na stovkách strojů. AMPlab na tuto myšlenku také rychle navázal a vyvinul Shark, engine, který by mohl spouštět dotazy SQL nad Sparkem a umožnit interaktivní využití analytiky i datovými vědci. Shark byl poprvé vydán v roce 2011. Po těchto prvních verzích se rychle ukázalo, že nejvýkonnějšími doplňky Sparku budou nové knihovny, a tak se projekt začal řídit přístupem "standardních knihoven", který má dnes. Konkrétně různé skupiny AMPlab začaly vytvářet MLlib, Spark Streaming a GraphX. Zajistily

také, aby tato rozhraní API byla vysoce interoperabilní, což poprvé umožnilo psát end-to-end big data aplikace ve stejném enginu.

V roce 2013 se projekt rozrostl do široké sféry použití a podílelo se na něm více než 100 přispěvatelů z více než 30 organizací mimo UC Berkeley. AMPlab přesunul Spark pod Apache Software Foundation jako projekt nezávislý na dodavateli. Tým AMPlab na počátku také založil společnost Databricks, která se připojila se ke komunitě dalších společností a organizací přispívajících do Sparku. Od té doby komunita Apache Spark vydala

- v roce 2014 verzi Spark 1.0
- v roce 2016 verzi Spark 2.0
- v roce 2020 verzi Spark 3.0 (akt. verze 3.1.2 z června 2021)

a pokračuje v pravidelných vydáních, která do projektu přinášejí nové funkce.

A konečně, základní myšlenka Sparku spočívající v kompozitních rozhraních API byla v průběhu času také zdokonalována. Rané verze Sparku (před verzí 1.0) do značné míry definovaly toto rozhraní API z hlediska funkčních operací – paralelních operací, jako jsou mapování a redukce nad kolekcemi objektů Java. Počínaje verzí 1.0 projekt přidal Spark SQL, nové rozhraní API pro práci se strukturovanými daty – tabulkami s pevným formátem dat, který není vázán na reprezentaci Javy v paměti. Spark SQL umožnil nové výkonné optimalizace napříč knihovnami a rozhraními API díky detailnějšímu porozumění formátu dat i uživatelskému kódu, který nad nimi běží. Postupem času projekt přidal množství nových rozhraní API, která staví na tomto výkonnějším strukturovaném základu, včetně DataFrames, pipelines pro strojové učení a Structured Streaming, vysokoúrovňového, automaticky optimalizovaného streamovacího rozhraní API.

Spark je na trhu již několik let, ale stále získává na popularitě a roste počet případů použití a úspěšných instalací. Spark bude v dohledné budoucnosti i nadále základním kamenem společností zabývajících se analýzou velkých objemů dat, zejména vzhledem k tomu, že se tento projekt stále rychle rozvíjí což se dá ilustrovat např. na společnosti Databricks (to jsou ti samí lidé, co Spark vyvinuli), která investuje nemalé úsilí do zlepšování celého ecosystému Spark, ve své vlastní komerční distribuci. Každý datový vědec nebo inženýr, který potřebuje řešit problémy s velkými daty, pravděpodobně jako první šáhne po Sparku.

4.2 Prostředí

Apache Spark je možné použít v různých typech prostředí ať už z pohledu:

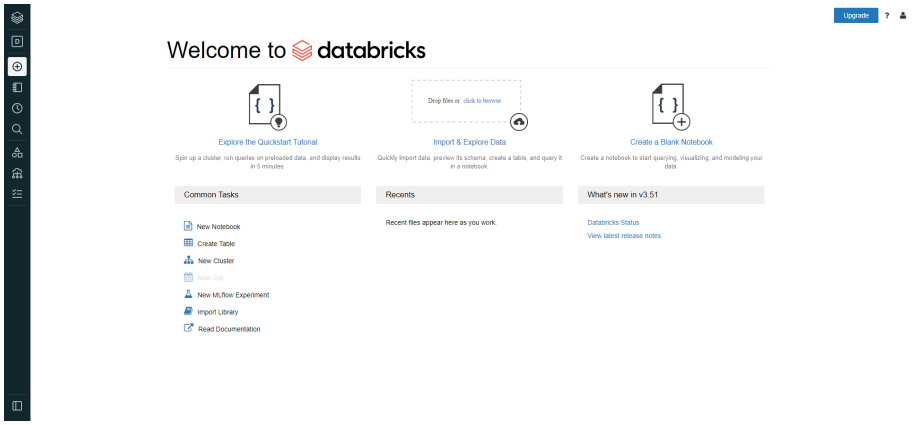
- operačního systému (Linux, Windows)
- standalone / cluster konfigurace
- na samostatném laptopu tak i na stovkách až tisících komoditních serverech (uzlech) jak on-premise, tak ve všech hlavních cloud platformách (AWS, Microsoft Azure, Google Cloud).

My se však nejdřív podíváme na prostředí, které je zdarma a které nám nabízí přímo Databricks[20]. Následně si krátce představíme Colab od Google. Pro samotnou instalaci Sparku doporučím buď oficiální dokumentaci https://spark.apache.org/docs/latest/ anebo step-by-step návod v mé bakalářské práci [6].

Databricks

Prvním krokem je registrace účtu na https://databricks.com/try-platform, po zadání základní iniciálů si můžeme vybrat zda chceme využít nabídku pro společnosti (Databricks Platform) a mít 14 denní zdarma přístup k plné funkcionalitě ecosystému Databricks anebo nabídka pro studenty (Community Edition), která je v podstatě limitována 15GB paměti, základními notebooky pro spolupráci max 3 uživatelů a základní funkcionalitou, která je však pro naše pokusy dostačující.

[20] **Databricks** je společnost, kterou založili tvůrci Apache Spark. Společnost také vytvořila Delta Lake, MLflow a Koalas, open source projekty, které zahrnují datové inženýrství, datovou vědu a strojové učení. Databricks vyvíjí webovou platformu pro práci se Sparkem, která poskytuje automatizovanou správu clusterů a notebooky ve stylu IPythonu. Společnost podílí na organizaci MOOC o Sparku a pořádá konferenci pro komunitu Sparku - Data + AI Summit, dříve známou jako Spark Summit. Krásně napsaný příběh společnosti je dostupný na https://forbes.cz/mld-nahodou-pribeh-akademiku-kteri-nechteli-vydelat-ani-cent/



Obrázek 25 - Úvodní stránka Databricks

Dalším krokem je vytvoření základního clusteru. A to je vše – „ready to use“. Samotná práce probíhá ve stylu pythonovských notebooků, kde si zvolíme defaultní jazyk, ale s tím, že jsme schopni danou buňku přepínačem „%“ změnit na jiný jazyk (např: „`%sql`“ nebo `java`, `scala`, `python`, `r`) a v následující buňce pokračujeme defaultním jazykem anebo si změníme zase úplně jiný jazyk.

Google Colaboratory (Colab)

Google Colaboratory (Colab) je bezplatný produkt společnosti Google, který nám poskytuje přístup k notebooku Jupyter běžícímu v cloudu s možností připojení k výkonným GPU pro urychlení operací Data Science a Machine Learningu. Oficiálně jsou podporovány pouze jazyky Python a Swift

Python

Abychom rozběhali Apache Spark v Google Colab, nejdříve potřebuje nainstalovat všechny zavislosti v prostředí Colab - Apache Spark 3.2.1 s Hadoopem 3.2, Java 11 a Findspark, který nám lokalizuje Spark v systému. Samotná instalace probíhá přímo v noteboooku samotného Colabu.

```
# install java
!apt-get install openjdk-11-jdk-headless -qq > /dev/null

# install spark (change the version number if needed)
!wget -q https://downloads.apache.org/spark/spark-3.1.2/spark-3.1.2-bin-hadoop3.2.tgz

# unzip the spark file to the current folder
!tar xf spark-3.1.2-bin-hadoop3.2.tgz

# install findspark using pip
```

```
!pip install findspark  --quiet

# set your spark folder to your system path environment.
import os
os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-11-openjdk-amd64"
os.environ["SPARK_HOME"] = "/content/spark-3.1.2-bin-hadoop3.2"
os.environ["SPARK_PYTHON"] = "$SPARK_HOME$/python"

# init findspark
import findspark
findspark.init()

# create Spark session
from pyspark.sql import SparkSession
spark = SparkSession.builder.master("local").appName("ColabSpark").con-
fig('spark.ui.port', '4050').getOrCreate()
```

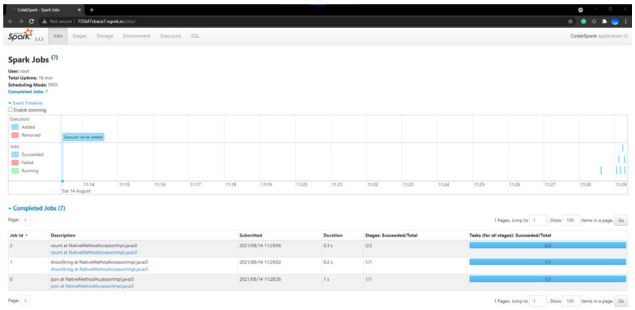
Teď máme teoreticky vše připraveno, ale vzhledem k tomu, že běžíme v cloudu a nejsme schopni napřímo adresovat lokální port pro přístup k Spark UI, pomůžeme si utilitou *ngrok*, což je nástroj, který umožňuje vytvořit zabezpečený tunel z veřejné URL k lokálnímu serveru na našem stroji. Můžeme tak k našemu localhostu přistupovat odkudkoliv.

```
# install ngrok
!wget -q https://bin.equinox.io/c/4VmDzA7iaHb/ngrok-stable-linux-amd64.zip
!unzip -q ngrok-stable-linux-amd64.zip

# create a URL through you can access the Spark UI
get_ipython().system_raw('./ngrok http 4050 &')
get_ipython().system_raw('./ngrok authtoken "1wM00*X*M*h*whAEdUU" &')

# wait for 10s to access the URL
!curl -s http://localhost:4040/api/tunnels
```

Jak vidíme na obrázku 26, tak už máme přístup přímo k Spark UI.



Obrázek 26 - Spark UI v Colab

Scala

Scala není v současné době integrovaným jazykem pro Colab, takže než začneme psát kód, je třeba provést několik kroků k instalaci podpory jazyka Scala do našeho prostředí. Protože se souborový systém Colabu po určité době nečinnosti resetuje, budeme muset tento instalační program spustit znovu, kdykoli se po nějaké době vrátíme k notebooku.

```
# install the Scala Kernel on Google Colab
%%shell
SCALA_VERSION=2.12.8 ALMOND_VERSION=0.3.0+16-548dc10f-SNAPSHOT
curl -Lo coursier https://git.io/coursier-cli
chmod +x coursier
./coursier bootstrap \
    -r jitpack -r sonatype:snapshots \
    -i user -I user:sh.almond:scala-kernel-api_$SCALA_VERSION:$ALMOND_VERSION \
    sh.almond:scala-kernel_$SCALA_VERSION:$ALMOND_VERSION \
    --sources --default=true \
    -o almond-snapshot --embed-files=false
rm coursier
./almond-snapshot --install --global --force
rm almond-snapshot
```

```
%%shell
echo "{
  \"language\" : \"scala\",
  \"display_name\" : \"Scala\",
  \"argv\" : [
    \"bash\",
    \"-c\",
    \"env LD_PRELOAD=/usr/lib/x86_64-linux-gnu/libpython3.6m.so:\$LD_PRELOAD java -
jar /usr/local/share/jupyter/kernels/scala/launcher.jar --connection-file {con-
nection_file}\"
  ]
}" > /usr/local/share/jupyter/kernels/scala/kernel.json
```

A nyní můžeme otestovat, zdali nám Scala na Google Colab poběží.

```
println("Hello, UTB!")
```

```
Hello, UTB!
```

Cloudové služby

Co se týče Colabu, tak jsme si představili spíše prostředí vhodné na vývoj a testování daných technologií či modelů. Pokud však chceme jít s hotovou aplikací do produkce, je třeba se poohlédnout po robustném a osvědčeném řešení. Jednou z variant je použití Databricks, ale to je samozřejmě placený ecosystém, pokud chceme zůstat u zdarma řešení (z

licenčního hlediska samotného Sparku), tak buď se nám nabízí vlastní instalace on-premise, anebo využití cloudových služeb.

AWS

Společnost Amazon nabízí několik možností, jakým způsobem používat Spark ve svém cloudu. První a logickou službou je:

- AWS EMR (Elastic MapReduce) služba – jedná se o cloudová platforma pro zpracování velkého množství dat s využitím open source nástrojů, jako jsou Apache Spark, Apache Hive, Apache HBase, Apache Flink, Apache Hudi a Presto. Amazon EMR usnadňuje nastavení, provoz a škálování prostředí pro zpracování velkých dat díky automatizaci časově náročných úkolů, jako je zajišťování kapacity a ladění clusterů.
- AWS Glue – další alternativou je použití Glu Jobs – což je serveless služba pro integraci dat, která usnadňuje vyhledávání, přípravu a kombinování dat pro analýzu, strojové učení a vývoj aplikací. Toto řešení je spíš vhodné pro použití, kdy Spark nevyužívá HDFS (Hadoop), ale vystačí s Amazoní S3 storagem [71].
- AWS Fargate – je další variantou zejména pro případy, kdy nepotřebujeme celý cluster, ale vystačíme si se single instalací a zpracováváme menší datasety. Toto řešení je zajímavé zejména z pohledu nákladů [71].

Azure

Microsoft na svém cloudovém řešení nabízí službu HDInsight, což je analytická služba s otevřeným zdrojovým kódem v cloudu. Azure HDInsight umožňuje snadné, rychlé a nákladově efektivní zpracování obrovského množství dat. Nabízí použití nejoblíbenějších open-source frameworků, jako jsou Hadoop, Spark, Hive, Kafka, Storm, R a další. Pomocí těchto frameworků je možné pokrýt širokou škálu scénářů, jako je extrakce, transformace a načítání (ETL), datové sklady, strojové učení a internet věcí.

GCP

Google nabízí službu *Dataproc*, což je plně spravovaná a vysoce škálovatelná služba pro provoz Apache Spark, Hadoop, Flink, Presto a více než 30 dalších open source nástrojů a

frameworků, která umožňují využívat open source datové nástroje pro dávkové zpracování, dotazování, streamování a strojové učení.

4.3 Komponenty

Jak jsme si už představili v minulé kapitole, Spark se skládá z několika základních komponent, tak se na ně pojďme podívat trochu blíže.

4.3.1 Spark Core

Spark Core neboli jádro je základem distribuovaného enginu pro zpracování dat ve Sparku. Skládá se ze dvou částí: infrastruktury pro distribuované výpočty a programování RDD abstraktní vrstvy.

Infrastruktura distribuovaných výpočtů je zodpovědná za distribuci, koordinaci a plánování výpočetních úloh na mnoha strojích v clusteru. To umožňuje provádět paralelní zpracování velkého objemu dat efektivně a rychle na velkém clusteru strojů. Dalšími dvěma důležitými povinnostmi infrastruktury distribuovaných výpočtů jsou řešení výpadků výpočetních úloh a efektivní přesun dat mezi stroji, který je znám jako promíchávání dat.

Klíčová programovací abstrakce ve Sparku se nazývá RDD a její znalost je nezbytná pro všechny vývojáře Sparku, zejména její rozhraní API a hlavní koncepty. Technická definice RDD říká, že se jedná o neměnnou a proti chybám odolnou kolekci objektů rozdělených v clusteru, s nimiž lze manipulovat paralelně. V podstatě poskytuje vývojářům sadu API, která jim umožňuje snadno a efektivně provádět rozsáhlé zpracování dat, aniž by se museli starat o to, kde se data v clusteru nacházejí, nebo řešit výpadky uzlů v clusteru. Uveďme příklad, že máme 10 TB soubor dat, který se nachází na různých uzlech distribuovaného úložiště, a potřebujeme zjistit počet řádků obsahujících slovo ‘error’. Můžeme vytvořit instanci RDD, která bude reprezentovat všechny slova v tomto souboru, a Spark je může rozdělit mezi uzly v clusteru tak, aby se logika filtrování a počítání mohla provádět paralelně a urychlit tak logiku vyhledávání a počítání. Rozhraní API RDD je vystaveno v několika programovacích jazycích (Scala, Java a Python) a umožňuje uživatelům používat lokální funkce, které se mají spouštět v clusteru na jednotlivých uzlech. Tzn. že data nedotahujeme na místo zpracování, ale zpracování přesunujeme na místo uložení dat, což je jedna z mnoha výhod distribuovaného zpracování dat, kdy nezatěžujeme síť přesunem ohromných objemů dat a namísto toho přesouváme po síti pouze programové příkazy.

4.3.1.1 RDD, Dataframe, Dataset

RDD

RDD neboli Resilient Distributed Datasets je základní abstraktní datová struktura Sparku. Jedná se o kolekci objektů, která je schopna ukládat data rozdělená mezi více uzlů clusteru a také jim umožňuje paralelní zpracování. Je odolná proti výpadkům, pokud nad RDD provedeme více transformací a pak z nějakého důvodu některý uzel selže, RDD se v takovém případě dokáže automaticky obnovit. Jednoduše, RDD je neměnná distribuovaná kolekce záznamů, která je rozdělena do jednoho nebo více oddílů (partitions), které mohou být distribuovány a počítány v různých uzlech clusteru. RDD mohou obsahovat libovolný typ objektů Pythonu, Javy nebo Scaly nebo uživatelsky definovaných tříd. RDD se dá vytvořit třemi způsoby:

- paralelizací existující kolekce dat
- načtením externí sady dat
- využitím již existující RDD.

```
## 1. paralelizace datové kolekce
muj_list = [1, 2, 3, 4, 5]
muj_list_rdd = sc.parallelize(muj_list)

## 2. načtení externí sady dat
muj_soubor_rdd = sc.textFile("cesta_k_souboru")

## 3. přepoužití existujícího RDD
muj_slovnik_rdd = muj_soubor_rdd.map(s => (s.charAt(0), s))
```

RDD jsou základním stavebním kamenem každé aplikace Spark a zkratka znamená:

- Resilient - odolný vůči chybám a je schopen obnovit data při selhání.
- Distribuovaný - rozděluje data mezi více uzlů v clusteru.
- Dataset - kolekce rozdělených dat s hodnotami.

Má neměnnou povahu (immutable[21]) a řídí se *lazy*[22] transformacemi (data uvnitř RDD jsou dostupná nebo transformovaná až po provedení nějaké akce). Transformace totiž negenerují

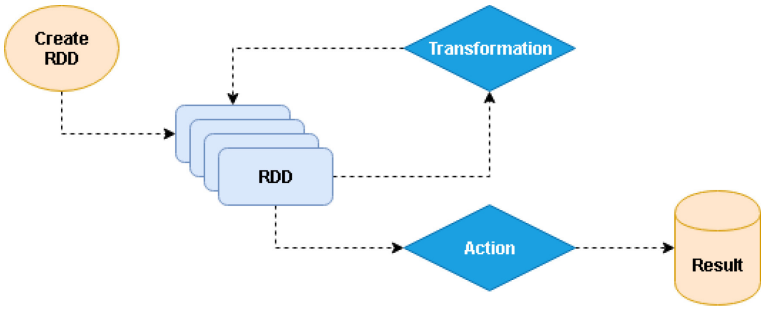
[21] **Immutable** - objekt, jehož stav nelze po vytvoření měnit

[22] **Lazy** znamená, že Spark bude čekat až do poslední chvíle, než provede exekuci výpočetních instrukcí. Ve Sparku se místo okamžité úpravy dat při vyjádření nějakou operaci, vytvoříme plán transformací. Tím, že Spark

žádné data či hodnoty, lze je chápat jako instrukce, jak manipulovat s množinou dat, které Spark při průchodu kódem zapisuje do grafu (Directed Acyclic Graph - DAG). Při volání akce (*action*) na RDD se Spark podívá na tento graf a poté provede transformace, které vedou ke výslednému souboru dat. Zachováním transformací může Spark vhodně optimalizovat přístup k datům, ke kterým se skutečně přistupuje až když jsou potřeba. Poslední důležitá vlastnost - možnost ukládání do mezipaměti - znamená, že všechna data můžeme držet v trvalém "úložišti", jako je paměť (výchozí a nejpreferovanější) nebo disk (nejméně preferovaný kvůli rychlosti přístupu).

RDD nabízejí dva typy operací:

- *transformation* – pomocí transformací definujeme logický transformační plán (např. jednou z běžných transformací je filtrování dat; dalšími operacemi mohou být *map* nebo *flatMap)*
- *action* – pomocí akcí dáváme Sparku pokyn k výpočtu výsledku ze série transformací (příklady operací: *collect, count, first, saveAsTextFile*). Akce mohou být trojího druhu:
  - Akce pro zobrazení dat
  - Akce pro sběr dat do nativních objektů v příslušném jazyce
  - Akce pro zápis do výstupních zdrojů dat



Obrázek 27 - Operace s RDD

čeká s provedením kódu až do poslední chvíle, dochází k optimalizaci celého toku dat. Příkladem může být tzv. predicate pushdown ze světa SQL, kdy se filtr podmínka protlačí až na nejvnořenější selekt a dochází k efektivnímu fitrování.

Dataframes

Poprvé byl představen ve Sparku verze 1.3, aby překonal omezení Spark RDD. Dataframes jsou distribuovanou kolekcí dat, ale zde jsou data uspořádána do pojmenovaných sloupců. Koncepčně odpovídá tabulce v relační databázi nebo datovému rámci v jazyce Python. Dataframy mohou číst data z a zapisovat do různých formátů, jako jsou tabulky CSV, JSON, AVRO, HDFS a HIVE.

Datasets

Dataset, který byl představen ve Sparku verze 1.6, si klade za cíl poskytnout to nejlepší z obou světů: známý objektově orientovaný styl programování a typovou bezpečnost[23] rozhraní RDD API při kompilaci, ale s výkonnostními výhodami optimalizovaného enginu. Datové sady také využívají stejný efektivní mechanismus ukládání mimo haldu jako rozhraní API DataFrame. Pokud jde o serializaci dat, rozhraní Dataset API má koncept kodérů, které jsou velmi pokročilé v tom, že generují bajtový kód pro interakci s daty mimo haldu a poskytují přístup k jednotlivým atributům na vyžádání, aniž by bylo nutné de-serializovat celý objekt. Rozhraní Dataset API je navíc navrženo tak, aby fungovalo stejně dobře v jazycích Java i Scala. Pro další studium rozdílů mezi RDD, Dataframe a Dataset doporučuji článek od Databricks - *A Tale of Three Apache Spark APIs: RDDs vs DataFrames and Datasets* [72].

[23] **Typovou bezpečností** se rozumí, že kompilátor při kompilaci ověří datové typy všech sloupců v datové sadě a v případě neshody datových typů vyhlásí chybu.

4.3.2 Spark SQL

Spark SQL je komponenta postavená nad jádrem Spark Core, která je určena pro strukturované zpracování dat ve velkém měřítku. SQL se stal velmi oblíbeným jazykem pro dotazování a zpracování dat, protože uživatelé v něm mohou poměrně snadno pomocí jednoduchých příkazů vyjádřit svůj záměr a prováděcí stroj pak provede potřebné inteligentní optimalizace[24]. Spark SQL přináší tento jazyk z relačního světa terabajtů do světa zpracování dat na úroveň petabajtů. Uživatelé Sparku mohou zadávat dotazy SQL k provádění zpracování dat nebo využívat vysokoúrovňovou abstrakci vystavěnou prostřednictvím rozhraní API DataFrames[25]. Celá myšlenka je v podstatě inspirována datovými rámci v jazycích R a Python. Další funkcí, kterou Spark SQL poskytuje, je schopnost číst data z různých strukturovaných formátů a systémů ukládání, jako jsou JavaScript Object Notation (JSON), soubory CSV (comma-separated value), soubory Parquet[26] nebo ORC[27], relační databáze, Hive a další, a zapisovat do nich data. Spark SQL lze také použít jako nástroj pro převod dat, který snadno převede data z jednoho formátu do jiného.

4.3.2.1 Příklad – Analýza pomocí SQL

V následujícím příkladu si představíme pouze základní příkazy SQL, selekci, projekci, filtraci, agregaci a relační spojení. Jako testovací data využijeme vzorová data z Oracle distribuce:

- tabulku EMP (zaměstnanci) vyexportujeme do CSV formátu
- tabulku DEPT (oddělení) vyexportujeme do víceřádkového JSON formátu

V *první části* příkladu budeme s těmito daty provádět následující operace:

- načteme CSV data do DataFrame, zkontrolujeme schéma a zobrazíme data

[24] Spark SQL využívá **optimalizátor** Catalyst k provádění optimalizací, které se běžně provádějí v mnoha analytických databázových strojích.
[25] **DataFrame** je distribuovaná kolekce dat uspořádaná do pojmenovaných sloupců, tzn. zde představuje abstraktní objekt koncepčně ekvivalentní tabulce v relační databázi.
[26] Apache **Parquet** je open source formát souborů, který je k dispozici pro všechny projekty v ekosystému Hadoop. Je navržen pro efektivní a výkonný sloupcový formát ukládání dat ve srovnání se soubory založenými na řádcích, jako je CSV.
[27] Apache **ORC** (Optimized Row Columnar) je bezplatný a open-source formát sloupcově orientovaného ukládání dat ekosystému Apache Hadoop.

- provedeme ukázkovou projekci a selekci
- načteme JSON data do DataFrame a opět zkontrolujeme schéma a zobrazíme data
- spojíme tyto dva DataFramy přes cizí klíč *dept_id*
- provedeme ukázkovou
  - filtraci dat (where)
  - agregaci (group by) a znovu následnou
  - filtraci data (having)

```
# načtení zdrojových dat ve formátu CSV
empDF = spark.read.csv('data/emp.csv', inferSchema=True, header=True)
empDF.printSchema()
empDF.describe().show()
empDF.show()
```

```
root
 |-- emp_id: integer (nullable = true)
 |-- ename: string (nullable = true)
 |-- position: string (nullable = true)
 |-- mgr_id: integer (nullable = true)
 |-- hiredate: string (nullable = true)
 |-- salary: integer (nullable = true)
 |-- dept_id: integer (nullable = true)

+-------+------------------+-----+--------+------------------+----------+-----------------+------------------+
|summary|            emp_id|ename|position|            mgr_id|  hiredate|           salary|           dept_id|
+-------+------------------+-----+--------+------------------+----------+-----------------+------------------+
|  count|                14|   14|      14|                13|        14|               14|                14|
|   mean| 7726.571428571428| null|    null| 7739.307692307692|      null| 2073.214285714286|22.142857142857142|
| stddev|178.29436087795676| null|    null|103.71466033898376|      null|1182.5032235162716| 8.017837257372731|
|    min|              7369|ADAMS| ANALYST|              7566|01-05-1981|              800|                10|
|    max|              7934| WARD|SALESMAN|              7902|28-09-1981|             5000|                30|
+-------+------------------+-----+--------+------------------+----------+-----------------+------------------+

+------+-----+---------+------+----------+------+-------+
|emp_id|ename| position|mgr_id|  hiredate|salary|dept_id|
+------+-----+---------+------+----------+------+-------+
|  7839| KING|PRESIDENT|  null|17-11-1981|  5000|     10|
|  7698|BLAKE|  MANAGER|  7839|01-05-1981|  2850|     30|
|  7782|CLARK|  MANAGER|  7839|09-06-1981|  2450|     10|
|  7566|JONES|  MANAGER|  7839|02-04-1981|  2975|     20|
|  7788|SCOTT|  ANALYST|  7566|  13-07-87|  3000|     20|
+------+-----+---------+------+----------+------+-------+
only showing top 5 rows
```

```
# projekce - výběr sloupců
empDF.select('ename','position').show(5)
```

```
+-----+---------+
|ename| position|
+-----+---------+
| KING|PRESIDENT|
|BLAKE|  MANAGER|
|CLARK|  MANAGER|
|JONES|  MANAGER|
|SCOTT|  ANALYST|
+-----+---------+
only showing top 5 rows
```

```
# selekce - výběr řádků
empDF.where("emp_id = 7788 OR salary > 4000 OR ename = 'BLAKE'").show()
```

```
+------+-----+---------+------+----------+------+-------+
|emp_id|ename| position|mgr_id|  hiredate|salary|dept_id|
```

```
+------+-----+---------+------+----------+------+-------+
|  7839| KING|PRESIDENT|  null|17-11-1981|  5000|     10|
|  7698|BLAKE|  MANAGER|  7839|01-05-1981|  2850|     30|
|  7788|SCOTT|  ANALYST|  7566|  13-07-87|  3000|     20|
+------+-----+---------+------+----------+------+-------+
```

```
# načtení zdrojových dat ve formátu multiline JSON
deptDF = spark.read             \
  .option("multiLine", True)    \
  .option("mode", "PERMISSIVE") \
  .json('data/dept.json')
deptDF.printSchema()
deptDF.show()
```

```
root
 |-- dept_id: long (nullable = true)
 |-- dname: string (nullable = true)
 |-- location: string (nullable = true)

+-------+----------+--------+
|dept_id|     dname|location|
+-------+----------+--------+
|     10|ACCOUNTING|NEW YORK|
|     20|  RESEARCH|  DALLAS|
|     30|     SALES| CHICAGO|
|     40|OPERATIONS|  BOSTON|
+-------+----------+--------+
```

```
# join tabulek
empDF.join(deptDF, empDF.dept_id == deptDF.dept_id, "inner").show(5)
```

```
+------+-----+---------+------+----------+------+-------+-------+----------+--------+
|emp_id|ename| position|mgr_id|  hiredate|salary|dept_id|dept_id|     dname|location|
+------+-----+---------+------+----------+------+-------+-------+----------+--------+
|  7839| KING|PRESIDENT|  null|17-11-1981|  5000|     10|     10|ACCOUNTING|NEW YORK|
|  7698|BLAKE|  MANAGER|  7839|01-05-1981|  2850|     30|     30|     SALES| CHICAGO|
|  7782|CLARK|  MANAGER|  7839|09-06-1981|  2450|     10|     10|ACCOUNTING|NEW YORK|
|  7566|JONES|  MANAGER|  7839|02-04-1981|  2975|     20|     20|  RESEARCH|  DALLAS|
|  7788|SCOTT|  ANALYST|  7566|  13-07-87|  3000|     20|     20|  RESEARCH|  DALLAS|
+------+-----+---------+------+----------+------+-------+-------+----------+--------+
only showing top 5 rows
```

```
# agregační funkce + where + having klauzule
from pyspark.sql.functions import sum, avg, col, count

empDF.join(deptDF, empDF.dept_id == deptDF.dept_id, "inner") \
     .where(empDF.dept_id > 10)                              \
     .groupBy("dname")                                       \
     .agg(sum("salary").alias("sum_salary"),                 \
          avg("salary").alias("avg_salary"),                 \
          count("salary").alias("cnt")                       \
         )                                                   \
     .where(col("sum_salary") >= 8000)                       \
     .show()
```

```
+--------+----------+------------------+---+
|   dname|sum_salary|        avg_salary|cnt|
+--------+----------+------------------+---+
|   SALES|      9400|1566.6666666666667|  6|
|RESEARCH|     10875|            2175.0|  5|
+--------+----------+------------------+---+
```

V *druhé části* příkladu se podíváme na to, jak můžeme nad DataFrame zaregistrovat temporární tabulku a přistupovat k datům přímo jazykem SQL. Následně si ukážeme podobné operace jako nad DataFrame, ale přímo za využití SQL, tzn. selekci, projekci, spojení, filtraci, agregaci a filtraci nad agregací.

```
# vytvoření dočasných tabulek z DataFramů
empDF.createOrReplaceTempView("t_emp")
deptDF.createOrReplaceTempView("t_dept")
spark.sql("SELECT * FROM t_emp").show(5)
spark.sql("SELECT * FROM t_dept").show(5)
```

```
+------+-----+---------+------+----------+------+-------+
|emp_id|ename| position|mgr_id|  hiredate|salary|dept_id|
+------+-----+---------+------+----------+------+-------+
|  7839| KING|PRESIDENT|  null|17-11-1981|  5000|     10|
|  7698|BLAKE|  MANAGER|  7839|01-05-1981|  2850|     30|
|  7782|CLARK|  MANAGER|  7839|09-06-1981|  2450|     10|
|  7566|JONES|  MANAGER|  7839|02-04-1981|  2975|     20|
|  7788|SCOTT|  ANALYST|  7566|  13-07-87|  3000|     20|
+------+-----+---------+------+----------+------+-------+
only showing top 5 rows

+-------+----------+--------+
|dept_id|     dname|location|
+-------+----------+--------+
|     10|ACCOUNTING|NEW YORK|
|     20|  RESEARCH|  DALLAS|
|     30|     SALES| CHICAGO|
|     40|OPERATIONS|  BOSTON|
+-------+----------+--------+
```

```
# join tabulek
spark.sql("SELECT ename, dname FROM t_emp JOIN t_dept ON t_emp.dept_id = t_dept.dept_id"
).show(5)
```

```
+-----+----------+
|ename|     dname|
+-----+----------+
| KING|ACCOUNTING|
|BLAKE|     SALES|
|CLARK|ACCOUNTING|
|JONES|  RESEARCH|
|SCOTT|  RESEARCH|
+-----+----------+
only showing top 5 rows
```

```
# join tabulek + filtrace, agregace, filtrace
spark.sql("SELECT dname                           \
               , round(avg(salary)) AS avg_sal    \
            FROM t_emp                            \
            JOIN t_dept                           \
              ON t_emp.dept_id = t_dept.dept_id   \
           WHERE t_emp.dept_id IN (20,30,40)      \
           GROUP BY dname                         \
          HAVING dname != 'ACCOUTING'             \
        ")                                        \
     .show(5)
```

```
+--------+-------+
|   dname|avg_sal|
+--------+-------+
|   SALES| 1567.0|
```

```
|RESEARCH| 2175.0|
+--------+-------+
```

Jak je vidět z předešlých příkladů, Spark SQL nám nabízí pro analýzu dat přístup pomocí podporovaných API jazyků (v našem případě jsme prezentovali využití Pythonu, ale ostatní jazyky (Scala, Java, R) mají velmi podobné API). Druhou možností je využití nativní SQL ve specifikaci ANSI 2003. Pozorní čtenář si mohl všimnout, že API pro přístup k DataFrame jsou občas trochu krkolomné, na druhou stranu nabízí zase plnou podporu svého jazyka, lambda funkce a jiné výhody. Pro konkrétní implementaci daného datového projektu, je potřeba vždy potřeba přihlédnout k daným požadavkům a volit tu kterou variantu. Jak už bylo zmiňováno dříve, Spark je možné využít i na převod dat z různých formátů, což jsme krásně demonstrovali na spojení tabulek, kdy jedna byla uložena v CSV formátu a druhá v JSON formátu.

4.3.3 Spark Streaming

Modul Spark Streaming umožňuje zpracovávat proudová data z různých datových zdrojů v reálném čase s vysokou propustností a odolností proti chybám. Data lze přijímat ze zdrojů, jako je Kafka, Flume, Kinesis, Twitter, HDFS nebo TCP sokety.

Hlavní abstrakce v první generaci zpracování Spark Streaming se nazývá diskrétní proud (*DStream*), který implementuje inkrementální model zpracování datového proudu rozdělením vstupních dat na malé dávky (na základě časové intervalu), které mohou pravidelně kombinovat aktuální stav zpracování a vytvářet nové výsledky. Jinými slovy, jakmile jsou příchozí data rozdělena do malých dávek, každá dávka je zpracována jako RDD a replikuje se na cluster, aby mohly být odpovídajícím způsobem zpracovány.

Od verze Spark 2.1 se používá nový škálovatelný a proti chybám odolný engine pro proudové zpracování nazvaný *Structured Streaming*, který je postaven nad jádrem Spark SQL. Tento engine přistupuje ke streamovaným výpočtům stejným způsobem, jakým Spark přistupuje k dávkovým výpočtům na statických datech a vývojářům umožnuje se odprostit od faktu, že zpracovávají streamové data. Tento nový engine provádí zpracování přírůstkově a nepřetržitě, tak jak přicházejí nová streamovaná data. Novou a důležitou funkcí, kterou Structured Streaming poskytuje, je možnost zpracovávat příchozí proudová data na základě času události.

4.3.3.1 Příklad – Analýza proudu dat

V tomto příkladu se podíváme na zpracování proudu dat. V první části si data zpracujeme interaktivně (batchově), tzn. od začátku budeme mít přístup ke všem záznamům a v druhé části budeme simulovat streaming dat tak, jako by nám přicházela v čase. Pro tento příklad budeme mít jako zdroj 50 JSON souborů, kdy každý obsahuje 2000 záznamů s časovou značkou a hodnotou {Open, Close}.

Batch zpracování

Definujeme si statický DataFrame na souborech a zaregistrujeme k němu název tabulky.

```
from pyspark.sql.types import *

inputPath = "/databricks-datasets/structured-streaming/events/"

# protože známe formát JSONu, tak si tuto strukturu zadefinujeme
```

```
jsonSchema = StructType([ StructField("time", TimestampType(), True), Struct-
Field("action", StringType(), True) ])

# statický dataframe reprezentující data v souborech JSON
staticInputDF = (
  spark
    .read
    .schema(jsonSchema)
    .json(inputPath)
)

display(staticInputDF)
```

time	action
2016-07-28T04:19:28.000+0000	Close
2016-07-28T04:19:28.000+0000	Close
2016-07-28T04:19:29.000+0000	Open
2016-07-28T04:19:31.000+0000	Close
2016-07-28T04:19:31.000+0000	Open

Nyní můžeme vypočítat počet akcí "Open" a "Close" s hodinovým oknem. Za tímto účelem se seskupíme podle sloupce akcí a hodinových oken nad sloupcem času.

```
from pyspark.sql.functions import *

staticCountsDF = (
  staticInputDF
    .groupBy(
       staticInputDF.action,
       window(staticInputDF.time, "1 hour"))
    .count()
)
staticCountsDF.cache()

# DataFrame zaregistrujeme jako tabulku s názvem 'static_counts'
staticCountsDF.createOrReplaceTempView("static_counts")
```

Nyní se můžeme za pomocí jazyka SQL přesvědčit, jak se nám ‘naokénkovaly’ data. Je vidět, že data se agregovaly po jednotlivých hodinách a akcích, tj. první řádek se sloučil z původních 1028 řádků pro akci ‘Close’ v období od 2016-07-26T13:00:00.000+0000 do 2016-07-26T14:00:00.000+0000.

```
%sql
select *
  from static_counts
```

action	window	count
Close	{"start":"2016-07-26T13:00:00.000+0000","end":"2016-07-26T14:00:00.000+0000"}	1028
Open	{"start":"2016-07-26T18:00:00.000+0000","end":"2016-07-26T19:00:00.000+0000"}	1004

```
Close    {"start":"2016-07-27T02:00:00.000+0000","end":"2016-07-27T03:00:00.000+0000"}    971
Open     {"start":"2016-07-27T04:00:00.000+0000","end":"2016-07-27T05:00:00.000+0000"}    995
Open     {"start":"2016-07-27T05:00:00.000+0000","end":"2016-07-27T06:00:00.000+0000"}    986
```

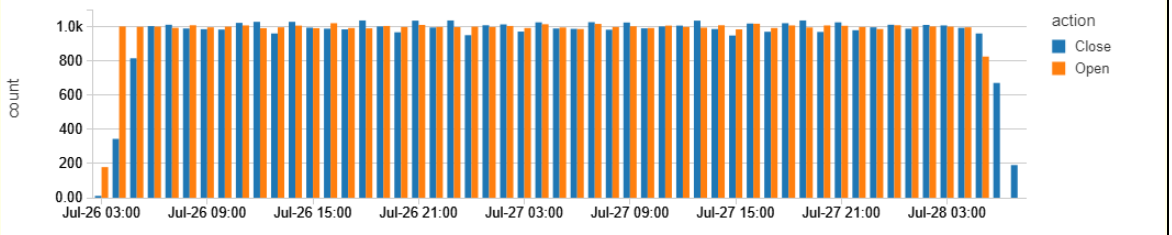
```
%sql
select count(1)
  from static_counts;

select action, sum(count) as total_count
  from static_counts
 group by action
```

Celkem tedy máme 104 nových agregovaných řádků z původních 50000 záznamů pro ‘Open’ a 50000 záznamů pro ‘Close’ akci.

Na závěr této ‘Batch’ sekce se ještě podíváme na časovou osu těchto 104 záznamů (v druhé polovině toho využijeme pro vizuální porovnání, jak nám budou přicházet streamovaná data).

```
%sql
select action
     , date_format(window.end, "MMM-dd HH:mm") as time
     , count
  from static_counts
 order by time, action
```



Stream zpracování

Nyní, když jsme data analyzovali interaktivně, převedeme je na proudový dotaz, který se průběžně aktualizuje podle toho, jak data přicházejí. Protože máme pouze statickou sadu souborů, budeme z nich emulovat proud čtením jednoho souboru po druhém v chronologickém pořadí, v jakém byly vytvořeny. Dotaz, který musíme napsat, je v podstatě stejný jako výše uvedený interaktivní dotaz.

```
from pyspark.sql.functions import *

# podobně jako u staticInputDF, použijeme`readStream` namísto `read`
streamingInputDF = (
  spark
```

```
    .readStream
    .schema(jsonSchema)
    .option("maxFilesPerTrigger", 1)  # pro simulaci streamu bereme pouze 1 file
    .json(inputPath)
)

# stejný dotaz jako staticInputDF
streamingCountsDF = (
  streamingInputDF
    .groupBy(
      streamingInputDF.action,
      window(streamingInputDF.time, "1 hour"))
    .count()
)

# kontrola, zda se jedná o streaming DF
streamingCountsDF.isStreaming
```

```
True
```

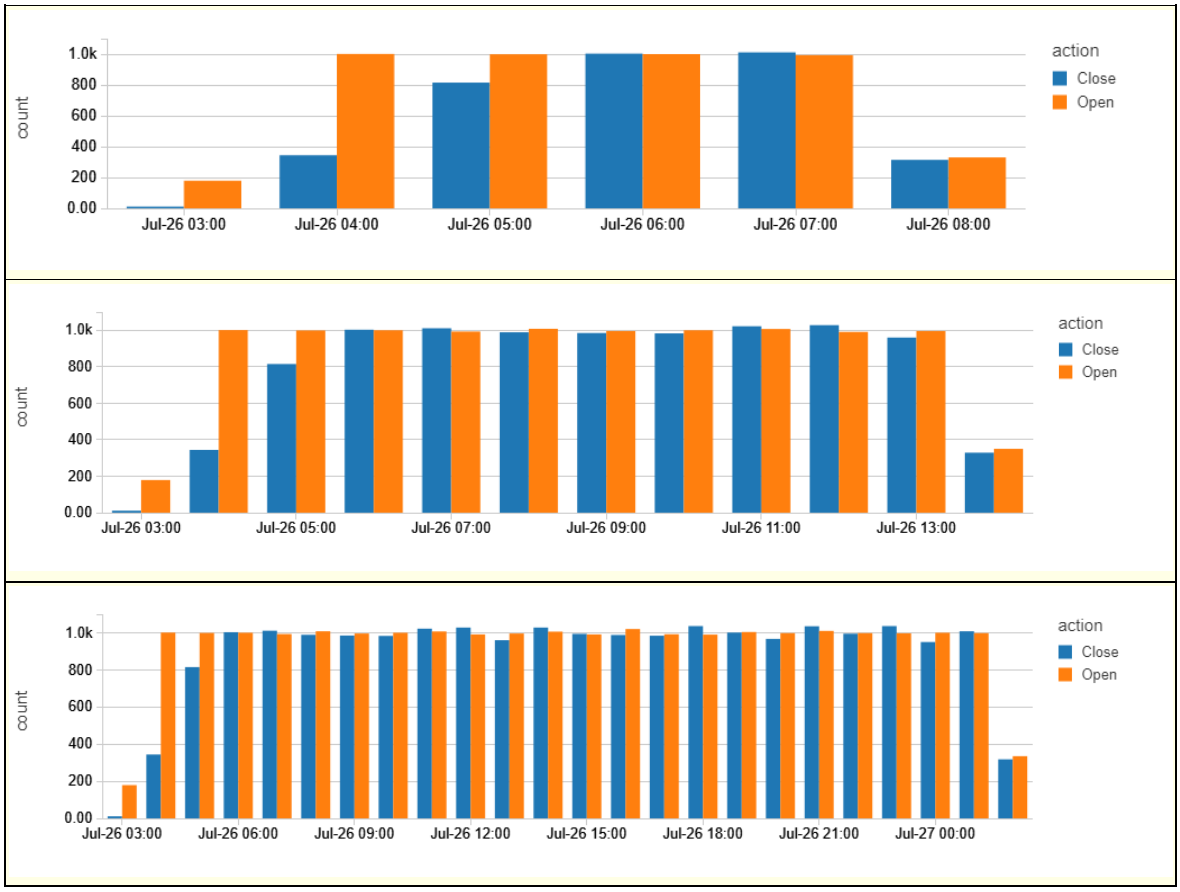
Nyní můžeme spustit samotnou simulaci streamingu definováním *vstupu* a jeho spuštěním. V našem případě se chceme interaktivně dotazovat na počty (stejné dotazy jako výše), takže podle toho nadefinujeme dotaz s využitím dat v tabulce v paměti.

```
spark.conf.set("spark.sql.shuffle.partitions", "2")  # keep the size of shuffles small

query = (
  streamingCountsDF
    .writeStream
    .format("memory")        # memory = uložení tabulky v paměti
    .queryName("counts")     # counts = název tabulky v paměti
    .outputMode("complete")  # complete = všechny výpočty budou v tabulce
    .start()
)
```

query je handle na streamovací dotaz, který běží na pozadí. Tento dotaz průběžně vybírá soubory a aktualizuje počty v definovaném okně. Teď nám už běží samotná simulace streamování, o které se přesvědčíme v následujících dotazech, s malou pauzou mezi dotazy, abychom viděli, jak nám postupně přibývá počet zpracovaných dat.

```
from time import sleep
sleep(5)  # počkáme, ať se nám něco zpracuju
%sql select action, date_format(window.end, "MMM-dd HH:mm") as time, count
       from counts order by time, action
sleep(5)
%sql select action, date_format(window.end, "MMM-dd HH:mm") as time, count
       from counts order by time, action
sleep(5)
%sql select action, date_format(window.end, "MMM-dd HH:mm") as time, count
       from counts order by time, action
```



Na závěr tohoto experimentu je třeba znovu zdůraznit, že zpracování Streamingový dat se z pohledu datového vědce neliší od zpracování formou batch zpracování.

Zdroj: Databricks [73]

4.3.4 Spark GraphX

Zpracování grafů pracuje s datovými strukturami složenými z vrcholů a hran. které je spojují. Datová struktura grafu se často používá k reprezentaci reálných sítí. propojených entit, včetně profesních sociálních sítí na LinkedIn, sítí propojených webových stránek na internetu apod. Spark GraphX je komponenta, která umožňuje paralelní výpočty tím, že poskytuje abstrakci směrovaného multigrafu s vlastnostmi připojenými ke každému vrcholu a hraně. Komponenta GraphX obsahuje kolekci běžných algoritmů pro zpracování grafů, včetně pořadí stránek, propojených komponent, nejkratších cest a dalších. Samotné API pro práci s grafy je velmi dobře dokumentováno a zásadně nemění od své první verze. Časem byla dodána knihovna nové generace pro analýzu grafů na platformě Spark: GraphFrames. GraphFrames rozšiřuje GraphX a poskytuje API DataFrame o podporu pro různé funkce Sparku.

4.3.4.1 Příklad – Analýza nad grafem

Teorie a zpracování grafů

Zpracování grafů je důležitým aspektem analýzy, který se uplatňuje v mnoha případech použití. Teorie a zpracování grafů se v zásadě týkají definování vztahů mezi různými *uzly* a *hranami*. Uzly jsou objekty, zatímco hrany jsou vztahy, které jsou mezi nimi definovány. To se skvěle hodí pro analýzu sociálních sítí anebo algoritmů, jako je PageRank pro webové stránky, které umožňují lépe pochopit vztahy mezi objekty. Některé obchodní případy použití by mohly být zaměřeny na ústřední osoby v sociálních sítích (kdo je nejoblíbenější ve skupině přátel), důležitost článků v bibliografických sítích (které články jsou nejvíce odkazovány) a samozřejmě zmiňované hodnocení webových stránek.

Cílem tohoto příkladu je ukázat, jak lze použít GraphFrames k analýze grafů. Využijeme k tomu data o sdílení jízdních kol v oblasti San Franciska získané ze stránky Kaggle z let 2013-2015 [74]. Analýzu budeme koncipovat tak, že každý *uzel* bude stanice a každá cesta se stane *hranou* spojující dvě stanice. Tím vznikne směrový graf.

```
%python
# definice zdroje dat
file_location_1 = "/FileStore/tables/bike_station.csv"
file_location_2 = "/FileStore/tables/bike_trip.csv"
file_type = "csv"

# CSV config
infer_schema = "true"
first_row_is_header = "true"
```

```
delimiter = ","

# definice struktury budoucich tabulek
df1 = spark.read.format(file_type) \
  .option("inferSchema", infer_schema) \
  .option("header", first_row_is_header) \
  .option("sep", delimiter) \
  .load(file_location_1)

df2 = spark.read.format(file_type) \
  .option("inferSchema", infer_schema) \
  .option("header", first_row_is_header) \
  .option("sep", delimiter) \
  .load(file_location_2)

# vytvoření tabulek ve formátu parquet
df1.write.format("parquet").saveAsTable("bike_station_csv")
df2.write.format("parquet").saveAsTable("bike_trip_csv")
```

SQL příkazem si ověříme, že se tabulky vytvořily v pořádku a obsahují data.

```
%sql
select * from bike_station_csv;
```

id	name	lat	long	dock_count	city	installation_date
2	San Jose Diridon Caltrain Station	37.329732	-121.901782	27	San Jose	08/06/2013
3	San Jose Civic Center	37.330698	-121.888979	15	San Jose	08/05/2013
4	Santa Clara at Almaden	37.333988	-121.894902	11	San Jose	08/06/2013
5	Adobe on Almaden	37.331415	-121.8932	19	San Jose	08/05/2013
6	San Pedro Square	37.336721	-121.894074	15	San Jose	08/07/2013

```
%sql
select * from bike_trip_csv;
```

id	duration	start_date	start_station_name	start_station_id	end_date	end_station_name	end_station_id	bike_id
123717	305	12/12/2013 18:41	San Francisco Caltrain (Townsend at 4th)	70	12/12/2013 18:46	2nd at South Park	64	478
123718	764	12/12/2013 18:41	Townsend at 7th	65	12/12/2013 18:54	Powell Street BART	39	486
123719	451	12/12/2013 18:42	Grant Avenue at Columbus Avenue	73	12/12/2013 18:49	Powell at Post (Union Sq)	71	553
123720	459	12/12/2013 18:42	Grant Avenue at Columbus Avenue	73	12/12/2013 18:50	Powell at Post (Union Sq)	71	503
123721	725	12/12/2013 18:42	Townsend at 7th	65	12/12/2013 18:54	Market at 4th	76	556

Dalším krokem je vytvoření DataFrame z tabulek a její kontrola.

```
# vytvoření dataframe z tabulek
val bikeStations = sqlContext.sql("SELECT * FROM bike_station_csv")
val tripData = sqlContext.sql("SELECT * FROM bike_trip_csv")

# kontrola dataframe
display(bikeStations)
```

```
bikeStations: org.apache.spark.sql.DataFrame = [id: int, name: string ... 5 more fields]
tripData: org.apache.spark.sql.DataFrame = [id: int, duration: int ... 9 more fields]
```

	id	name	lat	long	dock_count
1	2	San Jose Diridon Caltrain Station	37.329732	-121.90178200000001	27
2	3	San Jose Civic Center	37.330698	-121.888979	15
3	4	Santa Clara at Almaden	37.333988	-121.894902	11
4	5	Adobe on Almaden	37.331415	-121.8932	19
5	6	San Pedro Square	37.336721000000004	-121.894074	15

```
# kontrola datové struktury, že máme správné typy přiřazené ke správným sloupcům
bikeStations.printSchema()
```

```
root
 |-- id: integer (nullable = true)
 |-- name: string (nullable = true)
 |-- lat: double (nullable = true)
 |-- long: double (nullable = true)
 |-- dock_count: integer (nullable = true)
 |-- city: string (nullable = true)
 |-- installation_date: string (nullable = true)
```

Vytvoření grafu

Po importu dat je třeba sestavit graf. To se skládá ze dvou částí. Sestavíme strukturu vrcholů (neboli uzlů) a sestavíme strukturu hran. S GraphFrames je tento proces velmi jednoduchý. Jediné, co musíme udělat, je získat jedinečné hodnoty *id* v tabulce vrcholů a přejmenovat počáteční a koncové stanice na *src*, respektive *dst* pro tabulky hran, což jsou požadované konvence pro vrcholy a hrany v GraphFrames.

```
# import potřebných balíčků
import org.apache.spark.sql._
import org.apache.spark.sql.functions._
import org.graphframes._

# definice grafu
val stationVertices = bikeStations
  .distinct()

val tripEdges = tripData
  .withColumnRenamed("start_station_name", "src")
  .withColumnRenamed("end_station_name", "dst")

# sestavení grafu
val stationGraph = GraphFrame(stationVertices, tripEdges)

# uložení dat do mezipaměti kvůli optiomalizaci, protože předpokládáme, že k datům budeme v průběhu analýz často přistupovat
tripEdges.cache()
stationVertices.cache()
```

Analýza nad grafem

V následující snippetech se pokusím demonstrovat některé z analýz, které lze s grafy provádět.

```
# základní statistika
println("Celkový počet stanic: " + stationGraph.vertices.count)
println("Celkový počet výletů v grafu: " + stationGraph.edges.count)
println("Celkový počet výletů ve zdrojových datech: " + tripData.count)
```

```
Celkový počet stanic: 70
Celkový počet výletů v grafu: 669959
Celkový počet výletů ve zdrojových datech: 669959
```

Jednou z otázek, které si můžeme položit, je, jaké jsou nejčastější cíle v souboru dat. To můžeme zjistit provedením operátoru seskupení a sečtením počtů hran. Tím získáme nový graf s tím rozdílem, že každá hrana bude nyní součtem všech sémanticky stejných hran. V následujícím dotazu uvidíte, které cesty jsou nejčastější, a vypíšeme 10 nejčastějších.

```
# analýza nejčastějších cest
val topTrips = stationGraph
  .edges
  .groupBy("src", "dst")
  .count()
  .orderBy(desc("count"))
  .limit(5)

display(topTrips)
```

src	dst	count
San Francisco Caltrain 2 (330 Townsend)	Townsend at 7th	6216
Harry Bridges Plaza (Ferry Building)	Embarcadero at Sansome	6164
Townsend at 7th	San Francisco Caltrain (Townsend at 4th)	5041
2nd at Townsend	Harry Bridges Plaza (Ferry Building)	4839
Harry Bridges Plaza (Ferry Building)	2nd at Townsend	4357

Tato analýza nám potvrzuje, že mezi nejoblíbenější trasy patří spojnice mezi hlavním vlakovým nádražím (Townsend) a přístavem přes záliv (Ferry building). Další analýzou může být zjištění, zda čas potřebný na překonání z místa A do místa B odpovídá samotné jízdě, anebo zákazníci využívají kolo i na jízdu mimo přímý směr. Dále bych si navrhl udělat histogram časů výpůjček, zda odpovídají denním špičkám při dojíždění do práce, a zda kola využívají spíše zaměstnanci anebo turisté, kteří přijedou do SF vlakem. Výsledkem může být návrh využití vhodné reklamy (ve vlaku, na lodi, ..).

Další analýzou je, zda daná stanice se více užívá pro příjezd či odjezd.

```
# analýza stanic pro příjezd
val inDeg = stationGraph.inDegrees
display(inDeg.orderBy(desc("inDegree")).limit(5))
```

id	inDegree
San Francisco Caltrain (Townsend at 4th)	63179
San Francisco Caltrain 2 (330 Townsend)	35117
Harry Bridges Plaza (Ferry Building)	33193
Embarcadero at Sansome	30796
2nd at Townsend	28529

```
# analýza stanic pro odjezd
val inDeg = stationGraph.inDegrees
display(inDeg.orderBy(desc("outDegree")).limit(5))
```

id	outDegree
San Francisco Caltrain (Townsend at 4th)	49092
San Francisco Caltrain 2 (330 Townsend)	33742
Harry Bridges Plaza (Ferry Building)	32934
Embarcadero at Sansome	27713
Temporary Transbay Terminal (Howard at Beale)	26089

Zajímavá otázka, kterou bychom si mohli položit, je, která stanice má nejvyšší poměr vstupních hran, ale nejmenší počet hran výstupních, tzn. která cesty končí, ale málokdy z ní začínají. To jsou stanice, které nám jako provozovateli budou dělat největší problém – budou se nám tam hromadit kola, které si nikdo nechce půjčit, a které nám mohou chybět na jiné stanici a které budeme muset manuálně převážet. Právě stanice, kde máme více odjezdů než příjezdů vyžadují naši pozornost ve smyslu, že je právě budeme muset zásobovat koly z jiných stanic – viz druhá analýza.

```
# poměr příjezdů vs. odjezdů - kola nám přebývají
val degreeRatio = inDeg.join(outDeg, inDeg.col("id") === outDeg.col("id"))
  .drop(outDeg.col("id"))
  .selectExpr("id", "double(inDegree)/double(outDegree) as degreeRatio")

degreeRatio.cache()

display(degreeRatio.orderBy(desc("degreeRatio")).limit(5))
```

id	degreeRatio
Redwood City Medical Center	1.453376206
Redwood City Public Library	1.300469484
San Francisco Caltrain (Townsend at 4th)	1.286951031
Washington at Kearny	1.272367195
MLK Library	1.233038348

```
# poměr odjezdů vs. příjezdů - kola nám chybí
display(degreeRatio.orderBy(asc("degreeRatio")).limit(5))
```

id	outDegree
Grant Avenue at Columbus Avenue	0.56470011
2nd at Folsom	0.605646173
Powell at Post (Union Square)	0.688700384
San Jose City Hall	0.692854158
San Francisco City Hall	0.749724366

Jak je vidět, potenciál, který nám zpracování grafové analýzy poskytuje je neskutečný, a to jsme teprve začali. Zvlášť když si ke stolu sednou lidi z businessu a datový analytik či vědec můžeme čekat zajímavé objevy, které pro velké společnosti mohou přinést mnoho příležitostí, jak dělat business lepší.

Zdroj: Databricks [75]

4.3.5 Spark MLlib

Spark MLlib je kolekce knihoven Sparku pro strojové učení (ML), které lze používat jako API pro implementaci algoritmů ML. Celkovým cílem je umožnit praktické a snadné škálování ML. Kromě toho, že knihovna Spark MLlib poskytuje více než 50 běžných algoritmů strojového učení, poskytuje také abstrakce pro správu a zjednodušení mnoha úloh tvorby modelů strojového učení.

Spark MLlib poskytuje nástroje, jako jsou tyto:

- ML algoritmy – běžné učící algoritmy jako jsou klasifikace, regrese, shlukování a kolaborativní filtrování pro doporučovací systémy
- Featurizace – extrakce features, transformace, redukce dimenzí a selekce
- Perzistence – nahrávání a ukládaní algoritmů, modelů a pipeline
- Pipeliny – nástroje pro konstrukci, vyhodnocování a ladění ml pipeline

4.3.5.1 Příklad - Korelace

Základní operací ve statistice je výpočet korelace mezi dvěma řadami dat. Balíček *spark.ml* poskytuje flexibilitu pro výpočet párové korelace mezi mnoha řadami dat. Momentálně jsou podporovány dvě korelační metody: Pearsonova korelace a Spearmanova korelace. Korelace vypočítá korelační matici pro vstupní datovou sadu vektorů pomocí zadané metody korelace. Pearsonova korelace je číslo mezi -1 a 1, které udává, do jaké míry spolu dvě proměnné lineárně souvisejí. Výstupem bude DataFrame, který obsahuje korelační matici sloupce vektorů.

```
from pyspark.ml.linalg import Vectors
from pyspark.ml.stat import Correlation

data = [(Vectors.sparse(4, [(0, 2), (1, -2.0)]),),
        (Vectors.dense([4.0, 5.0, 0.0, 3.0]),),
        (Vectors.dense([6.0, 7.0, 0.0, 8.0]),),
        (Vectors.sparse(4, [(0, 5.0), (3, -1.0)]),)]
dataFrame = spark.createDataFrame(data, ["features"])

rowPearson = Correlation.corr(dataFrame, "features").head()
print("Pearsonova korelační matice:\n" + str(rowPearson[0]))

rowSpearman = Correlation.corr(dataFrame, "features", "spearman").head()
print("Spearman korelační matice:\n" + str(rowSpearman[0]))
```

```
Pearsonova korelační matice:
DenseMatrix([[1.        , 0.71976336,        nan, 0.60368161],
             [0.71976336, 1.        ,        nan, 0.90265513],
             [       nan,        nan, 1.        ,        nan],
             [0.60368161, 0.90265513,        nan, 1.        ]])
```

```
Spearman korelační matice:
DenseMatrix([[1. , 0.8, nan, 0.4],
             [0.8, 1. , nan, 0.8],
             [nan, nan, 1. , nan],
             [0.4, 0.8, nan, 1. ]])
```

4.3.5.2 Příklad - Testování hypotéz

Testování hypotézy se provádí za účelem zjištění, zda je výsledek statisticky významný. nebo ne. V současné době balíček spark.ml podporuje Pearsonovy chí-kvadrát ($\chi 2$) testy pro nezávislost. ChiSquareTest provede Pearsonův test nezávislosti pro každou funkci vůči labelu. Pro každou feature se dvojice (feature, labelů) převedou na kontingenční matici, pro kterou se vypočítá statistika chí-kvadrát.

```
from pyspark.ml.linalg import Vectors
from pyspark.ml.stat import ChiSquareTest

data = [(0.0, Vectors.dense(0.5, 10.0)),
        (0.0, Vectors.dense(1.5, 20.0)),
        (1.0, Vectors.dense(1.5, 30.0)),
        (0.0, Vectors.dense(3.5, 30.0)),
        (0.0, Vectors.dense(3.5, 40.0)),
        (1.0, Vectors.dense(3.5, 40.0))]
dataFrame = spark.createDataFrame(data, ["label", "features"])

r = ChiSquareTest.test(dataFrame, "features", "label").head()

print("P hodnota: " + str(r.pValues))
print("Stupeň volnosti: " + str(r.degreesOfFreedom))
print("Statistiky: " + str(r.statistics))
```

```
P hodnota: [0.6872892787909721,0.6822703303362126]
Stupeň volnosti: [2, 3]
Statistiky: [0.75,1.5]
```

4.3.5.3 Příklad - Shlukování pomocí k-means

K-means je metoda, jejímž cílem je rozdělit *n* pozorování do *k* shluků, v nichž každé pozorování patří do shluku s nejbližším průměrem (středy shluků nebo centroidy shluků), který slouží jako prototyp shluku. Výsledkem je rozdělení datového prostoru na shluky, kdy k-means minimalizuje rozptyly uvnitř shluku.

```
from pyspark.ml.clustering import KMeans
from pyspark.ml.evaluation import ClusteringEvaluator

# načtení dat
```

```
dataset = spark.read.format("libsvm").load("sample_data/kmeans_data.txt")[28]

# trénování k-means modelu
kmeans = KMeans().setK(2).setSeed(1)
model = kmeans.fit(dataset)
centers = model.clusterCenters()

# prezentace výsledků
print("Středy shluků: ")
for center in centers:
    print(center)
```

```
Středy shluků:
[9.1 9.1 9.1]
[0.1 0.1 0.1]
```

Na velmi krátkém příkladu jsme si ukázali jednu z typických datových analýz k-means. Vzhledem k omezeným prostorovým možnostem není možné jít více do hloubky, proto bych laskavému čtenáři doporučil velmi kvalitně a detailně zpracovaný příklad na adrese [76].

[28] **libsvm** – je textový formát, který se používá pro řídké data, kdy potřebujeme optimalizovat množství uložených dat (format: `label index1:value1 index2:value2 ...`)

ZÁVĚR

Tato diplomová práce se zabývá tématem **Data Science** a je rozdělena na dvě části – teoretickou a praktickou.

V teoretické části je čtenář uveden do problematiky dat, respektive do historie samotného vzniku a zpracování dat s následným objasněním přeměny dat na informace a znalosti.

Následující kapitola teoretické části se dívá na data z pohledu procesního a projektového a pojednává o málo zpracovávaném tématu životním cyklu dat, představuje metodiku pro datové projekty CRISP-DM v komparaci s jinými metodikami pro Data Science a v závěru se věnuje tématu modelu dospělosti či spíše vyspělosti v této oblasti, což je téma které se v závěrečných vysokoškolských pracích nevyskytuje vůbec.

Třetí kapitola je věnována samotné definici pojmu Data Science, představuje základní oblasti této multidisciplinární problematiky, vysvětluje téma Big Data a jak vše do sebe zapadá či se prolíná. Zaměření se na oblast Data Engineeringu dává z hlediska Data Science opodstatnění, nakolik je integrální součastí všech Data Science projektů, přičemž zabírá signifikantní objem práce v celkovém datovém projektu všeobecně odhadovaném na 60-80% celkového času a nákladů projektu. Jsou zde využity autorovi dlouholeté znalosti z oblasti zpracování dat a kdy přináší svůj vlastní kritický pohled na věc. Na závěr teoretické části je představeno několik variant možností studia oboru Data Science, ať už na akademické půdě v ČR, tak zejména v posledních 10 letech velmi úspěšnému formátu MOOC.

Ve druhé praktické části je představena technologie Apache Spark jako univerzální model distribuovaného zpracování ohromného množství dat v clusteru. Krátce je představena historie vzniku této technologie. Dále jsou prezentovány možnosti instalace či běhu systému ať už on-premise či v cloudu různých poskytovatelů cloudových služeb.

Hlavním cílem této kapiloly, kromě představení samotné architektury systému, tkví právě v demonstraci případů použití jednotlivých komponent. Na samostatných datových setech jsou postupně představovány jednotlivé příklady, jak se ta která komponenta dá využít. V rámci jednotlivých příkladů jsou využity různé programovací jazyky, aby byla dokázano, že Spark je multi-programovací engine a v podstatě příliš nezáleží na tom, jaký programovací jazyk datový vědec ovládá.

Ačkoli by se mohlo zdát, že v případě Data Science se jedná o něco nového a průlomového, opak je částečně pravdou. Samotné algoritmy již existují po mnoho let, co je však nového je právě množství a rychlost zpracovávaných dat, což si vyžaduje nové formy a přístupy k řešení této problematiky. A odpovědí na tuto výzvu je právě Apache Spark.

Nutno podotknout, že v této práci bylo pohlíženo na Data Science s ptačí perspektivy a celá práce je záměrně koncipována jako lehký úvod do problematiky. Z tohoto důvodu byly voleny i základní případy užití tak, aby byly čtenáři jednoduchou formou představeny jednotlivé komponenty technologie, namísto jednoho velkého projektu, který by se na Data Science díval z procesního hlediska a očekával by zpracování ‚projektu' od A do Z.

Na jednom místě zde tak čtenář získává veškerý potřebný a ucelený úvod do oblasti Data Science a příbuzných oborů. Přestože je práce určena ke čtení shora dolů, čtenáři různých úrovní mohou volně přejít přímo ke kapitole nebo technologii, která je zajímá.

Pokračování této práce je možné spratřovat v možnosti zpracování celého projektu podle některé z metodik a vybrání si konkrétní kategorie jako klasifikace, shlukování či regrese. Další možností je zaměření se na jednu konkrétní komponentu, anebo porovnání jednotlivých dílčích algoritmů nad veřejným datasetem.

Na závěr nezbývá než přání, aby tato práce zaujmula a přesvědčila její čtenáře pro výběr studia a budoucí kariéry právě v oblasti Data Science.

SEZNAM POUŽITÉ LITERATURY

[1] FITCH, Tecumseh W. The evolution of speech: A comparative review. Trends in Cognitive Sciences. Trends in Cognitive Sciences. 2000, Vol. 4, No. 7. DOI: 10.1016/S1364-6613(00)01494-7.

[2] POWELL, Barry B. Writing : Theory and History of the Technology of Civilizati-on. New York: John Wiley & Sons, 2012. ISBN 978-1-118-25532-2.

[3] MARR, Bernard. A brief history of big data everyone should read. World Economic Forum. [Online] 2015-02-25. [cit 2021-08-01]. Dostupné z: https://www.weforum.org/agenda/2015/02/a-brief-history-of-big-data-everyone-should-read

[4] Tally stick. Wikipedia. [Online]. [cit 2021-08-01]. Dostupné z: https://en.wikipedia.org/wiki/Tally_stick

[5] MAYER-SCHÖNBERGER, Viktor and Kenneth CUKIER. Big Data: A Revolution That Will Transform How We Live, Work, and Think. Boston: Mariner Books, 2014. ISBN 978-0-544-22775-0.

[6] HANZLÍK, Roman. Big Data Ecosystem. Zlín: Univerzita Tomáše Bati ve Zlíně, 2019, 151 s. (239 038 znaků). Dostupné také z: http://hdl.handle.net/10563/44471. Univerzita Tomáše Bati ve Zlíně. Fakulta aplikované informatiky, Ústav informatiky a umělé inteligence. Vedoucí práce Šenkeřík, Roman.

[7] From Data to Wisdom: The Path of the Most Successful Investors. Value Walk. [Online] 2017-04-26. [cit. 2021-08-01]. Dostupné z: https://www.value-walk.com/2017/04/data-wisdom-path-successful-investors/

[8] VISHWAKARMA, Ashish. Difference between Structured, Semi-structured and Unstructured data. geeksforgeeks.org. [Online] [cit. 2021-08-01]. Dostupné z: https://www.geeksforgeeks.org/difference-between-structured-semi-structured-and-unstructured-data

[9] KALYVAS, James R. and Michael R. OVERLY. Big Data: A Business and Legal Guide. Boca Raton, Florida: CRC Press, 2015. ISBN 978-1-4665-9237-7.

[10] What is the DIKW Pyramid? Ontotext. [Online] [cit. 2021-08-01]. Dostupné z: https://www.ontotext.com/knowledgehub/fundamentals/dikw-pyramid/

[11] KUMAR Sujith. What is Data Lifecycle Management?. Stealthbits. [Online] 2020-07-21. [cit 2021-08-01]. Dostupné z: https://stealthbits.com/blog/what-is-data-lifecycle-management

[12] Data Lifecycle Management (DLM). TechTarget. [Online] 2010-08. [cit 2021-08-01]. Dostupné z: https://searchstorage.techtarget.com/definition/data-life-cycle-management

[13] What is Data Lifecycle Management?. Blancco. [Online] 2017-08-08. [cit 2021-08-01]. Dostupné z: https://www.blancco.com/resources/article-what-is-data-lifecycle-management

[14] Lifecycle Management. NSG. [Online] 2018-01-08. [cit 2021-08-01]. Dostupné z: https://www.nordstargroup.com/life-cycle/attachment/lifecycle-management

[15] ROMADHONI, Firmansyah. What is Data Lifecycle Management? and What phases would it pass through?. Medium. [Online] 2020-03-11. [cit 2021-08-01]. Dostupné z: https://medium.com/jagoanhosting/what-is-data-lifecycle-management-and-what-phases-would-it-pass-through-94dbd207ff54

[16] WATTS, Stephen. Data Lifecycle Management (DLM) Explained. BMC. [Online] 2018-06-26. [cit 2021-08-01]. Dostupné z: https://www.bmc.com/blogs/data-lifecycle-management

[17] What is CRISP DM?. Data Science Process Alliance. [Online]. [cit 2021-08-01]. Dostupné z: https://www.datascience-pm.com/crisp-dm-2

[18] CHAPMAN, Pete, et al. CRISP-DM 1.0: Step-by-step data mining guide. [Online]. [cit. 2021-08-01]. Dostupné z: https://www.the-modeling-agency.com/crisp-dm.pdf

[19] BERKA, Petr. Dobývání znalostí z databází. Praha: Academia, 2003. ISBN 80-200-1062-9.

[20] RAUCH, Jan a Milan ŠIMŮNEK. Dobývání znalostí z databází, LISp-Miner a GUHA. Praha: Oeconomica, 2014. ISBN 978-8024520339.

[21] LAROSE, Daniel T. Discovering knowledge in data: An introduction to data mining. Hoboken, N.J.: Wiley-Interscience, 2005. ISBN 978-0471666578

[22] VORHIES, William. CRISP-DM – a Standard Methodology to Ensure a Good Outcome. Data Science Central. [Online] 2016-06-26. [cit 2021-08-01]. Dostupné z: https://www.datasciencecentral.com/profiles/blogs/crisp-dm-a-standard-methodology-to-ensure-a-good-outcome

[23] SALTZ, Jeffrey, Ivan SHAMSHURIN, Colin CONNORS. Comparing Data Science Project Management Methodologies via a Controlled Experiment. Syracuse University. [Online] 2017. [cit 2021-08-01]. Dostupné z: https://scholarspace.manoa.hawaii.edu/bitstream/10125/41273/1/paper0124.pdf

[24] SALTZ, Jeffrey S., Ivan SHAMSHURIN. Big data team process methodologies: A literature review and the identification of key factors for a project's success. IEEE. [Online]. [cit 2021-08-01]. Dostupné z: https://ieeexplore.ieee.org/abstract/document/7840936

[25] SALTZ, Jeffrey, Ivan SHAMSHURIN, Colin CONNORS. Predicting data science sociotechnical execution challenges by categorizing data science projects. Wiley. [Online] 2017-11-22. [cit 2021-08-01]. Dostupné z: https://onlinelibrary.wiley.com/doi/abs/10.1002/asi.23873

[26] LAU, Cher Han. 5 Steps of a Data Science Project Lifecycle. Towards Data Science. [Online]. [cit 2021-08-01]. Dostupné z: https://towardsdatascience.com/5-steps-of-a-data-science-project-lifecycle-26c50372b492

[27] What is TDSP?. Data Science Process Alliance. [Online]. [cit 2021-08-01]. Dostupné z: https://www.datascience-pm.com/tdsp

[28] What is a Data Science Life Cycle?. Data Science Process Alliance. [Online]. [cit 2021-08-01]. Dostupné z: https://www.datascience-pm.com/domino-data-science-life-cycle

[29] STEELE, Mac. Introducing the Data Science Maturity Model. Domino Data Lab. [Online] 2016-11-22. [cit 2021-08-01]. Dostupné z: https://blog.dominodatalab.com/introducing-the-data-science-maturity-model

[30] Introducing Domino Data Lab’s - Data Science Maturity Model. Domino Data Lab's. [Online]. [cit 2021-08-01]. Dostupné z: https://www.bastagroup.nl/wp-content/uploads/2019/05/Data-science-maturity-model-final.pdf

[31] The Bardess Data Science Maturity Curve. Bardess. [Online] 2019-06-16. [cit 2021-08-01]. Dostupné z: https://www.bardess.com/the-bardess-data-science-maturity-curve

[32] HORNICK, Marc. Data Science Maturity Model - Summary Table for Enterprise Assessment (Part 12). Oracle. [Online] 2018-06-28. [cit 2021-08-01]. Dostupné z:

https://blogs.oracle.com/r/data-science-maturity-model-summary-table-for-enterprise-assessment-part-12

[33] HORNICK, Mark. A Data Science Maturity Model for Enterprise Assessment. Oracle. [Online] 2020-06-16. [cit 2021-08-01]. Dostupné z: https://www.oracle.com/a/devo/docs/data-science-maturity-model.pdf

[34] STRAFFA, Mike. Data Science Maturity Models (DSMMs): A Broad Consensus Composite. Arcalea. [Online] 2020-06-25. [cit 2021-08-01]. Dostupné z: https://arcalea.com/blog/data-science-maturity-models

[35] DHAR, V. Data science and prediction. Communications of the ACM. 56 (12): 64–73. DOI:10.1145/2500499.

[36] LEEK, Jeff. The key word in "Data Science" is not Data, it is Science. Simply Statistics.

[37] CHIKIO, Hayashi. What is Data Science? Fundamental Concepts and a Heuristic Example. Springer Japan, 2013. s. 40–51. DOI:10.1007/978-4-431-65950-1_3. ISBN 978-4431702085.

[38] Data Science - A Complete Introduction. Omnisci. [Online]. [cit 2021-08-01]. Dostupné z: https://www.omnisci.com/learn/data-science

[39] CONWAY, Drew. The Data Science Venn Diagram. drewconway.com. [Online] 2010-09-30. [cit 2021-08-01]. Dostupné z: http://drewconway.com/zia/2013/3/26/the-data-science-venn-diagram

[40] Data Science Venn Diagram. The Data Scientist. [Online]. [cit 2021-08-01]. Dostupné z: https://thedatascientist.com/data-science-without-programming/data_science_venn_diagram

[41] TIERNAY, Brendan. Data Science Is Multidisciplinary. Oralytics. [Online] 2012-06-13. [cit. 2021-08-01]. Dostupné z: https://www.oralytics.com/2012/06/data-science-is-multidisciplinary.html

[42] DAVENPORT Thomas H., D. J. PATIL. Data Scientist: The Sexiest Job of the 21st Century. Harward Business Review. [Online] 2012-10. [cit 2021-08-01]. Dostupné z: https://hbr.org/2012/10/data-scientist-the-sexiest-job-of-the-21st-century

[43] PORTER M. Theodore. Karl Pearson - British mathematician. Britanica. [Online]. [cit 2021-08-01]. Dostupné z: https://www.britannica.com/biography/Karl-Pearson#ref205946

[44] HENDL, J. Přehled statistických metod: Analýza a metaanalýza dat. 4. vydání. Praha: Portál, 2012. ISBN 978-80-262-0200-4.

[45] Statistika. Wikipedia. [Online]. [cit 2021-08-01]. Dostupné z: https://cs.wikipedia.org/wiki/Statistika

[46] BLITZ, Shelby. Exploratory and Confirmatory Analysis: What's the Difference?. Sisence. [Online]. [cit 2021-08-01]. Dostupné z: https://www.sisense.com/blog/exploratory-confirmatory-analysis-whats-difference

[47] Mathematics as a Monarch. Analytics Vidhya. [Online]. [cit 2021-08-01]. Dostupné z: https://www.analyticsvidhya.com/blog/2021/04/mathematics-in-data-science

[48] Online Sample Size Calculators - Users Beware. Sigma Magic. [Online]. [cit 2021-08-01]. Dostupné z: https://www.sigmamagic.com/blogs/online-sample-size-calculators

[49] Hustota pravděpodobnosti. Wikipedia. [Online]. [cit 2021-08-01]. Dostupné z: https://cs.wikipedia.org/wiki/Hustota_pravd%C4%9Bpodobnosti

[50] Statistická významnost. Wikipedia. [Online]. [cit 2021-08-01]. Dostupné z: https://cs.wikipedia.org/wiki/Statistick%C3%A1_v%C3%BDznamnost

[51] SIRSAT, Manisha. What is Confusion Matrix and Advanced Classification Metrics?. Blogspot. [Online] 2019-04-29. [cit 2021-08-01]. Dostupné z: https://manisha-sirsat.blogspot.com/2019/04/confusion-matrix.html

[52] Checkland: Soft Systems Methodology. [Online]. [cit 2021-08-01]. Dostupné z: https://web.archive.org/web/20050223131247/http://www.brl.com/pdfs/checklnd.pdf

[53] Chyby typu I a II. Wikipedia. [Online]. [cit 2021-08-01]. Dostupné z: https://cs.wikipedia.org/wiki/Chyby_typu_I_a_II

[54] 8 Best Math Courses for Machine Learning in 2021- Find the Best One!. MLTut. [Online]. [cit 2021-08-01]. Dostupné z: https://www.mltut.com/best-math-courses-for-machine-learning

[55] Lineární algebra. Wikipedia. [Online]. [cit 2021-08-01]. Dostupné z: https://cs.wikipedia.org/wiki/Line%C3%A1rn%C3%AD_algebra

[56] GREEN, Wiliam. Scalars, Vectors and Matrices, oh my!. Medium. [Online] 2018-07-11. [cit 2021-08-01]. Dostupné z: https://medium.com/@dskswu/scalars-vectors-and-matrices-oh-my-9054d2bda1f9

[57] REINSEL, David, John GANTZ and John RYDNING. Data Age 2025. The Digitization of the World. [Online] 2018-11. [cit. 2021-08-01]. Dostupné z: https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf

[58] Big Data: Definitions and Concepts. Big Data : Concepts, Challenges and Soluti-ons. [Online] [cit. 2021-08-01]. Dostupné z: http://bigdata-tech.blogspot.com/p/big-data-definitions-and-concepts.html

[59] Data Science Hierarchy of Needs. [Online] 2018-06-20. [cit 2021-08-01]. Dostupné z: https://www.slideshare.net/DylanGregersen/data-science-hierarchy-of-needs

[60] ROGATI, Monice. The AI Hierarchy of Needs. Rogati. [Online] 2017-06-12. [cit 2021-08-01]. Dostupné z: https://hackernoon.com/the-ai-hierarchy-of-needs-18f111fcc007

[61] WILLIAMS, Hugh. The Pyramid of Data Needs (and why it matters for your career). Medium. [Online] 2018-01-23. [cit 2021-08-01]. Dostupné z: https://medium.com/@hugh_data_science/the-pyramid-of-data-needs-and-why-it-matters-for-your-career-b0f695c13f11

[62] HUANG, Iris. Voicing for Data Engineering, the unsung hero. Towards Data Science. [Online] 2019-01-03. [cit 2021-08-01]. Dostupné z: https://towardsdatascience.com/voicing-for-data-engineering-the-unsung-hero-b91b6ef39dcd

[63] Data Management: Schema-on-Write Vs. Schema-on-Read. upsolver. [Online] 2020-11-25. [cit 2021-08-01]. Dostupné z: https://www.upsolver.com/blog/manage-your-data-schema-on-read-vs-schema-on-write#Schema-on-Write_What_Why_and_How

[64] What is Data Engineering: Explaining the Data Pipeline, Data Warehouse, and Data Engineer Role. Altexsoft. [Online] 2019-06-25. [cit 2021-08-01]. Dostupné z: https://www.altexsoft.com/blog/datascience/what-is-data-engineering-explaining-data-pipeline-data-warehouse-and-data-engineer-role

[65] WAIBEL, Xinran. Introduction to Data Engineering. KDnuggets. [Online] 2020-12. [cit 2021-08-01]. Dostupné z: https://www.kdnuggets.com/2020/12/introduction-data-engineering.html

[66] Jediné datově a analyticky orientované MBA ve střední Evropě. databusiness.cz. [Online]. [cit 2021-08-01]. Dostupné z: https://www.databusiness.cz/dabm

[67] METWALLI, Sara A.. 10 Different Data Science Job Titles and What They Mean. Towards Data Science. [Online] 2020-11-08. [cit 2021-08-01]. Dostupné z: https://towardsdatascience.com/10-different-data-science-job-titles-and-what-they-mean-d385fc3c58ae

[68] MILLER, Kelsey. 11 Data Science Careers Shaping Our Future. Northeastern University. [Online] 2020-06-04. [cit 2021-08-01]. Dostupné z: https://www.northeastern.edu/graduate/blog/data-science-careers-shaping-our-future

[69] Hadoop MapReduce Comprehensive Description. Distributed Systems Architectu-re. [Online] [cit. 2021-08-01]. Dostupné z: https://0x0fff.com/hadoop-mapreduce-comprehensive-description/

[70] ZAHARIA, M. et al. Spark: Cluster Computing with Working Sets. HotCloud. (2010). University of California, Berkeley. [Online] [cit. 2021-08-01]. Dostupné z: https://www.usenix.org/legacy/event/hotcloud10/tech/full_papers/Zaharia.pdf

[71] STRZELECKI, Mariusz. Running Apache Spark on AWS. Medium. [Online] 2019-12-17. [cit 2021-08-01]. Dostupné z: https://medium.com/acast-tech/running-apache-spark-on-aws-81a5f766d3a6

[72] DAMJI, Jules. A Tale of Three Apache Spark APIs: RDDs vs DataFrames and Datasets. Databricks. [Online] 2016-07-14. [cit 2021-08-01]. Dostupné z: https://databricks.com/blog/2016/07/14/a-tale-of-three-apache-spark-apis-rdds-dataframes-and-datasets.html

[73] Structured Streaming demo Python notebook. Databricks. [Online] 2020-11-24. [cit 2021-08-01]. Dostupné z: https://docs.databricks.com/spark/latest/structured-streaming/demo-notebooks.html#structured-streaming-demo-python-notebook

[74] SF Bay Area Bike Share - Anonymized bike trip data from August 2013 to August 2015. Kaggle. [Online]. [cit 2021-08-01]. Dostupné z: https://www.kaggle.com/benhamner/sf-bay-area-bike-share

[75] Graph analysis tutorial with GraphX. Databricks. [Online] 2020-03-17. [cit 2021-08-01]. Dostupné z: https://docs.databricks.com/spark/latest/graph-analysis/graph-analysis-graphx-tutorial.html

[76] Rikard Sandström. Running KMeans clustering on Spark. Data Scientist Blog. [Online]. [cit 2021-08-01]. Dostupné z: https://rsandstroem.github.io/sparkkmeans.html

[77] Nařízení Evropského Parlamentu - GDPR. EU Parlament. [Online] 2016-04-27. [cit 2021-08-01]. Dostupné z: https://eur-lex.europa.eu/eli/reg/2016/679/oj

[78] Sarbanes–Oxley Act. Wikipedia. [Online]. [cit 2021-08-01]. Dostupné z: https://en.wikipedia.org/wiki/Sarbanes%E2%80%93Oxley_Act

[79] Information Lifecycle Management. Wikipedia. [Online]. [cit 2021-08-01]. Dostupné z: https://en.wikipedia.org/wiki/Information_lifecycle_management

[80] Minimum viable product. Wikipedia. [Online]. [cit 2021-08-01]. Dostupné z: https://en.wikipedia.org/wiki/Minimum_viable_product

[81] CI/CD. Wikipedia. [Online]. [cit 2021-08-01]. Dostupné z: https://en.wikipedia.org/wiki/CI/CD

[82] Capability Maturity Model. Wikipedia. [Online]. [cit 2021-08-01]. Dostupné z: https://en.wikipedia.org/wiki/Capability_Maturity_Model

[83] Diferenciální počet. Wikipedia. [Online]. [cit 2021-08-01]. Dostupné z: https://cs.wikipedia.org/wiki/Diferenci%C3%A1ln%C3%AD_po%C4%8Det

[84] Integrální počet. Wikipedia. [Online]. [cit 2021-08-01]. Dostupné z: https://cs.wikipedia.org/wiki/Integr%C3%A1ln%C3%AD_po%C4%8Det

[85] Gradientní sestup. Wikipedia. [Online]. [cit 2021-08-01]. Dostupné z: https://cs.wikipedia.org/wiki/Gradientn%C3%AD_sestup

[86] MLOps. Wikipedia. [Online]. [cit 2021-08-01]. Dostupné z: https://en.wikipedia.org/wiki/MLOps

[87] Paradigma. Wikipedia. [Online]. [cit 2021-08-01]. Dostupné z: https://cs.wikipedia.org/wiki/Paradigma

[88] Programovací paradigma. Wikipedia. [Online]. [cit 2021-08-01]. Dostupné z: https://cs.wikipedia.org/wiki/Programovac%C3%AD_paradigma

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

CDA	Confirmation Data Analysis
CRISP-DM	The CRoss Industry Standard Process for Data Mining
DE	Data Engineeting
DG	Data Governance
DIKW	Data Information Knowledge Wisdom
DLM	Data Lifecycle Management
DM	Data Mining / Data Model / Data Mart
DS	Data Science
DSMM	Data Science Maturity Model
DW	Data Warehouse
EDA	Exploratory Data Analysis
EDW	Enterprise Data Warehouse
ELT	Extract Load Transform
ETL	Extract Transform Load
MDM	Master Data Manangement
ML	Machine Learning
ODS	Operational Data Store
SQL	Structured Query Language

SEZNAM OBRÁZKŮ

SEZNAM TABULEK