Univerzita Pardubice

Fakulta ekonomicko-správní

# Vytvoření aplikace pro analýzu big dat v Pythonu

Diplomová práce

2025 Bc. Jan Holeček

Univerzita Pardubice
Fakulta ekonomicko-správní
Akademický rok: 2024/2025

# ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

| | |
|---|---|
| Jméno a příjmení: | **Bc. Jan Holeček** |
| Osobní číslo: | **E23041** |
| Studijní program: | **N0613A140041 Aplikovaná informatika – Data Science pro business** |
| Téma práce: | **Vytvoření aplikace pro analýzu big dat v Pythonu** |
| Zadávající katedra: | **Ústav systémového inženýrství a informatiky** |

## Zásady pro vypracování

Cílem práce je charakterizovat současné přístupy k analýze big dat, porovnat existující nástroje a metody, provést sběr a předzpracování dat, analyzovat data, navrhnout modely, a nakonec implementovat celé řešení prostřednictvím aplikace v Pythonu.

**Osnova:**

- Vymezení, přehled a porovnání nástrojů a metod.
- Příprava, sběr a předzpracování dat.
- Analýza dat a návrh modelů.
- Implementace aplikace v Pythonu.
- Vyhodnocení výsledků a diskuse.

Rozsah pracovní zprávy: **cca 50 stran**
Rozsah grafických prací:
Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:

CIELEN, Davy; MEYSMAN, Arno D. B.; ALI, Mohamed. *Introducing Data Science: Big data, machine learning, and more, using Python tools*. Shelter Island: Manning, 2016. ISBN 9781633430037.
GANDOMI, Amir; HAIDER, Murtaza. Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management*, 2015, 35.2: 137-144.
HENDL, Jan. *Big data: věda o datech – základy a aplikace*. Praha: Grada Publishing, 2021. Průvodce. ISBN 978-80-271-3031-3.
JIN, Xiaolong, et al. Significance and challenges of big data research. *Big Data Research*, 2015, 2.2: 59-64.
JOHNSON, Jeff S.; FRIEND, Scott B.; LEE, Hannah S. Big data facilitation, utilization, and monetization: Exploring the 3Vs in a new product development process. *Journal of Product Innovation Management*, 2017, 34.5: 640-658.
MCKINNEY, Wes. *Python for data analysis: data wrangling with Pandas, NumPy, and IPython*. Second edition. Sebastopol: O'Reilly, 2018. ISBN 978-1-491-95766-0.

Vedoucí diplomové práce: **RNDr. Ing. Oldřich Horák, Ph.D.**
Ústav systémového inženýrství a informatiky

Datum zadání diplomové práce: **1. září 2024**
Termín odevzdání diplomové práce: **30. dubna 2025**

L.S.

**prof. Ing. Jan Stejskal, Ph.D.** v.r.
děkan

**prof. Ing. Petr Hájek, Ph.D.** v.r.
garant studijního programu

V Pardubicích dne 1. září 2024

## Prohlášení:

Prohlašuji:

Práci s názvem Vytvoření aplikace pro analýzu big dat v Pythonu jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 7/2019 Pravidla pro odevzdávání, zveřejňování a formální úpravu závěrečných prací, ve znění pozdějších dodatků, bude práce zveřejněna prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 30.04.2025

Bc. Jan Holeček

## PODĚKOVÁNÍ

Rád bych poděkoval svému vedoucímu, RNDr. Ing. Oldřichu Horákovi, Ph.D., za jeho ochotu, odborné vedení a cenné rady, které mi poskytl při zpracování této diplomové práce.

Dále bych chtěl poděkovat své rodině za neustálou podporu, trpělivost a povzbuzení během celého studia.

S hlubokou vděčností bych rád poděkoval i svému zesnulému bratrovi, který mě vždy podporoval, povzbuzoval a jehož rady pro mě byly neocenitelné. Jeho podpora mi pomohla překonat mnohé výzvy během mého studia.

## ANOTACE

*Práce se zaměřuje na současné přístupy k analýze big dat a provádí porovnání existujících nástrojů a metod. V rámci práce je realizován sběr a předzpracování dat, následná analýza získaných dat, návrh vhodných modelů a jejich implementace podle metodiky CRISP-DM. Na závěr je navrženo komplexní řešení, které je realizováno prostřednictvím aplikace vyvinuté v jazyce Python.*

## KLÍČOVÁ SLOVA

*Big data, Python, aplikace, modely, grafy, analýzy*

## TITLE

*Developing an application for big data analysis in Python*

## ANNOTATION

*The thesis focuses on current approaches to big data analysis and compares existing tools and methods. It involves data collection and preprocessing, followed by the analysis of the obtained data, the design of appropriate models, and their implementation according to the CRISP-DM methodology. Finally, a comprehensive solution is proposed, which is implemented through an application developed in Python.*

## KEYWORDS

*Big data, Python, application, models, graphs, analyses*

# OBSAH

# SEZNAM OBRÁZKŮ

# SEZNAM TABULEK

# SEZNAM ZKRATEK A ZNAČEK

| | |
|---|---|
| API | Application Programming Interface |
| AWS | Amazon Web Services |
| CLEAR | Citizen Law Enforcement Analysis and Reporting |
| CRISP-DM | Cross-Industry Standard Process for Data Mining |
| CSV | Comma-Separated Values |
| HDFS | Hadoop Distributed File System |
| HTML | Hypertext Markup Language |
| IUCR | Illinois Uniform Crime Reporting |
| JSON | JavaScript Object Notation |
| PDF | Portable Document Format |
| RDD | Resilient Distributed Dataset |
| SQL | Structured Query Language |

# ÚVOD

V dnešní digitální době neustále narůstá množství dat, což přináší nové výzvy spojené s jejich analýzou a interpretací. Tuto problematiku řeší pojem big data, který je chápán jako komplexní soubory informací, které nemohou být efektivně zpracovávány tradičními nástroji a metodami. Tento nárůst je způsobován nejen rostoucím objemem dat, ale také jejich rychlostí, různorodostí a nároky na přesnost a spolehlivost výsledků. Kvůli těmto charakteristikám jsou na platformy, nástroje, ale také na infrastruktury potřebné pro ukládaní a zpracování data, kladeny stále nové požadavky, jak z těchto dat získat potřebné informace. K využití potenciálu big dat jsou proto vyvíjeny pokročilé metody a rámce, stejně jako se objevují nové platformy a nástroje, a jsou budovány škálovatelné, odolné a zabezpečené infrastruktury pro práci s těmito daty. Mezi nejpoužívanější přístupy lze zařadit distribuovaný programovací model MapReduce. Klíčovým nástrojem pro distribuované zpracování dat je pak například platforma Apache Spark.

Nárůst dat se týká všech sektorů a odvětví, ale ne všechna data jsou volně dostupná pro účely zpracování a analýz. To se ale netýká tzv. open dat, která jsou naopak zdarma zveřejňována na datových portálech a lze je využívat podle podmínek daných licencí, pod kterou je daný dataset zveřejněn. Většinou se jedná o data veřejného sektoru, tzn. open government data, ale lze nalézt i open business data nebo open science data. Portály, na kterých jsou tato data k dispozici, nabízí i různé funkcionality a nástroje pro jejich zpracování, ale ty většinou nepodporují big datasety. Proto je nutné daný dataset nebo datasety stáhnout nebo se napojit přes Application Programming Interface (API) a zpracovat a analyzovat je v jiném nástroji nebo nástrojích. Zároveň je nutné, aby daný uživatel nebo přímo organizace, mohly big dataset analyzovat přes uživatelsky přívětivé a použitelné aplikační rozhraní založené na otevřených technologiích.

V této diplomové práci je proto navržena aplikace pro analýzu big dat, konkrétně s využitím big datasetu týkajícího se kriminality. Její hlavní cíle spočívají v návrhu a implementaci efektivního řešení zpracování těchto dat pomocí moderních otevřených technologií. Celý postup řešení cíle práce je metodicky zasazen do kroků a fází metodiky Cross-Industry Standard Process for Data Mining (CRISP-DM).

V první kapitole jsou vymezeny základní pojmy jako data science a big data spolu s jejich charakteristikami, přínosy a riziky. Data science je představena jako interdisciplinární oblast, která umožňuje odhalování skrytých vzorců v masivních datových souborech. Tyto poznatky mohou ovlivňovat strategické rozhodování a optimalizaci procesů. Zároveň jsou diskutovány

vlastnosti big dat, jako je objem (volume), rychlost (velocity) a spolehlivost (veracity), a jakým způsobem tyto vlastnosti ovlivňují jejich zpracování. Zvláštní pozornost je věnována přínosům a rizikům big dat. Dále jsou představeny pojmy Python, Jupyter Notebook a metodika CRISP-DM. Následující část práce se zaměřuje na popis nástrojů, platforem a metod používaných při práci s big daty. Je založena na rešerši související literatury a porovnání vhodných nástrojů a platforem. Důraz je kladen na Apache Spark a PySpark, což je Python API umožňující využívat funkce Apache Sparku přímo v Pythonu.

Hlavní část práce se zabývá vytvořením aplikace pro analýzu big dat. Tento proces zahrnuje výběr nástrojů a platforem, instalaci a konfiguraci softwarového prostředí, včetně Apache Sparku, PySparku a dalších knihoven pro Python, jako jsou Pandas nebo Matplotlib. Následuje popis návrhu a vytvoření aplikace. Klíčová kapitola 4 celý proces vytvoření aplikace specifikuje v kontextu metodiky CRISP-DM. Konkrétně se jedná o fáze: (1) porozumění problematice, (2) porozumění datům – sběr, popis, průzkum a ověření kvality dat, (3) příprava dat – čištění dat a jejich transformování do vhodného formátu pro další zpracování, (4) modelování – analýza dat je prováděna pomocí vhodných metod odpovídajících charakteristikám big datasetu, (5) vyhodnocení výsledků a (6) využití výsledků v praxi.

Vytvořená aplikace je navržena s důrazem na uživatelskou přívětivost a flexibilitu, aby byla využitelná pro různé typy analýz nad big datasety. K jejímu vývoji je využit technologický stack zahrnující Apache Spark, Python, PySpark a Jupyter Notebooks. Díky použití otevřených technologií se aplikace stává transparentní, snadno rozšiřitelnou a přizpůsobitelnou různým potřebám uživatelů. Apache Spark umožňuje distribuované zpracování dat, což je klíčové pro analýzu big dat, zatímco Python a PySpark poskytují bohaté knihovny pro datovou manipulaci a analýzu. Jupyter Notebooks pak slouží jako interaktivní prostředí pro vývoj, testování a dokumentaci kódu, což usnadňuje spolupráci a sdílení výsledků. Aplikace tedy usnadňuje analýzu big dat dostupných na datových portálech a zpřístupňuje informace obsažené v těchto datech širšímu okruhu uživatelů. Otevírá tak cestu k hlubšímu porozumění datům a podporuje využití otevřených technologií.

# 1 VYMEZENÍ ZÁKLADNÍCH POJMŮ

V této kapitole jsou definovány základní pojmy související s cílem práce. Je vysvětleno, jak data science propojuje matematiku, statistiku a informatiku s cílem analyzovat big data. Ta jsou pak popsána z pohledu jejich charakteristik, jako jsou objem, rychlost a různorodost, a jsou zdůrazněny jejich přínosy pro podnikání, například zlepšení rozhodování nebo personalizace služeb. Zároveň jsou uvedena rizika, včetně bezpečnostních hrozeb a etických otázek. Nakonec jsou vysvětleny pojmy Python a Jupyter notebook.

## 1.1 Data science

Data science (datová věda) je interdisciplinární obor, který propojuje statistiku, matematiku, informatiku a specializované znalosti různých oborů s cílem analyzovat data a získávat tak užitečné informace (Cielen et al., 2016). V současném digitálním světě, kde různé organizace i lidé generují a ukládají velké množství dat, je data science klíčová pro další rozvoj. Umožňuje organizacím lépe chápat dynamiku trhu, optimalizovat vnitřní procesy, zlepšovat rozhodovací mechanismy a identifikovat nové příležitosti pro rozvoj. Jak uvádí van der Aalst (2014), data science je klíčovým faktorem digitální transformace a organizacím, které ji využívají, nabízí strategickou výhodu v neustále se měnícím prostředí. Data science je často chápaná jako sada kroků, které je nutné aplikovat na data, aby z nich byla získána požadovaná informace (Cielen et al., 2016).

Jedním z největších přínosů využívání metod data science je schopnost zlepšit zákaznickou zkušenost prostřednictvím personalizace. Díky analýze dat o chování a preferencích zákazníků mohou firmy lépe přizpůsobit své produkty a služby konkrétním potřebám jednotlivců. To vede k vyšší spokojenosti zákazníků a zvyšuje jejich loajalitu. Výzkumy ukazují, že firmy, které aplikují pokročilé analytické metody, získávají více zákazníků a výrazně zlepšují zákaznickou zkušenost (Wassouf et al., 2020). Další důležitou oblastí, kde data science hraje klíčovou roli, je optimalizace procesů a zefektivnění provozu. Strojové učení a prediktivní analýzy umožňují firmám přesněji plánovat výrobní kapacity, předvídat poptávku a optimalizovat dodavatelské řetězce. To je zvláště důležité v odvětvích, kde jsou logistické náklady významné a efektivita provozu je klíčovým faktorem úspěchu. Firmy jako Tesla využívají data science k optimalizaci výrobních procesů a dodavatelských řetězců, což jim umožňuje udržovat konkurenceschopnost a flexibilně reagovat na změny trhu (Rožanec et al., 2021).

V moderním podnikatelském prostředí je také zásadní schopnost činit rozhodnutí na základě dat, tzv. data-driven rozhodování. Data science poskytuje nástroje pro prediktivní analýzu, které umožňují manažerům předvídat budoucí trendy, hodnotit rizika a identifikovat nové obchodní příležitosti. Výzkumy ukazují, že firmy, které implementují datovou analytiku do svých strategických rozhodovacích procesů, mají vyšší míru úspěchu v inovacích a lepší finanční výsledky než firmy, které spoléhají pouze na tradiční postupy (Provost a Fawcett, 2013; Sarker, 2021). Jednou z klíčových aplikací data science je také predikce a prevence podvodných aktivit, zejména ve finančním sektoru. Banky a finanční instituce používají algoritmy strojového učení k analýze transakčních dat, což jim umožňuje detekovat podezřelé vzorce a minimalizovat finanční ztráty způsobené podvody. Tyto technologie nejen zvyšují bezpečnost finančních operací, ale také posilují důvěru zákazníků v poskytované služby (Ali et al., 2022). Kromě optimalizace provozu a zlepšování bezpečnosti má data science významný dopad na inovace a vývoj nových produktů. Firmy využívají data science a analýzu velkých objemů dat k identifikaci nových tržních trendů, což jim umožňuje vyvíjet produkty, které lépe odpovídají potřebám zákazníků (Johnson et al., 2017; Natividade Joergensen a Zaggl, 2024).

## 1.2 Big data

Big data představují datové sady, které jsou specifické svou velikostí, různorodostí datových typů, rychlostí jejich generování a aktualizace, ale především potenciální hodnotou a přínosy, které lze jejich analýzou získat. Tato data vznikají a jsou propojována z různých zdrojů, včetně online transakcí, e-mailů, videí, obrázků, logů, příspěvků, vyhledávacích dotazů, sociálních interakcí, senzorů nebo mobilních aplikací (Hendl, 2021; Johnson et al., 2017). S tím souvisí to, že jejich efektivní zpracování tradičními nástroji, jako jsou relační databázové systémy, není možné. Tradiční metody správy dat nedokáží zvládnout požadavky na správu těchto objemů, což vedlo k rozvoji nových technik v oblasti data science. Vztah mezi big daty a data science lze přirovnat k procesu rafinace surové ropy – kde data science funguje jako rafinerie, která extrahuje hodnotu z big dat (Cielen et al., 2016).

Big data jsou tradičně definována třemi charakteristikami, známými jako 3V: objem (volume), rychlost (velocity) a různorodost (variety). Někteří autoři k nim přidávají další charakteristiky, např. pravdivost (veracity) a hodnotu (value), což rozšiřuje definici na 5V. Zatímco pravdivost se zaměřuje na přesnost a důvěryhodnost dat, hodnota označuje jejich schopnost přinášet relevantní poznatky, které organizace mohou využít k lepšímu rozhodování a optimalizaci procesů. S neustálým růstem objemů dat rostou i nároky na jejich analýzu, což vyžaduje nové

rámce jako MapReduce, nástroje a platformy jako jsou Apache Hadoop nebo Spark, a zejména vysoce škálovatelnou infrastrukturu schopnou přizpůsobit se zvýšeným požadavkům, tzn. škálování výkonu např. s využitím cloud computingu (Cielen et al., 2016; Jin, 2015). Za posledních 15 let tedy došlo k enormnímu rozvoji v oblasti big dat, což potvrzuje systematická literární rešerše autorů Tosi et al. (2024).

Analýza big dat (big data analytics) je dnes klíčovou činností při práci s daty v organizacích jak v soukromém, tak veřejném sektoru (Akter et al., 2016; Merhi a Bregu, 2020), často s využitím metod umělé inteligence a strojového učení (Zhou et al., 2017). Je jednou z hlavních oblastí data science (Hendl, 2021), když ji lze popsat jako využívání pokročilých analytických metod a nástrojů k zpracování a analýze velkých objemů dat. Zároveň sem patří i jejich ukládání, např. s využitím NoSQL a distribuovaných systémů, a vizualizace, které je důležitá pro pochopení informací získaných z analýzy a zároveň slouží jako vstup pro podporu rozhodování, zpravidla ve vazbě na business intelligence. Big data přinášejí organizacím velké množství příležitostí, ale také s sebou nesou významná rizika (Singh a Reddy, 2015).

### 1.2.1 Přínosy big dat

Tabulka 1 obsahuje přehled hlavních přínosů big dat, kam patří zejména zlepšení rozhodování, personalizace služeb a optimalizaci provozních procesů. Například analýza zákaznických dat umožňuje firmám zacílit reklamní kampaně na konkrétní segmenty trhu, zatímco analýza provozních dat může pomoci při snižování nákladů. Big data také přispívají k inovacím, kdy firmy na základě dat identifikují nové tržní trendy a potřeby zákazníků.

*Tabulka 1:* Přínosy big dat. Zdroj: vlastní.

| **Přínos** | **Popis** | **Zdroje** |
|---|---|---|
| **Zlepšení rozhodování** | Big data umožňují organizacím analyzovat rozsáhlé datové sady, což vede k informovaným a přesnějším rozhodnutím. Ty tak mohou rychle reagovat na trendy a změny na trhu. | Gandomi a Haider (2015), Hashem et al. (2015), Chen et al. (2012), McAfee et al. (2012), Sarker (2021) |
| **Personalizace služeb** | Díky big datům mohou organizace personalizovat nabídky a služby na základě | Davenport (2014), Chen et al. (2012), Provost |

| Přínos | Popis | Zdroje |
|---|---|---|
| | individuálních preferencí zákazníků, což zvyšuje jejich spokojenost a loajalitu. | a Fawcett (2013), Singh a Reddy (2015) |
| **Zvýšení provozní efektivity** | Analýza dat z různých zdrojů umožňuje optimalizovat provozní procesy, snížit náklady a zvýšit efektivitu výroby a služeb. | Davenport (2014), Hashem et al. (2015), Singh a Reddy (2015), White (2012) |
| **Inovace a vývoj produktů** | Big data pomáhají identifikovat nové tržní příležitosti a potřeby zákazníků, což vede k inovacím a vývoji nových produktů. Zároveň umožňují zrychlit celý proces a testovat různé varianty a kombinace, které by dříve nebyly možné analyzovat. | Gandomi a Haider (2015), Chen et al. (2012), Johnson et al. (2017), Natividade Joergensen a Zaggl (2024), Provost a Fawcett (2013) |

### 1.2.2 Rizika big dat

I když big data přinášejí mnoho výhod, jejich implementace přináší i řadu rizik (viz Tabulka 2). Největší výzvy představuje ochrana osobních údajů, kdy únik dat může vážně poškodit pověst organizace. Dalším problémem je nesprávná interpretace dat, kdy špatná analytika může vést k chybným rozhodnutím. Etické otázky, spojené s monitoringem zákazníků a jejich chováním jsou také stále více diskutovány. Pro menší podniky může být velkou překážkou vysoká cena potřebné infrastruktury pro zpracování a ukládání velkých datových objemů.

*Tabulka 2:* Rizika big dat. Zdroj: vlastní.

| Riziko | Popis | Zdroje |
|---|---|---|
| **Bezpečnost a ochrana osobních údajů** | Zpracování velkých objemů dat zvyšuje riziko narušení bezpečnosti a úniku citlivých informací. Nedostatečná ochrana dat může vést k právním problémům a ztrátě důvěry zákazníků. | Gandomi a Haider (2015), Hashem et al. (2015), Jin (2015), Natividade Joergensen a Zaggl (2024), Sun et al. (2020) |

| Riziko | Popis | Zdroje |
|---|---|---|
| **Nesprávné interpretace dat** | Bez správných analytických metod mohou být data špatně interpretována, což vede k nesprávným závěrům a špatným rozhodnutím. | Cielen et al. (2016), Davenport (2014), Provost a Fawcett (2013), Sarker (2021) |
| **Etické otázky** | Používání big dat vyvolává otázky ohledně etiky, zejména v oblasti sledování chování zákazníků a jejich soukromí. | Cielen et al. (2016), Hashem et al. (2015), McAfee et al. (2012), Sun et al. (2020) |
| **Velké náklady na infrastrukturu** | Zpracování a ukládání big dat vyžaduje nákladnou infrastrukturu, což může být pro menší podniky velkou finanční zátěží. | Cielen et al. (2016), Davenport (2014), Johnson et al. (2017), White (2012) |

## 1.3 Python

Python je interpretovaný, vysoce výkonný programovací jazyk, který se používá v různých oblastech jako jsou webové aplikace, automatizace, vědecké výpočty, analýza dat nebo strojové učení. Python je objektově orientovaný, což znamená, že umožňuje organizaci kódu do modulů a tříd, což zlepšuje přehlednost a opakovanou použitelnost kódu (Lutz, 2013). V oblasti data science je Python využívaný díky rozsáhlým knihovnám, jako jsou NumPy, která umožňuje vědecké výpočty, Pandas pro práci s datovými rámci a Matplotlib nebo Seaborn pro vizualizaci dat. Tyto knihovny usnadňují analýzu velkých datových souborů a poskytují různé nástroje pro statistickou analýzu, modelování a vizualizaci dat (Chen a Coulibaly, 2021; McKinney, 2017; Lutz, 2013). Python také obsahuje knihovny pro strojové učení, jako je TensorFlow a Scikit-learn, které umožňují vývoj pokročilých modelů pro prediktivní analýzy a rozpoznávání vzorů (Abadi et al., 2016).

Další výhodou Pythonu je jeho velká a aktivní komunita, která přispívá k vývoji nových nástrojů a rozšíření. To zajišťuje, že Python zůstává aktuální a relevantní pro různé technologické trendy, včetně strojového učení, automatizace a umělé inteligence. Díky tomu je Python jedním z nejdůležitějších programovacích jazyků pro datové vědce a vývojáře napříč obory (Abadi et al., 2016; McKinney, 2017).

## 1.4 Jupyter Notebook

Jupyter Notebook je open-source webová aplikace, která umožňuje uživatelům vytvářet a sdílet dokumenty, které obsahují kód, rovnice, vizualizace a text. Je široce používána ve výzkumu, data science a strojovém učení, protože umožňuje interaktivní analýzu dat. Uživatelé mohou spustit kód přímo v prohlížeči a okamžitě vidět výsledky, což umožňuje rychlou vizualizaci dat (Kluyver et al., 2016).

Jednou z hlavních výhod Jupyter Notebooku je jeho podpora pro více než 40 různých programovacích jazyků, včetně Pythonu, R a Julia, což z něj činí univerzální nástroj pro datové vědce a vývojáře. Díky jeho flexibilitě mohou uživatelé vytvářet dokumenty, které kombinují kód, komentáře a vizualizace, což usnadňuje sdílení a spolupráci na projektech (Perkel, 2018). Kromě toho umožňuje integraci s knihovnami jako Pandas a Matplotlib pro datovou manipulaci a vizualizaci, což poskytuje komplexní nástroje pro analýzu dat v reálném čase (Kluyver et al., 2016).

Další výhodou je schopnost Jupyteru podporovat interaktivní vizualizace a obsah, což je užitečné zejména při práci s velkými datovými sadami a při provádění komplexních analýz. Dokumenty vytvořené v Jupyter Notebooku lze snadno exportovat do různých formátů, jako jsou Portable Document Format (PDF) nebo Hypertext Markup Language (HTML), což umožňuje široké sdílení výsledků výzkumu. Jupyter je také oblíbeným nástrojem mezi učiteli a studenty, protože umožňuje interaktivní výukové materiály a zjednodušuje proces učení (Ragan-Kelley et al., 2014).

## 1.5 CRISP-DM

CRISP-DM je jedna z nejrozšířenějších metodik v oblasti datové analýzy a dolování dat (data mining). Je rozdělena do šesti základních fází, které se mohou iterativně opakovat. Díky své univerzálnosti může být tento přístup využit v různých odvětvích, což přispívá k vyšší efektivitě projektů založených na datech. Základ metodiky položili Fayyad et al. (1996), kteří ve svém článku zdůraznili význam jednotlivých fází, včetně příkladů z praxe a budoucích výzkumných směrů. Hlavní fáze, které se dále dělí na dílčí kroky, jsou (Wirth a Hipp, 2000):

- **Porozumění problematice:** jsou definovány cíle projektu z obchodního hlediska a zajišťuje se, aby byly sladěny s cíli organizace, a zároveň jsou tyto cíle převedeny do datově-analytických úkolů.

- **Porozumění datům:** data jsou analyzována a prozkoumána, aby byly odhaleny potenciální problémy s kvalitou a získány užitečné poznatky.
- **Příprava dat:** data jsou čištěna a transformována tak, aby byla vytvořena finální datová sada vhodná pro modelování, tato fáze je klíčová pro přesnost další analýz.
- **Modelování:** výběr vhodných analytických metod a algoritmů (např. klasifikace, shlukování) a optimalizace parametrů modelů.
- **Vyhodnocení výsledků:** výkonnost modelu je posuzována na základě obchodních cílů, kontroluje se, zda model splňuje požadované standardy a přináší očekávané výsledky.
- **Využití výsledků:** model je implementován do reálného prostředí, kde slouží k podpoře rozhodování a poskytování praktických poznatků.

S rostoucími objemy dat a požadavky na ně se mění i metodika CRISP-DM, např. co se týká agilních metodik, které umožňují zajištění větší flexibility a schopnosti reagovat na měnící se podmínky v datových projektech. Amirian et al. (2024) proto navrhují využívat dílčí časově omezení úkoly v každé fázi CRISP-DM, což podporuje kontinuální zpětnou vazbu a adaptaci. Zahrnutí principů agilního řízení doporučuje i Saltz (2021) a navrhuje rozšíření metodiky o specifické fáze pro automatizaci modelů, DevOps a využití cloudových nástrojů. Shimaoka et al. (2024) pak navrhuje přidání specifických kroků pro zpracování nestrukturovaných dat (např. textová data, obrázky), větší důraz na automatizaci procesů a zahrnutí etických principů a správy dat, které jsou dnes klíčové v datových projektech.

# 2 NÁSTROJE, PLATFORMY A METODY PRO BIG DATA

Kapitola se zaměřuje na nástroje, platformy a metody, které jsou využívány k analýze big dat. Nejdříve je diskutován přehled metod a přístupů k analýze big dat. Následně jsou porovnány související nástroje a platformy, a nakonec jsou podrobně popsány ty, které budou využity pro vytvoření aplikace pro analýzu big dat v Pythonu.

## 2.1 Přehled metod a přístupů k analýze big dat

Při sestavování teoretického přehledu byl brán ohled na konkrétní technologie a infrastrukturní požadavky potřebné pro efektivní práci s big daty, čímž by bylo možné propojit teoretické koncepty s praktickými aspekty implementace. Big data jsou využívána a analyzována napříč odvětvími, včetně zdravotnictví, maloobchodu, finančních služeb a výroby, nebo k vytváření konkurenční výhody a zlepšování provozní efektivity (Gandomi a Haider, 2015). Davenport (2014) a Hashem et al. (2015) rozebírají výzvy a přínosy big dat pro rozhodování v různých sektorech – od zdravotnictví až po finanční sektor. V maloobchodě se big data využívají pro segmentaci zákazníků a personalizaci marketingových kampaní, zatímco ve finančních službách pomáhají při detekci podvodů a řízení rizik (Chen et al., 2012).

Pro efektivní práci s big daty existuje řada metod a technologií. Mezi nejrozšířenější metody patří data mining, které umožňuje odhalovat skryté vzorce, trendy a vztahy v rozsáhlých datových sadách. Techniky jako klasifikace, shlukování a asociační pravidla se používají ke kategorizaci a organizaci dat, což pomáhá objevovat nové souvislosti. Další klíčovou metodou je strojové učení, které zahrnuje algoritmy trénované na identifikaci vzorců v datech a vytváření prediktivních modelů. Tyto metody jsou běžně využívány v prediktivní analýze pro předpovídání budoucích událostí a trendů (Hendl, 2021; Jin, 2015; Johnson et al., 2017; Provost a Fawcett, 2013). Zpracování v reálném čase je dalším důležitým aspektem big dat, kdy jsou data analyzována okamžitě, což organizacím umožňuje rychle reagovat na změny (Kini a Pai, 2024).

Pro analýzu big dat je k dispozici mnoho nástrojů a platforem. Jednou z nejpoužívanějších je open-source platforma Apache Hadoop, která umožňuje distribuované zpracování big dat na clusterech počítačů. Apache Hadoop zahrnuje rámec MapReduce, který efektivně rozděluje a zpracovává data, a Hadoop Distributed File System (HDFS) pro distribuované ukládání dat (White, 2012). Dalším nástrojem je Apache Spark, který umožňuje rychlé zpracování dat v paměti a podporuje pokročilé analytické úkoly, jako je strojové učení a analýza grafů (Chen

a Coulibaly, 2021; Zaharia et al., 2016). Kromě těchto platforem jsou také široce využívány cloudové služby, jako je Amazon Web Services (AWS), Microsoft Azure a Google Cloud, které nabízejí flexibilní infrastrukturu a škálovatelnost pro analýzu big dat, což firmám umožňuje rychle implementovat řešení bez potřeby velkých počátečních investic do infrastruktury pro zpracování a ukládání big dat (Hashem et al., 2015).

Big data tedy vyžadují pokročilou infrastrukturu, která podporuje distribuované zpracování a cloud computing. Pro zajištění efektivního zpracování big dat je nutné investovat do moderního hardwaru a infrastruktury s vysokým výpočetním výkonem, velkou kapacitou úložiště a schopností škálování. Například Apache Hadoop je navržen tak, aby umožňoval ukládání a zpracování dat na více serverech, což zvyšuje odolnost a efektivitu zpracování (White, 2012). Cloudové služby, jako je AWS, Microsoft Azure a Google Cloud, navíc poskytují organizacím flexibilní infrastrukturu, která se může škálovat podle potřeb a umožňuje firmám rychle implementovat řešení bez velkých investic do vlastního hardwaru (Hashem et al., 2015). Proto jsou důležité otevřené technologie, které nabízejí alternativu k nákladným komerčním platformám a umožňují širšímu okruhu uživatelů a organizací efektivně analyzovat big data. Jak uvádí Landset et al. (2015) ve své studii, otevřené nástroje pro strojové učení v ekosystému Hadoop představují robustní a nákladově efektivní řešení pro zpracování big dat. Tyto nástroje, jako je Apache Spark, nabízejí vysokou škálovatelnost a flexibilitu, což je klíčové pro analýzu big sad. Navíc, otevřený zdrojový kód umožňuje komunitní vývoj a inovace, což vede k neustálému zlepšování a rozšiřování funkcí těchto nástrojů. Otevřené technologie tedy mají zásadní vliv na rozvoj analýzy big dat (Tosi et al., 2024).

## 2.2 Porovnání souvisejících nástrojů a platforem

Nástroje a platformy pro práci s big daty hrají klíčovou roli při zpracování a analýze velkých objemů dat, které jsou nezbytné pro získávání cenných poznatků. V literatuře lze nalézt různé přehledy (Chawda a Thakur, 2016; Vijayaraj et al., 2016; Rao et al., 2019; Singh a Reddy, 2015), ale s ohledem na rychlý vývoj této oblasti jsou často již neaktuální, nebo požadavky na analýzu dat jsou specifické a nevhodné pro problematiku řešenou v této práci. Následující Tabulka 3 porovnává nástroje a platformy s ohledem na jejich podporu programovacího jazyka Python a možnosti využití pro analýzy v reálném čase.

*Tabulka 3:* Porovnání vybraných nástrojů a platforem pro analýzu big dat. Zdroj: vlastní.

| **Název** | **Open-source** | **Podpora a rozšíření Python** | **Podpora real-time analýz** |
|---|---|---|---|
| Amazon Elastic MapReduce | NE | ANO, Spark, Hadoop integration | ANO |
| Apache Hadoop | ANO | ANO, Pydoop | NE |
| Apache Spark | ANO | ANO, PySpark | ANO |
| Apache Storm | ANO | ANO, Multi-language support | ANO |
| Cloudera data platform | NE | ANO, Python support | ANO |
| Oracle Data Platform | NE | ANO, Python support | ANO |

Mezi open-source platformy patří Apache Hadoop, Apache Spark a Apache Storm, které jsou široce využívány pro svou komunitní podporu a dostupné knihovny (Hebabaze et al., 2024). Amazon EMR není open-source, ale podporuje nástroje jako Hadoop a Spark a poskytuje pro ně spravovanou službu (Mezzoudj et al., 2024). Cloudera Data Platform obsahuje některé open-source komponenty, zatímco Oracle Data Platform je plně proprietární řešení (Mezzoudj et al., 2024; Rao et al., 2019).

Co se týče podpory Pythonu, Apache Spark vyniká silnou integrací prostřednictvím PySpark, zatímco Apache Hadoop umožňuje použití Pythonu přes Pydoop a Hadoop Streaming (Hebabaze et al., 2024). Apache Storm podporuje více jazyků, včetně Pythonu, ale není tak nativní jako Spark (Mezzoudj et al., 2024). Amazon EMR podporuje Python díky integraci s Hadoopem a Sparkem, zatímco Cloudera Data Platform a Oracle Data Platform umožňují Pythonové skripty, avšak jejich podpora závisí na konkrétní konfiguraci a použitých službách (Mezzoudj et al., 2024; Rao et al., 2019).

Ve schopnosti analýzy v reálném čase Apache Spark vyniká díky in-memory zpracování a podpoře Structured Streaming API (Kini a Pai, 2024; Yenduri, 2024). Apache Storm je

specializován na real-time streamování, zatímco Amazon EMR tuto funkcionalitu umožňuje přes Spark a další nástroje (Mezzoudj et al., 2024). Cloudera Data Platform a Oracle Data Platform podporují analýzu v reálném čase, ale její efektivita závisí na konkrétní konfiguraci a implementovaných nástrojích (Mezzoudj et al., 2024; Rao et al., 2019).

Celkově platí, že open-source řešení jako Apache Hadoop, Apache Spark a Apache Storm nabízejí flexibilitu a komunitní podporu, zatímco proprietární platformy jako Cloudera Data Platform a Oracle Data Platform poskytují integrované podnikové služby. Výběr vhodné platformy závisí na potřebě real-time zpracování dat, preferenci open-source technologií a požadované podpoře Pythonu. Nicméně, kromě těchto základních aspektů je třeba zvážit i další faktory, které ovlivňují rozhodování, např. jak jednotlivé platformy zvládají různé objemy dat a typy úloh. Bezpečnostní aspekty, jako je správa přístupu a ochrana dat, jsou také klíčové, zejména pro organizace, které pracují s citlivými daty.

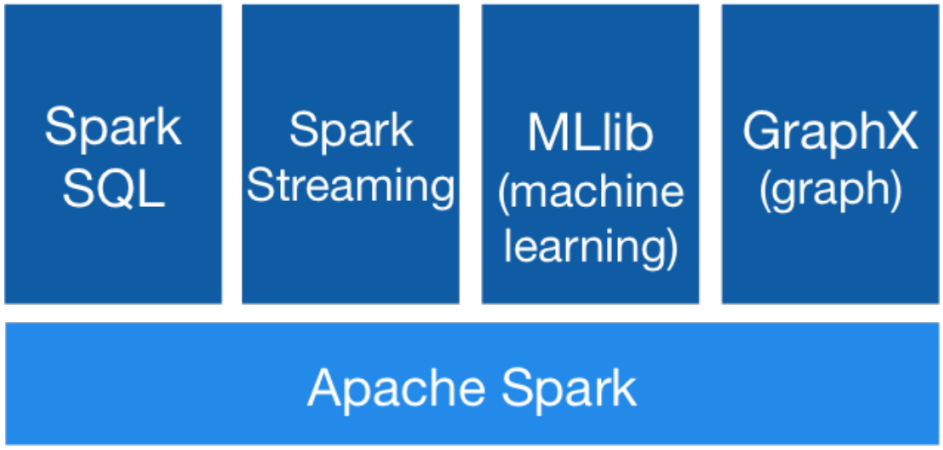
### 2.2.1 Apache Spark

Apache Spark patří mezi nejvýznamnější nástroje pro práci s big daty, zejména díky své rychlosti a flexibilitě. Jedná se o open-source platformu navrženou pro distribuované výpočty, která umožňuje efektivní zpracování dat přímo v paměti. Tento přístup z něj dělá ideální řešení jak pro dávkové zpracování, tak pro práci s daty v reálném čase. Díky své univerzálnosti se Spark využívá k řešení širokého spektra úloh – od základních analýz až po pokročilé aplikace, jako je strojové učení nebo analýza grafů (Chen a Coulibaly, 2021; Zaharia et al., 2016). Jednou z jeho hlavních výhod je schopnost paralelního zpracování a škálovatelnost. Spark efektivně pracuje s obrovskými objemy dat, což ho činí oblíbeným nástrojem pro vědce a analytiky. Na rozdíl od starší technologie Hadoop, která pracuje primárně s daty uloženými na disku, Spark zpracovává data v paměti. Tato vlastnost je klíčová zejména při interaktivní analýze nebo práci s daty v reálném čase, kde je nízká latence zásadní. Díky tomu mohou výzkumníci a firmy rychleji reagovat na změny a získávat přesnější výsledky (Zaharia et al., 2010).

Základem Sparku je technologie Resilient Distributed Dataset (RDD), která umožňuje paralelní zpracování dat. RDD je distribuovaná datová struktura, která rozkládá data na více uzlů v počítačovém klastru. To zvyšuje odolnost vůči selháním, protože data jsou automaticky zálohována. Díky tomu je proces zpracování spolehlivý a snižuje se riziko ztráty dat. Spark také využívá rámec MapReduce, který dělí výpočetní úlohy do dvou kroků – mapování a redukce (Dean a Ghemawat, 2008). Tento přístup umožňuje efektivní zpracování big sad, protože úlohy jsou paralelně rozděleny a následně spojeny do finálního výsledku (Zaharia et al., 2012).

Apache Spark je univerzální nástroj pro zpracování dat, jehož základ tvoří Spark Core využívající RDD pro efektivní práci s různými datovými zdroji (HDFS, Cassandra, HBase, Amazon S3). Díky modulům, jako jsou Spark Structured Query Language (SQL), Spark Streaming, MLlib a GraphX (viz Obrázek 1), nabízí robustní platformu pro analýzu big dat (Duvvuri a Singhal, 2016):

- **Spark SQL**: Umožňuje práci s databázemi a tabulkami pomocí SQL nebo DataFrame API, podporuje formáty Comma-Separated Values (CSV), JavaScript Object Notation (JSON), Parquet a integraci s Apache Hive.
- **Spark Streaming**: Zpracovává data v reálném čase s podporou "exactly-once" sémantiky a více programovacích jazyků (Scala, Java, Python).
- **MLlib**: Nabízí algoritmy pro strojové učení (klasifikace, shlukování, doporučování), škálovatelné pro offline i predikce v reálném čase.
- **GraphX**: Umožňuje analýzu grafů (PageRank, filtrování podgrafů) pro sociální sítě, geografická data či komplexní vztahy.




*Obrázek 1:* Moduly v Apache Spark. Zdroj: Meng at al. (2016).

Apache Spark se využívá v oblasti data science a business analýz díky své rychlosti a schopnosti zpracovávat velké množství dat z různých zdrojů. Jedním z jeho klíčových přínosů je monitorování klíčových ukazatelů výkonnosti, jako jsou prodeje, ziskovost nebo míra retence zákazníků. Například firmy mohou pomocí Sparku propojit data ze zákaznických systémů, transakčních databází a marketingových kampaní, což jim umožňuje lépe porozumět chování zákazníků. Takto propojená data pak slouží jako základ pro tvorbu interaktivních dashboardů a vizualizací, které pomáhají manažerům identifikovat oblasti pro zlepšení. Další klíčovou oblastí je analýza sentimentu zákazníků. Spark dokáže analyzovat recenze, příspěvky na

sociálních sítích nebo zpětnou vazbu z průzkumů a odhalit, jak zákazníci vnímají produkty nebo služby. Prediktivní analýzy, které Spark nabízí, umožňují firmám předpovídat budoucí chování zákazníků nebo poptávku po produktech (Duvvuri a Singhal, 2016; Chen a Coulibaly, 2021; Zaharia et al., 2016).

### 2.2.2 PySpark

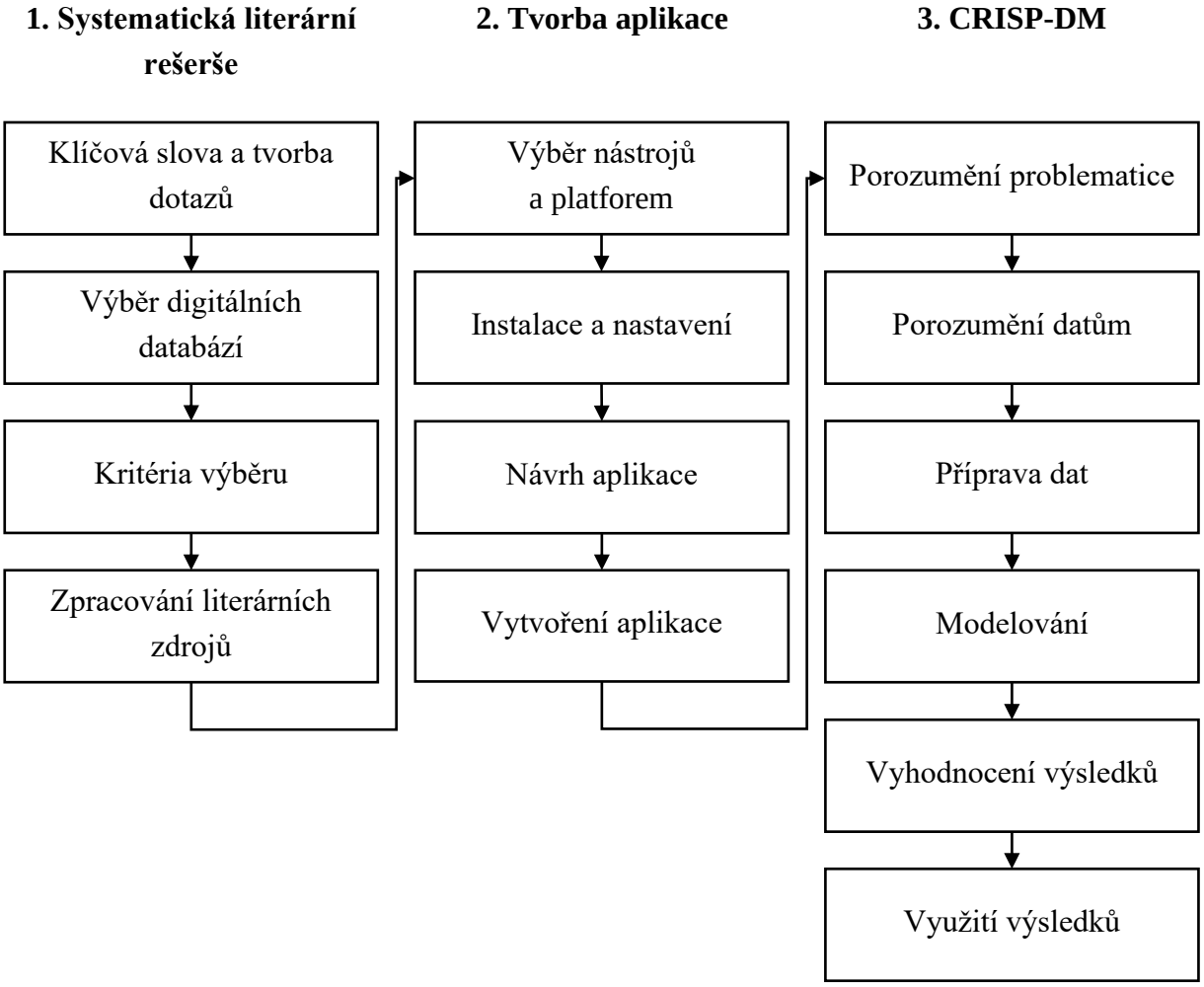
PySpark je distribuovaná výpočetní platforma pro zpracování big dat, která kombinuje sílu Apache Spark a výhody programovacího jazyka Python. Díky integraci se Sparkem umožňuje PySpark snadno provádět analýzu dat na distribuovaných výpočetních clusterech, což je ideální pro práci s velkými datovými sadami. Výhody Apache Spark, jako je vysoká rychlost zpracování a schopnost provádět analýzy v paměti, činí PySpark výkonným nástrojem pro analýzu big data (Chen a Coulibaly, 2021; Zaharia et al., 2016).

PySpark využívá klíčové koncepty Apache Spark, jako jsou RDD, což jsou základní datové struktury umožňující distribuované zpracování dat. Díky RDD mohou uživatelé snadno pracovat s datovými sadami, které jsou rozdělené mezi různé uzly clusteru, čímž se zvyšuje efektivita zpracování. Dále PySpark využívá rámec MapReduce, který umožňuje rozložit složité operace na menší podúlohy a následně je agregovat, což výrazně zlepšuje výkon při zpracování velkých objemů dat (Dean a Ghemawat, 2008).

Jednou z klíčových výhod PySpark je podpora modulů pro specifické typy analýz, jako je MLlib pro strojové učení, GraphX pro práci s grafy a Spark SQL pro dotazování nad strukturovanými daty. Díky těmto modulům mohou uživatelé snadno aplikovat pokročilé algoritmy strojového učení nebo provádět složité dotazy v rámci big data analýz. PySpark je proto široce používán v různých oblastech, jako jsou finanční analýzy, doporučovací systémy nebo zpracování přírodního jazyka. Jeho schopnost provádět analýzy v reálném čase z něj činí ideální volbu pro aplikace, kde je okamžitý přehled o datech nezbytný (Meng et al., 2016).

# 3 METODIKA A POSTUP VYTVOŘENÍ APLIKACE

V kapitole je popsán metodický postup řešení cíle práce, tzn. *charakterizovat současné přístupy k analýze big dat, porovnat existující nástroje a metody, provést sběr a předzpracování dat, analyzovat data, navrhnout modely, a nakonec implementovat celé řešení prostřednictvím aplikace v Pythonu*. Obsahuje kroky systematické literární rešerše, která byla provedena za účelem identifikace současných trendů k analýze big data, a porovnání a výběru vhodných nástrojů a platforem. Dále je zde uveden postup tvorby aplikace, včetně výběru a nastavení nástrojů a platforem, přípravy prostředí pro analýzu dat, a návrhu a vytvoření aplikace. Celý postup vytvoření aplikace pro analýzu big dat v Pythonu je následně podrobně specifikován podle fází metodiky CRISP-DM. Vše je zobrazeno na následujícím obrázku:




*Obrázek 2:* Metodický postup řešení cíle práce. Zdroj: vlastní.

## 3.1 Systematická literární rešerše

Kroky použité v rámci systematické literární rešerše jsou založeny na zjištěních Kitchenham et al. (2009). Autoři se ve své publikaci zaměřují na oblast softwarového inženýrství a doporučují důsledně aplikovat všechny potřebné kroky, především vymezení cíle a výzkumných otázek. S ohledem na cíl této diplomové práce tak byly zformulovány 3 oblasti, pro které byly následně definovány klíčová slova a vytvořeny dotazy. Pro **první oblast** zaměřenou na současné přístupy k analýze big dat byla definována klíčová slova tvořící dotaz: "big data analysis" OR "big data analytics" OR "big data processing" AND "method" OR "approach" AND "trend". Pro **druhou oblast** zaměřenou na nástroje pro analýzu big dat byla zvolena tato klíčová slova tvořící dotaz: "big data analysis" OR "big data analytics" OR "big data processing" AND "tool" OR "platform" OR "service" AND "comparison" OR "overview" OR "survey". Pro **poslední oblast**, které se věnuje návrh aplikace v Pythonu, byla definována klíčová slova tvořící dotaz: "big data analysis" OR "big data analytics" OR "big data processing" AND "application" OR "solution" AND "Python" AND "development". První dvě oblasti slouží jako základ pro teoretické a praktické porozumění problematice big dat a jejich analýzy. Třetí oblast je důležitá pro návrh vlastní aplikace, především co se týká její tvorby a použitých technologií a postupů.

Dalším krokem byl výběr digitálních databází, ve kterých byly vytvořené dotazy aplikovány. Za tímto účelem byly zvoleny databáze Scopus, Web of Science a Google Scholar. Nalezené výsledky byly dále filtrovány tak, aby definovaná klíčová slova byla obsažena přímo v názvu, abstraktu nebo klíčových slovech nalezeného zdroje. Kategorie zdrojů pak byly omezeny na Computer Science, Decision Sciences, Engineering, Information Systems, Software, Economics, Business a Management. Posledním kritériem byl přístup k plnému textu, tzn. buď open access nebo přes předplatné Univerzity Pardubice. Nakonec byly vybrány vhodné zdroje a zpracovány podle příslušnosti k dané oblasti.

## 3.2 Tvorba aplikace

### 3.2.1 Výběr nástrojů a platforem

Obrázek 3 zachycuje jednotlivé nástroje a platformy použité pro tvorbu aplikace pro analýzu big dat. Apache Spark je klíčovou platformou pro distribuované výpočty. Spark podporuje jak dávkové, tak proudové zpracování, což umožní efektivní agregaci a analýzu dat. Funkce Apache Spark jsou zpřístupněny prostřednictvím PySparku, což je Python API umožňující

pracovat s distribuovanými daty pomocí programovacího jazyka Python. Tímto je zajištěna snadná integrace a flexibilita při práci s daty.


*Obrázek 3:* Hlavní nástroje a platformy použití při vytváření aplikace. Zdroj: vlastní.

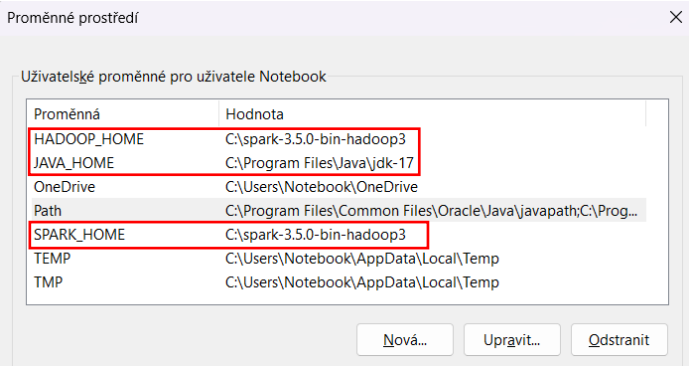
Pro ukládání dat je využit formát PARQUET, který je orientovaný na sloupce. Díky němu je optimalizována velikost ukládacích prostorů a zrychlen přístup k datům při jejich analýze. Běh Apache Spark je zajištěn prostřednictvím Javy 17, která poskytuje stabilitu a potřebný výkon pro distribuované aplikace. Tato verze Javy zajišťuje také kompatibilitu s moderními technologiemi. Hlavním programovacím jazykem je Python, který slouží jako propojující článek mezi všemi komponentami. Díky širokému spektru dostupných knihoven je možné provádět analýzu, vizualizaci dat, a dokonce i implementaci pokročilých algoritmů strojového učení. K vývoji, ladění a prezentaci je použito interaktivní prostředí Jupyter Notebook, které umožňuje kombinovat kód, text i vizualizace v jednom dokumentu. Pro čištění a přípravu dat před samotnou analýzou je použita aplikace Excel, která umožňuje odstranit nekonzistentní údaje, sjednotit formáty a připravit dataset pro další zpracování v jiných nástrojích.

Pro manipulaci a analýzu dat je použita knihovna Pandas, která umožňuje snadnou práci s datovými rámci. Pomocí této knihovny je provedena příprava dat, například filtrování záznamů na základě konkrétních hodnot nebo odstraňování chybějících dat. Také je využita pro základní agregace, jako je výpočet průměrného počtu zločinů v jednotlivých měsících, před dalším zpracováním v PySparku nebo Apache Sparku. Vizualizace dat jsou realizovány pomocí knihovny Matplotlib, která umožňuje tvorbu statických i interaktivních grafů. Geografická data jsou vizualizována pomocí knihovny Folium, která umožňuje zobrazit interaktivní mapy. Pro vědecké výpočty je použita knihovna NumPy, která slouží jako základní analýzu dat a pro práci s vícerozměrnými poli.

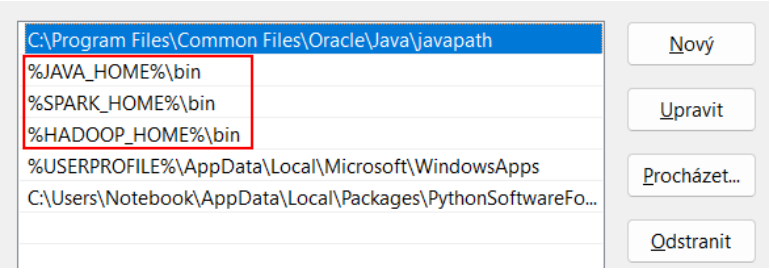
### 3.2.2 Instalace a nastavení

Níže popsaná instalace je určená pro Windows 11. Každý operační systém má odlišný postup instalace. Nejdříve je nutné stáhnout potřebné instalační soubory pro Apache Spark, Java JDK 17 a Winutils.exe. Stažený archiv s Apache Spark je potřeba extrahovat do počítače (např: *C:\Apache…*). Nainstalování Java JDK 17 (podporované verze Spark jsou 8, 11, 17). Nicméně je důležité poznamenat, že podpora pro Java 8 a 11 bude ukončena v příští hlavní verzi Spark. To znamená, že minimální podporovaná verze Javy bude Java 17. Z archivu winutils se extrahuje složka s verzí Hadoopu, která se nachází ve složce s Apache Spark. Tato extrahovaná složka se vloží do složky „bin“ (např: *C:\Apache…\bin*). V příkazovém řádku se provede instalace pomocí příkazu pro přechod do adresáře C:\ (*cd C:*) a poté zkopírování a následné vložení příkazu do příkazového řádku. Instalaci potřebných knihoven obsahuje Příloha I.

Pro nastavení systémových proměnných je nutné ve Windows 11 spustit systémovou aplikaci – do vyhledávacího pole operačního systému se zadá text: *Upravit proměnné prostředí systému.* Kliknutím na tlačítko: *Proměnné prostředí…* se otevře dialogové okno. V horní části Uživatelské proměnné pro uživatele… se klikne na tlačítko *Nová…* Jako Název se zadá: *JAVA_HOME* a jako Hodnota se zadává cesta do adresáře, kam se nainstalovala Java JDK 17 (Většinou je v adresáři: C:\Program Files\Java\jdk-17) a následné OK (viz Obrázek 4). Přidá se další proměnná s názvem: *SPARK_HOME* a jako hodnota se zadá cesta, kam se extrahovala složka s Apache Spark. Přidání další proměnné s názvem: *HADOOP_HOME* a jako hodnota se opět zadává cesta, kde je extrahována složka s Apache Spark. Následně se vyhledá proměnná *„Path“*, vybere se a klikne se na tlačítko: *Upravit…* Přidá se nová cesta: *%JAVA_HOME%\bin*, jako další cesta se zadá: *%SPARK_HOME%\bin* a třetí cesta: *%HADOOP_HOME%\bin* (viz Obrázek 5). Nakonec se ve všech otevřených oknech klikne na „*OK“*

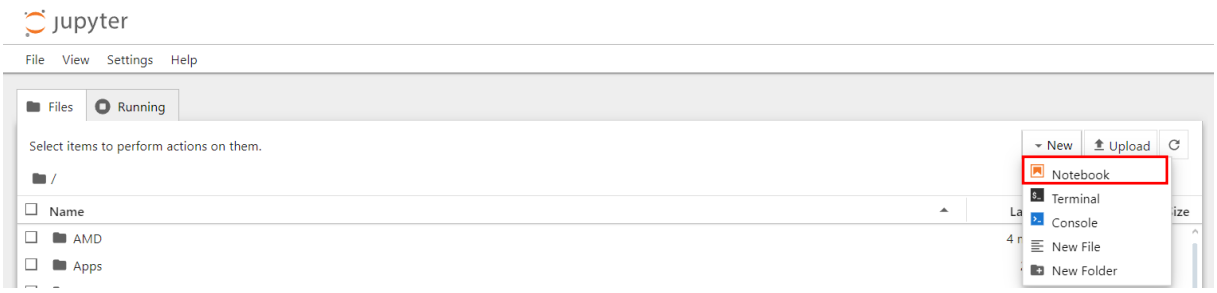



*Obrázek 4:* Ukázka nastavení systémových proměnných. Zdroj: vlastní.




*Obrázek 5:* Ukázka nastavení systémových proměnných v proměnné PATH. Zdroj: vlastní.

Dalším krokem je spuštění Jupyter Notebook. Do příkazové řádky v adresáři C:\ se zadá *jupyter notebook* a stiskne se Enter. Následně by se měl Jupyter otevřít v prohlížeči. V dalším kroku je potřeba se přesunout do nějaké složky a následně se v pravém horním rohu rozklikne *New* a výběr *Notebook* (viz Obrázek 6). Pokud by se nepřesunulo do složky, tak přímo v adresáři C:\ nelze zapnout Notebook. Popřípadě, pokud je potřeba se přepnout do jiného adresáře, tak stačí kliknout na Terminal a za pomocí cd (např. cd D:\) se lze přesunout do jiného adresáře a zadává se znovu příkaz Jupyter Notebook.




*Obrázek 6:* Ukázka zapnutí notebooku v prostředí Jupyter Notebook. Zdroj: vlastní.

### 3.2.3 Návrh aplikace

Návrh aplikace vychází z rešerše literatury, která poskytuje návody a doporučení toho, jak při vytváření aplikace pro analýzu big dat v Pythonu postupovat. Návrh aplikace pro analýzu big dat v jazyce Python vychází z postupů a metod popsaných v odborné literatuře. Hlavními zdroji jsou Cielen et al. (2016), Duvvuri a Singhal (2016), Chen a Coulibaly (2021), McKinney (2017) a Zaharia et al. (2016), které poskytují návod k efektivnímu zpracování, analýze a vizualizaci rozsáhlých datových souborů. Tyto principy jsou aplikovány s cílem zajistit maximální výkon, škálovatelnost a využitelnost aplikace.

Při návrhu architektury aplikace je kladen důraz na využití frameworku Tkinter pro uživatelské rozhraní a technologie Apache Spark pro zpracování dat. Tento přístup je podložen doporučeními Chen a Coulibaly (2021) a Zaharia et al. (2016), kteří zdůrazňují výhody distribuovaného zpracování dat pomocí Apache Spark, a McKinney (2017), jenž se zabývá

efektivním využíváním datových struktur v jazyce Python. Integrace grafického uživatelského rozhraní umožňuje přístupnost aplikace i pro uživatele bez hlubších technických znalostí, což odpovídá doporučením Cielen et al. (2016) týkajících se uživatelské přívětivosti analytických nástrojů.

Zpracování dat začíná jejich načtením a převodem z formátu CSV na Parquet, který je považován za nejvhodnější pro rozsáhlé datové operace díky své efektivitě a kompresním schopnostem (Duvvuri a Singhal, 2016). Tento krok zajišťuje rychlejší přístup k datům a umožňuje škálovatelné výpočty v rámci Apache Spark. Správná manipulace s datovými typy a zajištění integrity dat během této konverze jsou zásadní pro bezchybné fungování následných analytických procesů. Funkcionalita aplikace je postavena na využití SQL dotazů pro analýzu četnosti výskytů různých jevů v časových intervalech, přičemž důraz je kladen na agregaci a přesnost statistik. Tento přístup reflektuje doporučení popsaná McKinney (2017), který zdůrazňuje význam dobře strukturovaných dat pro analytické modely. Veškeré neplatné hodnoty jsou detekovány a odstraněny, čímž se zajišťuje, že výsledné vizualizace poskytují správné a přehledné informace. K prezentaci výsledků jsou využívány interaktivní grafy vytvořené pomocí knihovny Matplotlib. Tyto grafy umožňují intuitivní interpretaci dat, což odpovídá doporučením Cielen et al. (2016) o potřebě srozumitelných vizualizací při práci s big daty. Speciální důraz je kladen na formátování popisků, jasnou strukturu vizualizací a možnost uživatelského filtrování dat, což zvyšuje přehlednost a použitelnost aplikace.

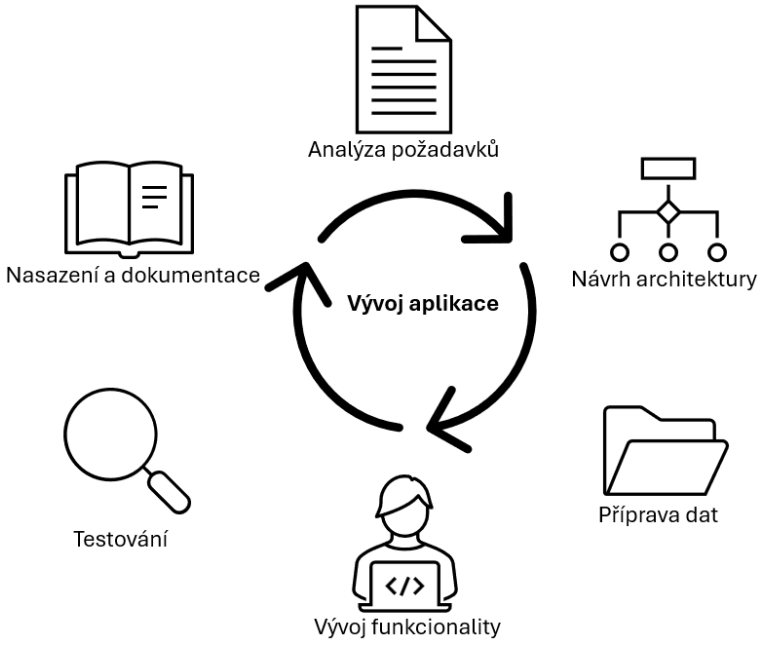
Při návrhu uživatelského rozhraní je zohledněna ergonomie a snadná navigace, jak doporučují Duvvuri a Singhal (2016). Menu aplikace umožňuje výběr specifických parametrů pro detailní analýzu, což uživatelům poskytuje flexibilitu při zkoumání dat. V neposlední řadě je v návrhu aplikace kladen důraz na bezpečnost a robustnost systému, aby byla minimalizována možnost chyb a ztráty dat, jak je popsáno v doporučeních Zaharia et al. (2016) týkajících se odolnosti distribuovaných výpočtových systémů.

Výsledná aplikace je navržena tak, aby byla výkonná, škálovatelná a uživatelsky přívětivá, přičemž vychází z moderních přístupů k analýze big dat. Její architektura a implementace reflektují osvědčené postupy popsané v odborné literatuře a umožňují efektivní práci s big daty.

### 3.2.4 Vytvoření aplikace

Proces vývoje aplikace byl rozdělen do několika klíčových fází (viz Obrázek 7), které zajistily systematický postup od počáteční analýzy až po nasazení aplikace. Při vývoji byl kladen důraz

na flexibilitu a pravidelnou kontrolu postupu, což umožnilo průběžné přizpůsobení podle aktuálních potřeb a priorit. Práce probíhala v jednotlivých etapách, které trvaly přibližně dva týdny, a po každé z nich byly výsledky vyhodnoceny a případné úkoly upraveny.




*Obrázek 7:* Fáze vývoje aplikace. Zdroj: vlastní.

**Analýza požadavků**

Na začátku vývoje proběhla důkladná analýza uživatelských a technologických požadavků. Nejprve byly stanoveny hlavní cíle projektu, identifikovány klíčové funkce a analyzovány dostupné datové sady. Tyto kroky vedly k vytvoření seznamu specifikací a funkcí, který posloužil jako základní rámec pro další fáze vývoje.

**Návrh architektury aplikace**

Na základě analýzy požadavků byl navržen technologický stack zahrnující Apache Spark, PySpark a Jupyter Notebook, a také struktura databáze. Součástí této fáze bylo vytvoření návrhů uživatelského rozhraní pomocí wireframů a výběr nástrojů pro vizualizaci dat, jako je Matplotlib nebo Folium.

**Příprava dat**

Dalším krokem byla příprava dat pro analýzu. Získaná data byla očištěna, sjednocena a převedena z formátu CSV na PARQUET, což zlepšilo efektivitu zpracování. Velká pozornost byla věnována odstranění nekonzistencí v datech a zachování jejich kvality.

### Vývoj funkcionality

Vývoj aplikace probíhal iterativně. Nejprve byly implementovány základní funkce, jako je načítání a zpracování dat, a postupně byly přidávány pokročilejší moduly. Ty zahrnovaly například vizualizaci trendů a geografickou analýzu. Průběžné testování zajistilo stabilitu aplikace a minimalizovalo riziko chyb.

### Integrace a testování

Po implementaci všech funkcí následovala jejich integrace. Dále bylo provedeno důkladné testování jednotlivých částí aplikace, aby se ověřila jejich funkčnost a vzájemná kompatibilita. Součástí tohoto kroku bylo také testování uživatelského rozhraní, zaměřené na snadné ovládání a přívětivost aplikace.

### Nasazení a dokumentace

V závěrečné fázi byla aplikace nasazena do produkčního prostředí. Vytvoření aplikace bylo završeno sepsáním dokumentace. Popis ovládání aplikace je uveden v kapitole *Uživatelské rozhraní a funkce aplikace*.

# 4 SPECIFIKACE ŘEŠENÍ APLIKACE PODLE METODIKY CRISP-DM

V této kapitole je podrobně rozepsáno vytvoření aplikace pro analýzu big dat v Pythonu podle fází metodiky CRISP-DM. Použití této metodiky poskytuje komplexní pohled na celý proces vytvoření aplikace, včetně jejích funkcionalit, které jsou ilustrovány na analýze reálného big datasetu.

## 4.1 Porozumění problematice

Jak bylo zmíněno již v úvodu, s narůstajícím množstvím open dat a různorodostí jejich formátů i struktur se stává jejich efektivní analýza čím dál složitějším úkolem. Každý den se na různé datové portály nahrávají velké objemy dat, které pocházejí z veřejných institucí, akademické sféry, neziskových organizací i soukromého sektoru. Tyto datasety mohou obsahovat informace o různých oblastech, jako jsou finance, doprava, zdravotnictví nebo kriminalita. Právě proto se stále větší důraz klade na nástroje, které uživatelům usnadní jejich analýzu a interpretaci.

Hlavním cílem datových portálů by mělo být umožnit uživatelům snadno pracovat s datasety. Proto je důležité, aby nabízely širokou škálu analytických nástrojů a vizualizací. Často se ale stává, že dostupné nástroje nejsou dostatečně pokročilé nebo intuitivní, což práci s daty komplikuje. Navržená aplikace proto přináší řešení, které pomůže s efektivním zpracováním big dat a jejich analýzou různými metodami.

Je důležité zmínit, že ideálním řešením by bylo přímo integrovat funkcionality navržené aplikace do samotných datových portálů. Taková implementace by však vyžadovala spolupráci různých aktérů, včetně správců portálů, vývojářů a datových analytiků. Jelikož takový proces může být zdlouhavý a náročný na koordinaci, hlavním smyslem této aplikace je ukázat, že efektivní analýza velkých datasetů je možná a že ji lze realizovat pomocí vhodných nástrojů a metod.

Jako zdroj big dat byl vybrán portál DATA.GOV, což je oficiální datový portál vlády Spojených států amerických. Tento portál byl zvolen z několika důvodů:

1. **Široká nabídka datasetů:** DATA.GOV poskytuje obrovské množství datasetů napříč různými oblastmi, od ekonomiky po životní prostředí. Mnoho z těchto datasetů splňuje kritéria big dat, jako jsou velký objem, různorodost a vysoká rychlost aktualizací.
2. **Metadata a popisy:** Každý dataset obsahuje podrobná metadata, popisy a informace o licencích, což usnadňuje jejich porozumění a legální využití.

3. **Anglický jazyk:** Všechny datasety jsou dostupné v anglickém jazyce, což je velkou výhodou, jelikož většina analytických nástrojů používaných v tomto projektu jsou také v angličtině. To eliminuje problémy spojené s jazykovými bariérami a usnadňuje implementaci.

S ohledem na stanovené cíle jsou tedy požadavky na aplikaci zaměřeny především na různé typy analýz a vizualizací, včetně těch interaktivních. Díky tomu budou moci uživatelé efektivně pracovat s big datasety a snadno z nich získat informace. Zároveň aplikace poskytne nástroje pro transformaci a vizualizaci dat tak, aby byla analýza co nejpřístupnější a nejefektivnější.

## 4.2 Porozumění datům

### 4.2.1 Sběr vstupních dat

Tento dataset zobrazuje hlášené případy trestné činnosti (s výjimkou vražd, kde jsou k dispozici data pro každou oběť), které se staly ve městě Chicago od roku 2001 až do současnosti, s výjimkou posledních sedmi dnů. Data jsou extrahována ze systému Citizen Law Enforcement Analysis and Reporting (CLEAR) Chicagského policejního oddělení. Aby bylo zachováno soukromí obětí trestných činů, adresy jsou zobrazeny pouze na úrovni bloků a konkrétní místa nejsou identifikována. Poskytuje náhledy do distribuce trestné činnosti, typů trestných činů a demografické ukazatele trestné činnosti po celém Chicagu (City of Chicago, 2025). Tento dataset je získán z webové stránky https://catalog.data.gov/dataset/crimes-2001-to-present.

V datasetu jsou data o datovém typu: rok zločinu, datum zločinu, textová data (např. Description, Primary Type atd.), boolean hodnoty (např. Arrest, Domestic).

### 4.2.2 Popis dat

V následující tabulce je vidět částečný datový slovník. Částečný z toho důvodu, že jsou ponechány v tabulce pouze atributy, které jsou použity pro analýzy. Kompletní datový slovník obsahuje Příloha II.

*Tabulka 4:* Popis sloupců ve vybraném datasetu. Zdroj: City of Chicago (2025).

| | |
|---|---|
| **Date** | Datum, kdy k incidentu došlo (někdy pouze jako odhad). |
| **Primary Type** | Primární popis kódu Illinois Uniform Crime Reporting (IUCR). |
| **Description** | Sekundární popis kódu IUCR, podkategorie primárního popisu. |
| **Location Description** | Popis místa, kde k incidentu došlo. |
| **Arrest** | Označuje, zda došlo k zatčení. |
| **Domestic** | Označuje, zda se incident týkal domácího prostředí, jak je definováno zákonem Illinois Domestic Violence Act. |
| **Year** | Rok, kdy k incidentu došlo. |
| **Updated On** | Datum a čas poslední aktualizace záznamu. |
| **Latitude** | Zeměpisná šířka místa, kde k incidentu došlo. |
| **Longitude** | Zeměpisná délka místa, kde k incidentu došlo. |

### 4.2.3 Průzkum dat

Prediktivní modelování trestné činnosti je založeno na využití analytických technik, díky nimž jsou identifikovány oblasti s vyšším rizikem trestné činnosti. Tímto způsobem mohou být policejní zásahy lépe cíleny, což umožňuje nejen předcházet budoucím trestným činům, ale také efektivněji řešit ty již spáchané. Dále je možné pomocí těchto modelů identifikovat potenciální pachatele nebo oběti. Výzkumy ukázaly, že datově založené predikční modely mohou předpovídat budoucí trestné činy s přesností až 90 % (Perry et al., 2013; Wood, 2022). Sociální a ekonomické faktory, jako je chudoba, nezaměstnanost, sociální nerovnost či rychlá urbanizace, byly označeny za klíčové činitele ovlivňující míru trestné činnosti. Zjistilo se, že vysoká nezaměstnanost a prudká urbanizace významně přispívají k nárůstu trestné činnosti. Tyto faktory byly opakovaně potvrzeny jako zásadní ve vztahu k výskytu kriminality v různých prostředích (Gokmenoglu et al., 2020; Vargas, 2023).

Optimalizace rozmístění policejních sil je dalším důležitým aspektem prevence kriminality. V této oblasti bylo využito metod alokace zdrojů, díky nimž může být efektivněji zajištěno rozmístění jednotek v rizikových oblastech. Bylo zjištěno, že správné rozmístění policejních sil výrazně zlepšuje rychlost reakce na trestné činy a celkově zvyšuje bezpečnost v daném

prostředí (Mukhopadhyay et al., 2016). Výzkum vztahu mezi sezónností a trestnou činností ukázal, že teplejší počasí často souvisí se zvýšením míry trestných činů, zejména u určitých typů kriminality. Demografické faktory, jako je věk, pohlaví, geografická poloha, vzdělání, etnicita či zneužívání návykových látek, byly identifikovány jako významné proměnné ovlivňující vzorce trestné činnosti a oběťmi těchto činů. Například rozdíly v demografických charakteristikách často vedou k odlišným rizikům spáchání nebo zažití konkrétních typů trestné činnosti (Flowers, 1989).

### 4.2.4 Ověření kvality dat

Tyto trestné činy mohou být založeny na předběžných informacích dodaných policejnímu oddělení hlášenými stranami, které nebyly ověřeny. Předběžné klasifikace trestných činů mohou být později změněny na základě dalšího vyšetřování a vždy existuje možnost mechanické nebo lidské chyby. Proto Chicagské policejní oddělení negarantuje přesnost, úplnost, aktuálnost nebo správné sekvencování informací a informace by neměly být používány pro srovnávací účely v čase. Data jsou aktualizována denně. Seznam kódů Chicagského policejního oddělení – IUCR je k dispozici na webové stránce města Chicago (City of Chicago, 2025).

## 4.3 Příprava dat

Tato část se věnuje zpracování dat před analýzou. Je popsán proces sběru dat, jejich čištění a převodu do vhodných formátů (například CSV na PARQUET). Je zde zdůrazněna důležitost odstraňování chyb a nekonzistencí v datových sadách. V datové sadě jsou ponechány všechny sloupce z důvodu možného použití pro budoucí analýzy.

### 4.3.1 Výběr dat

Pro analýzu byla vybrána data o kriminalitě v Chicagu za období 2001–současnost. Výběr dat byl proveden na základě jejich relevance pro predikci kriminality a možnosti jejich kvalitního zpracování. Nebyla zahrnuta data, která neobsahovala důležité informace pro modelování, například záznamy bez klíčových atributů.

### 4.3.2 Čištění dat

Ve sloupci Description jsou nahrazeny řetězce jako např. POSS: CRACK řetězcem POSSESS: CRACK, POSS: HEROIN nahrazeno POSSESS: HEROIN, AGG PRO.EMP: HANDGUN

nahrazeno AGGRAVATED PROTECTED EMPLOYEE – HANDGUN atd. Řetězce jsou nahrazeny z důvodu stejného významu, protože byl použit ze strany Chicagské policie zkrácený popis. Důvodem nahrazení také bylo, že kdyby se nechaly všechny tvary v datasetu (dva tvary o stejném významu), tak by byly modely nepřesné z důvodu počtu rozdílných hodnot, a tudíž špatné orientace a výsledku.

Co se týká prázdných hodnot, tak ty jsou pouze u geografických souřadnic. Tyto hodnoty jsou ponechány prázdné z důvodu zamezení, aby se analýza nestala méně kvalitní. Kdyby se doplňovaly souřadnice náhodně, tak při analýze by mohlo vyplynout, že mnoho zločinů se stalo na některých souřadnicích, kde ve skutečnosti nemusely některé zločiny proběhnout, a tudíž by byla analýza nepřesná a také nespolehlivá.

### 4.3.3 Tvorba a úpravy dat

Nebyla potřeba vytvářet nové atributy, protože dataset obsahoval dostatek informací. Nebylo nutné agregovat nebo odstraňovat záznamy, protože každý případ představuje unikátní událost. Tato skutečnost usnadnila proces předzpracování dat a umožnila se soustředit na samotnou analýzu a vizualizaci dat. Absence úprav také minimalizovala riziko zkreslení výsledků analýzy a zajistila, že výsledky jsou založeny na původních, neupravených datech.

### 4.3.4 Formátování dat

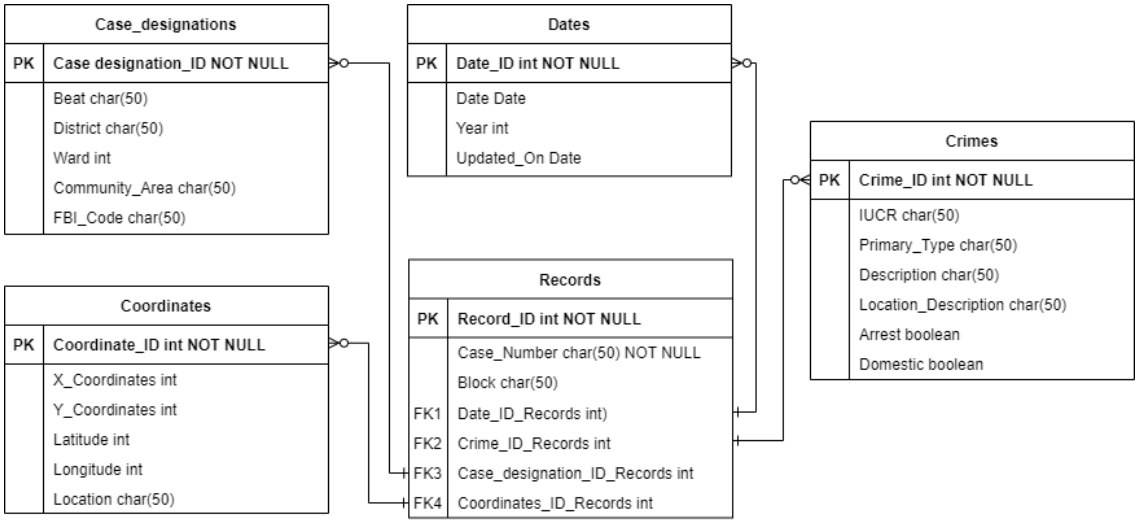
Data byla sjednocena do standardizovaného formátu, aby bylo možné efektivně provádět analýzy v PySparku a dalších big data nástrojích. Například datové typy byly sjednoceny pro lepší kompatibilitu s analytickými nástroji. Tento krok je klíčový pro zajištění konzistence a spolehlivosti výsledků analýzy.

### 4.3.5 Převod CSV na PARQUET

Big dataset CSV soubor se převádí na soubory PARQUET. Převod CSV na Parquet je výhodný z několika důvodů. Parquet je sloupcově orientovaný formát, který nabízí efektivní kompresi a kódování dat, což vede k výraznému snížení nároků na úložiště, až o 75 % ve srovnání s CSV. Tento formát také umožňuje rychlejší načítání specifických sloupců, což urychluje analýzu dat a snižuje vstupně-výstupní operace. Parquet podporuje evoluci schématu, což usnadňuje přidávání nových sloupců do dat bez nutnosti přeformátování celého datasetu. Dále je kompatibilní s Hadoop ekosystémem, včetně nástrojů jako Apache Spark a Hive, a podporuje

složité datové struktury, což je užitečné pro pokročilé analytické úlohy (Levy, 2023; Snowflake, 2024). Kód pro převod CSV souboru na PARQUET obsahuje Příloha III.

Obrázek 8 zachycuje návrh databáze, kde jsou data rozdělená do tabulek tak, aby se s datasetem lépe pracovalo. Takový způsob práce s databází usnadňuje nejen její správu, ale taky zrychluje analýzu dat. Když se totiž data správně rozdělí, při práci s konkrétní částí informací se načtou jen ty tabulky, které jsou potřeba, místo toho, aby se načítala celá databáze. To šetří paměť i čas, protože není nutné pracovat s obrovským množstvím dat najednou. Celý proces je tak rychlejší a méně náročný na výkon počítače.




*Obrázek 8:* Navržená databáze. Zdroj: vlastní.

### 4.3.6 Datová sada

Finální datová sada byla vytvořena na základě zaznamenaných trestných činů v Chicagu od roku 2001 po současnost. Každý záznam obsahuje informace o datu a čase spáchání činu, jeho primárním typu a podrobném popisu, údaj o případném zatčení, klasifikaci jako domácí násilí, přesnou geografickou polohu (pokud je dostupná), identifikátor policejního okrsku a územního celku a další související atributy. Dataset byl očištěn od nekonzistentních a redundantních informací, přičemž všechny zachované sloupce byly standardizovány pro usnadnění integrace do analytických modelů. Díky převodu do formátu Parquet je umožněno efektivní zpracování ve velkých distribuovaných systémech, což přispívá k lepším analýzám i prediktivnímu modelování kriminality.

## 4.4 Modelování

V kapitole jsou popsány techniky modelování dat, včetně výběru modelovacích metod. Dále je uveden návrh a testování modelů, jejich validace a vytvoření uživatelského rozhraní aplikace, která umožňuje interaktivní analýzy. Na následujícím obrázku (Obrázek 9) je výpis všech sloupců obsažených v datovém souboru včetně datových typů:

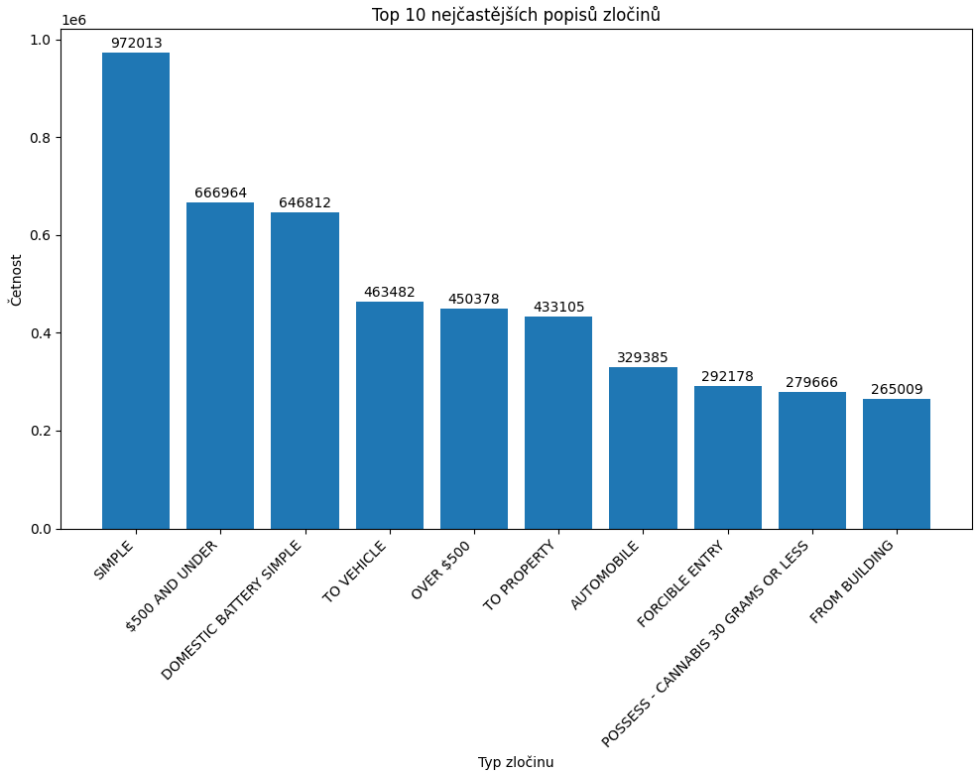
```
>>> data.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7981142 entries, 0 to 7981141
Data columns (total 22 columns):
 #   Column                Dtype
---  ------                -----
 0   ID                    int64
 1   Case Number           object
 2   Date                  object
 3   Block                 object
 4   IUCR                  object
 5   Primary Type          object
 6   Description           object
 7   Location Description  object
 8   Arrest                bool
 9   Domestic              bool
 10  Beat                  int64
 11  District              float64
 12  Ward                  float64
 13  Community Area        float64
 14  FBI Code              object
 15  X Coordinate          float64
 16  Y Coordinate          float64
 17  Year                  int64
 18  Updated On            object
 19  Latitude              float64
 20  Longitude             float64
 21  Location              object
dtypes: bool(2), float64(7), int64(3), object(10)
memory usage: 1.2+ GB
```

*Obrázek 9:* Výpis názvů sloupců a datových typů v datovém souboru. Zdroj: vlastní.

### 4.4.1 Výběr modelovacích technik

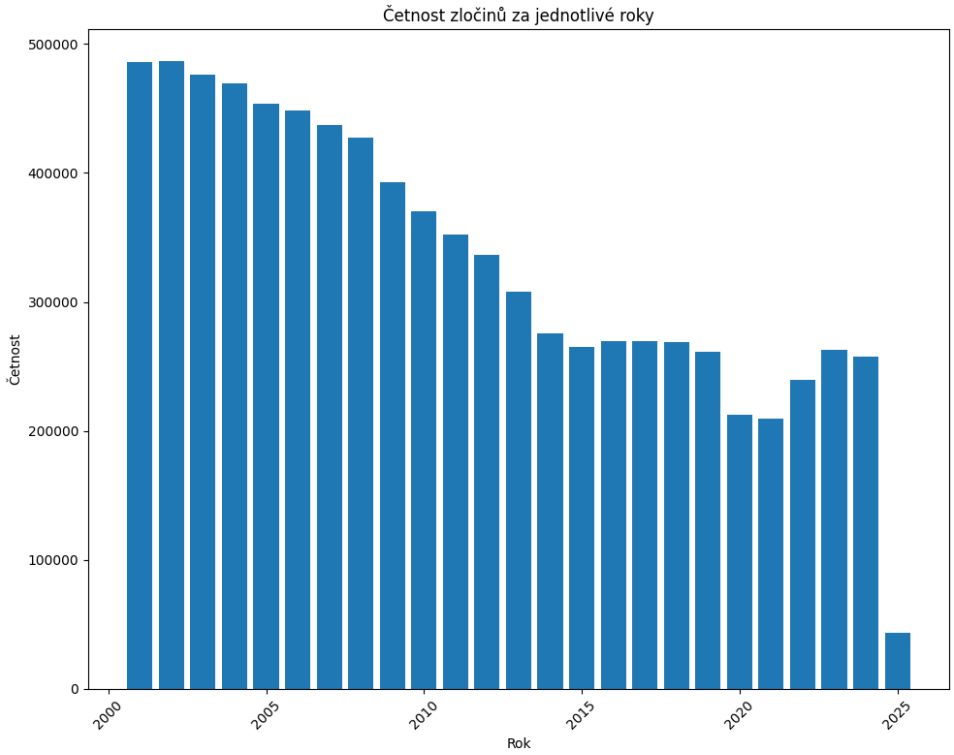
Níže je podrobně rozepsáno 11 vybraných analýz. Navržená aplikace nicméně obsahuje více funkcionalit a možností analýz big dat, které zde ale z důvodu přehlednosti nemohou být všechny uvedeny. Kódovou část vybraných analýz obsahuje Příloha IV.

**Analýza 1** – metoda vytváření sloupcových grafů pomocí knihovny Matplotlib byla zvolena pro vizualizaci četnosti popisů zločinů v datasetu 'crimes'. Vstupními daty jsou informace o popisech zločinů a jejich četnosti, které jsou využity k vytvoření sloupců grafu. Sloupec 'Description' obsahuje informace o popisech zločinů, které jsou reprezentovány jednotlivými sloupci grafu. Výstupem je sloupcový graf zobrazující 10 nejčastějších popisů zločinů a jejich četnosti (viz Obrázek 10). Tato analýza umožňuje snadné vizuální porovnání četnosti různých popisů zločinů a zjištění, které popisy zločinů jsou nejčastější.



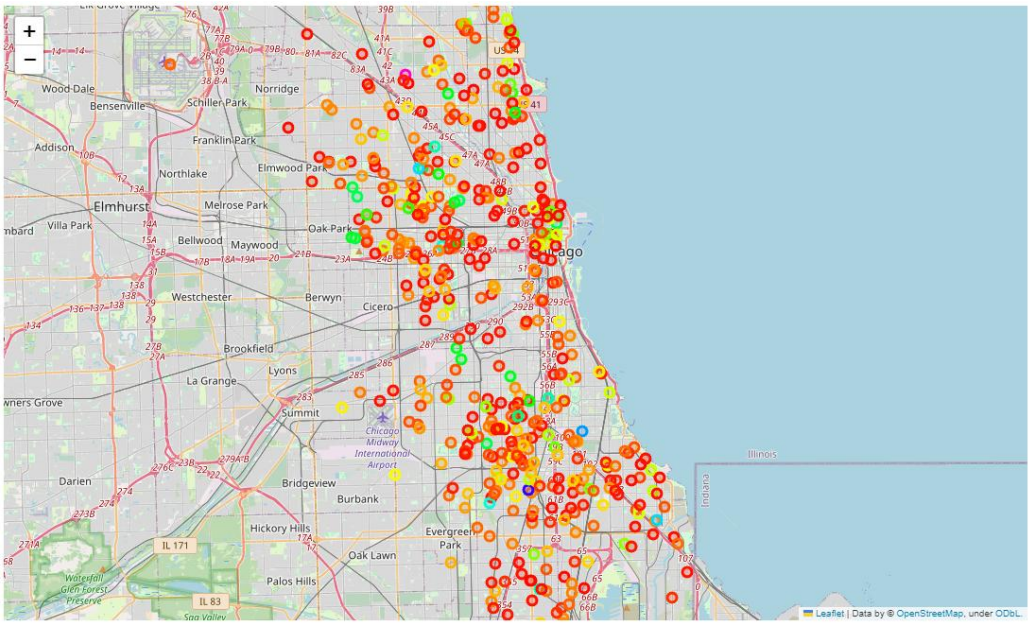

*Obrázek 10:* Sloupcový graf s top 10 nejčastějšími popisy zločinů. Zdroj: vlastní.

**Analýza 2** – metoda vytváření sloupcových grafů pomocí knihovny Matplotlib byla zvolena pro vizualizaci četnosti hodnot v sloupci ‘Year’ v datasetu ‘dates’. Vstupními daty jsou informace o jednotlivých rocích a jejich četnosti. Sloupec ‘Year’ obsahuje informace o roce, ve kterém se události staly, které jsou reprezentovány jednotlivými sloupci grafu. Výstupem je sloupcový graf zobrazující četnost hodnot v sloupci ‘Year’ (viz Obrázek 11). Tato analýza umožňuje snadné vizuální porovnání četnosti událostí v různých rocích.



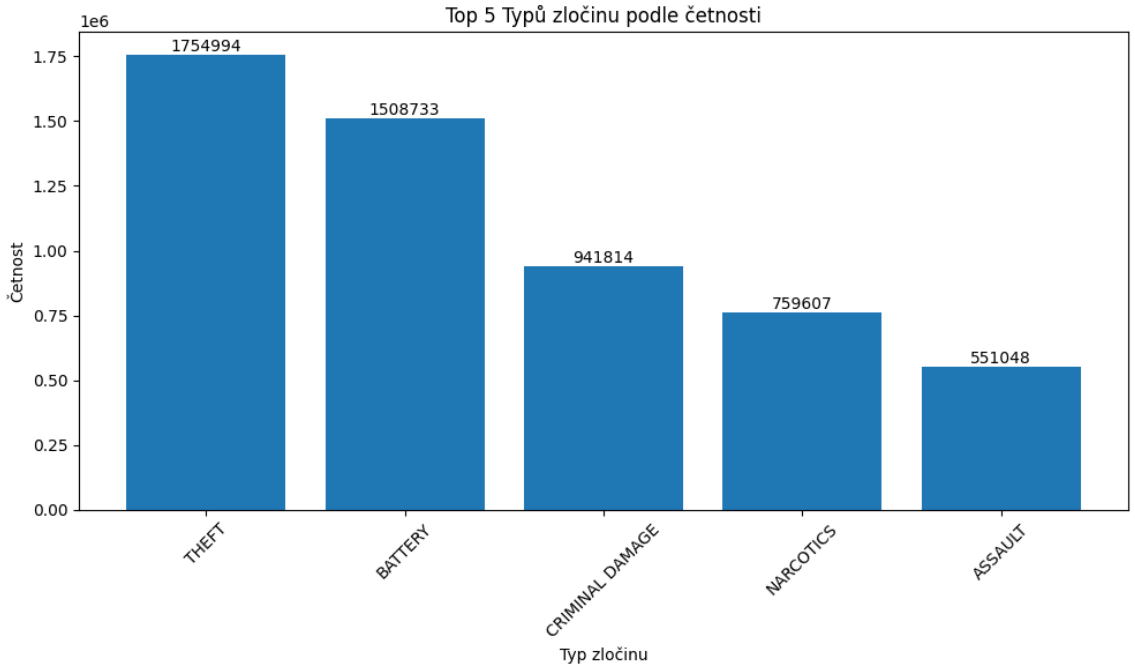

*Obrázek 11:* Sloupcový graf s četností záznamů za jednotlivé roky. Zdroj: vlastní.

**Analýza 3** – metoda zobrazování mapy pomocí knihovny Folium byla zvolena pro vizualizaci geografických dat o zločinech v Chicagu v roce 2008. Vstupními daty jsou informace o popisu zločinu, čísle případu, datu, roce, zeměpisné šířce a délce, které jsou využity k umístění bodů na mapě. Sloupec 'Description' obsahuje informace o popisech zločinů, které jsou reprezentovány barevnými kódy na mapě. Výstupem je interaktivní mapa zobrazující umístění zločinů podle jejich geografických souřadnic a rozdělení podle popisu zločinu (viz Obrázek 12). Tato analýza umožňuje snadné vizuální porovnání rozložení zločinů a zjištění, jestli existuje nějaká závislost mezi jejich geografickým umístěním a popisem zločinu.



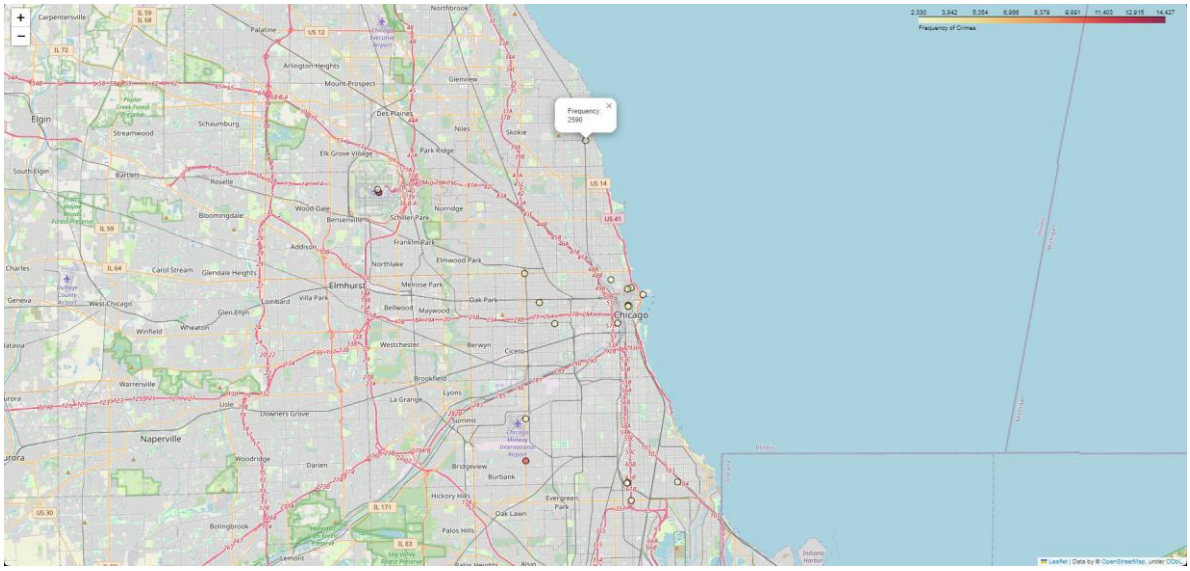

*Obrázek 12:* Interaktivní mapa s 500 náhodnými zločiny. Zdroj: vlastní.

**Analýza 4** – metoda vytváření sloupcových grafů pomocí knihovny Matplotlib byla zvolena pro vizualizaci četnosti hodnot v sloupci ‘Primary_Type’ v datasetu ‘crimes’. Vstupními daty jsou informace o primárním typu zločinu a jejich četnosti, které jsou využity k vytvoření sloupců grafu. Sloupec ‘Primary_Type’ obsahuje informace o primárních typech zločinů, které jsou reprezentovány jednotlivými sloupci grafu. Výstupem je sloupcový graf zobrazující četnost pěti nejčastějších primárních typů zločinů (viz Obrázek 13). Tato analýza umožňuje snadné vizuální porovnání četnosti různých primárních typů zločinů v datasetu.



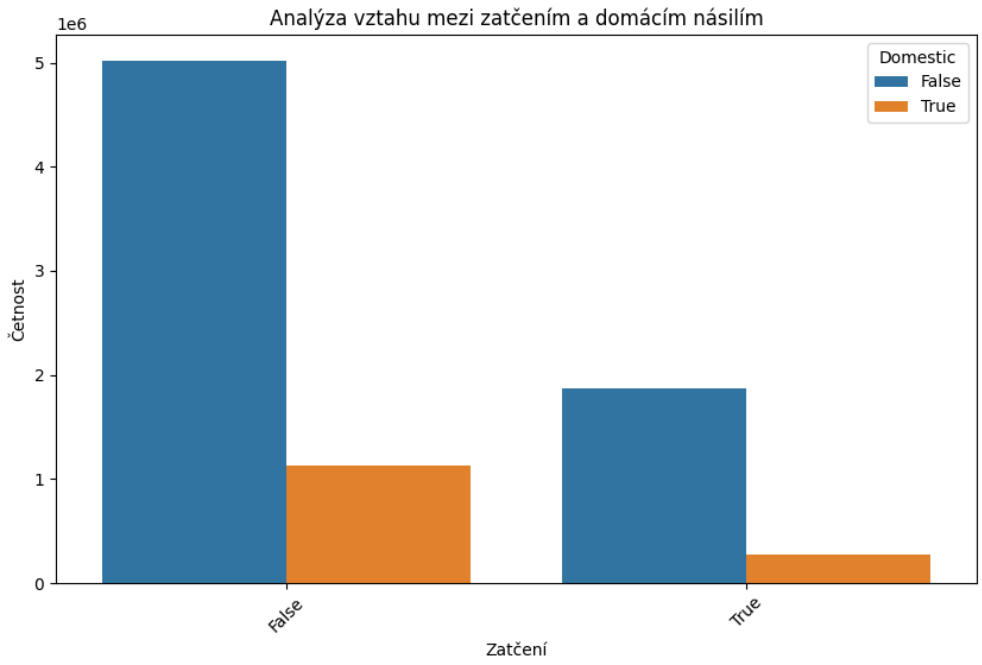

*Obrázek 13:* Sloupcový graf četnosti top 5 typů zločinu sloupce Primary_Type. Zdroj: vlastní.

**Analýza 5** – metoda zobrazování mapy pomocí knihovny Folium byla zvolena pro vizualizaci geografických dat o zločinech v Chicagu. Vstupními daty jsou informace o zeměpisné šířce a délce a jejich četnosti, které jsou využity k umístění bodů na mapě. Sloupce ‘Longitude’ a ‘Latitude’ obsahují informace o geografických souřadnicích, které jsou reprezentovány jednotlivými body na mapě. Výstupem je interaktivní mapa zobrazující umístění zločinů podle jejich geografických souřadnic a rozdělení podle četnosti zločinů (viz Obrázek 14). Tato analýza umožňuje snadné vizuální porovnání rozložení zločinů a zjištění, jestli existuje nějaká závislost mezi jejich geografickým umístěním a četností zločinů.



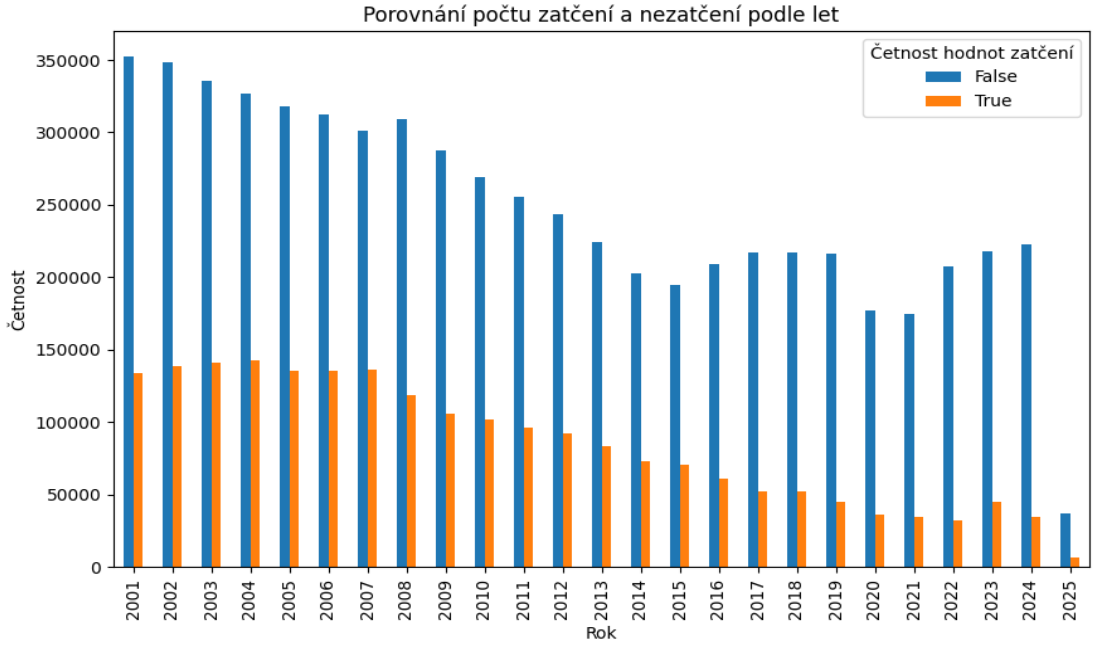

*Obrázek 14:* Interaktivní mapa s 20 body nejčastějších zločinů. Zdroj: vlastní.

**Analýza 6** – metoda vytváření sloupcových grafů pomocí knihovny Seaborn byla zvolena pro vizualizaci vztahu mezi zatčením a domácím násilím v datasetu ‘crimes’. Vstupními daty jsou informace o tom, zda byl zločin spojen se zatčením (‘Arrest’) a zda byl spojen s domácím násilím (‘Domestic’). Tyto informace jsou využity k vytvoření sloupcového grafu. Sloupce ‘Arrest’ a ‘Domestic’ obsahují relevantní informace, které jsou reprezentovány jednotlivými sloupci grafu. Výstupem je sloupcový graf zobrazující četnost zločinů s ohledem na to, zda bylo provedeno zatčení a zda bylo zločinem domácího násilí (viz Obrázek 15). Tato analýza umožňuje snadné vizuální porovnání četnosti zločinů spojených se zatčením a domácím násilím v datasetu.



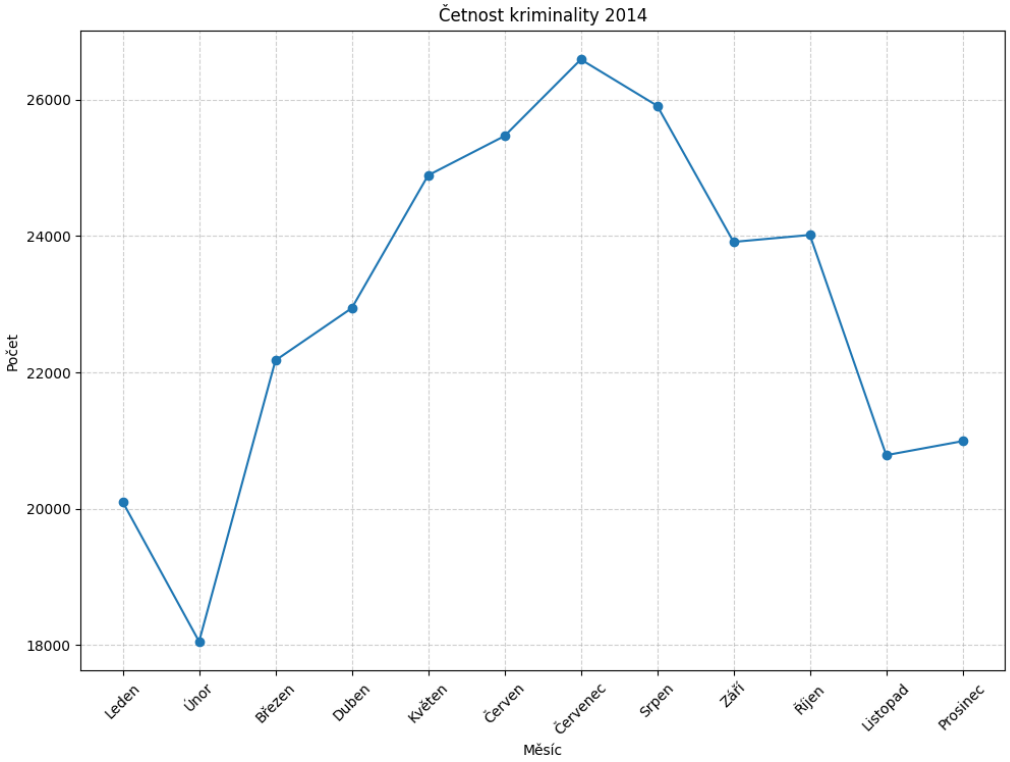

*Obrázek 15:* Sloupcový graf četnosti zatčení vs. domácího násilí. Zdroj: vlastní.

**Analýza 7** – metoda vytváření sloupcových grafů pomocí knihovny Matplotlib byla zvolena pro vizualizaci četnosti zatčení podle roku v datasetu ‘crimes’. Vstupními daty jsou informace o tom, zda bylo provedeno zatčení (‘True’ nebo ‘False’), a rok, ve kterém se zločin stal. Sloupce ‘Arrest’ a ‘Year’ obsahují informace o tom, zda bylo při zločinu provedeno zatčení a rok, ve kterém se zločin stal, které jsou reprezentovány jednotlivými sloupci grafu. Výstupem je sloupcový graf zobrazující četnost zatčení podle roku (viz Obrázek 16). Tato analýza umožňuje snadné vizuální porovnání četnosti zatčení v různých letech.



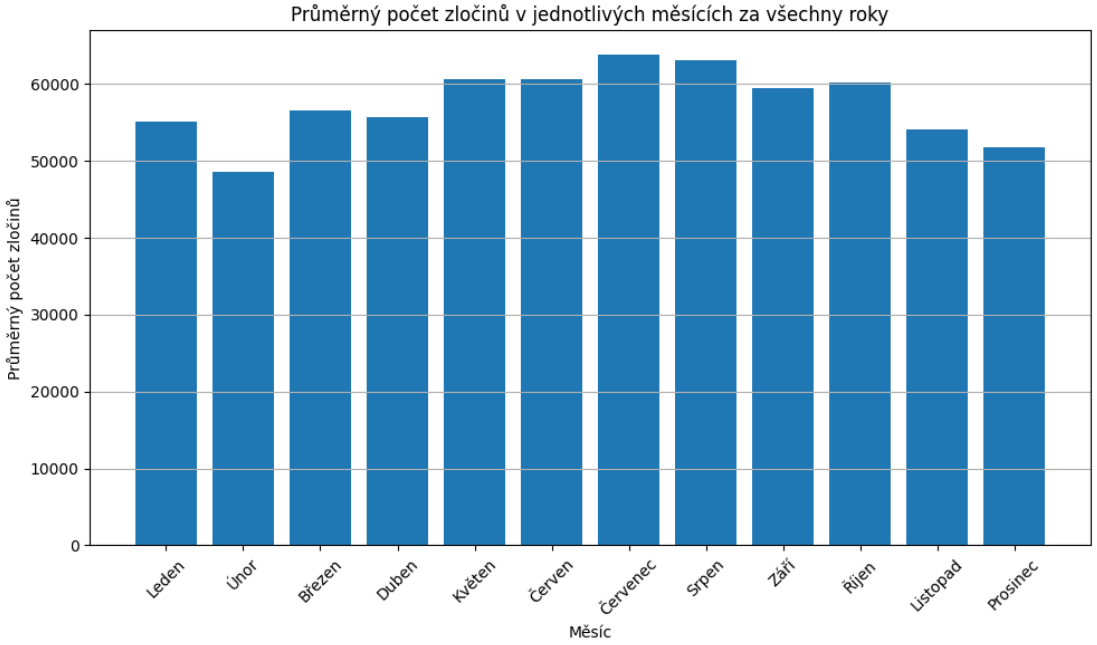

*Obrázek 16:* Sloupcový graf četnosti zatčení a nezatčení za jednotlivé roky. Zdroj: vlastní.

**Analýza 8** – následující metoda vytváří časovou křivku, která zobrazuje četnost zločinů podle vybraného měsíce pro vybraný rok. K tomu je použita knihovna Matplotlib pro vytvoření časové křivky a knihovna tkinter pro umožnění uživatelského výběru roku a měsíce. Vstupními daty je datum zločinu, které je uloženo ve sloupci ‘Date’ datasetu ‘dates’. Toto datum je převedeno na datetime objekt a z něj jsou extrahovány informace o roce a měsíci. Poté je vypočítán počet záznamů pro vybraný den. Výstupem je časová křivka, která zobrazuje četnost zločinů podle měsíce (viz Obrázek 17). Tato analýza umožňuje snadné vizuální porovnání četnosti zločinů v různých měsících.



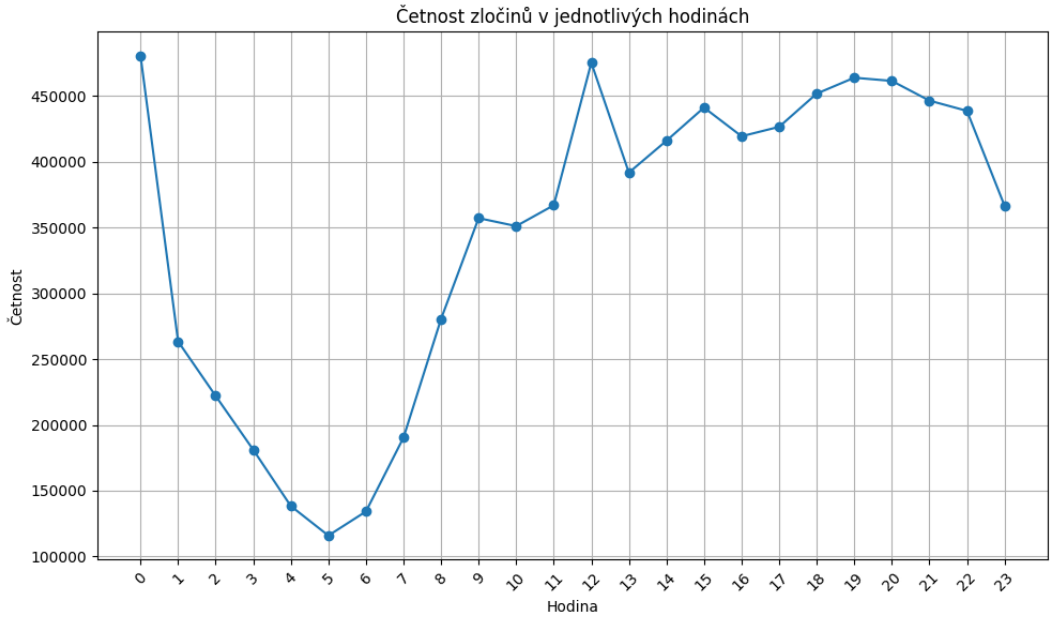

*Obrázek 17:* Časová křivka s četností kriminalit za jednotlivé měsíce v roce. Zdroj: vlastní.

**Analýza 9** – tato metoda vytváří graf, který zobrazuje průměrný počet zločinů v jednotlivých měsících napříč všemi roky dostupnými v datasetu. K vytvoření grafu je použita knihovna Matplotlib. Vstupními daty je datum zločinu, které je uloženo ve sloupci ‘Date’ datasetu ‘dates’. Toto datum je převedeno na datetime objekt a z něj jsou extrahovány informace o měsíci. Poté je vypočítán průměrný počet záznamů pro každý měsíc. Výstupem je graf, který zobrazuje průměrnou četnost zločinů pro každý měsíc (viz Obrázek 18). Tato analýza umožňuje zhodnotit sezónní trendy v kriminalitě během roku.



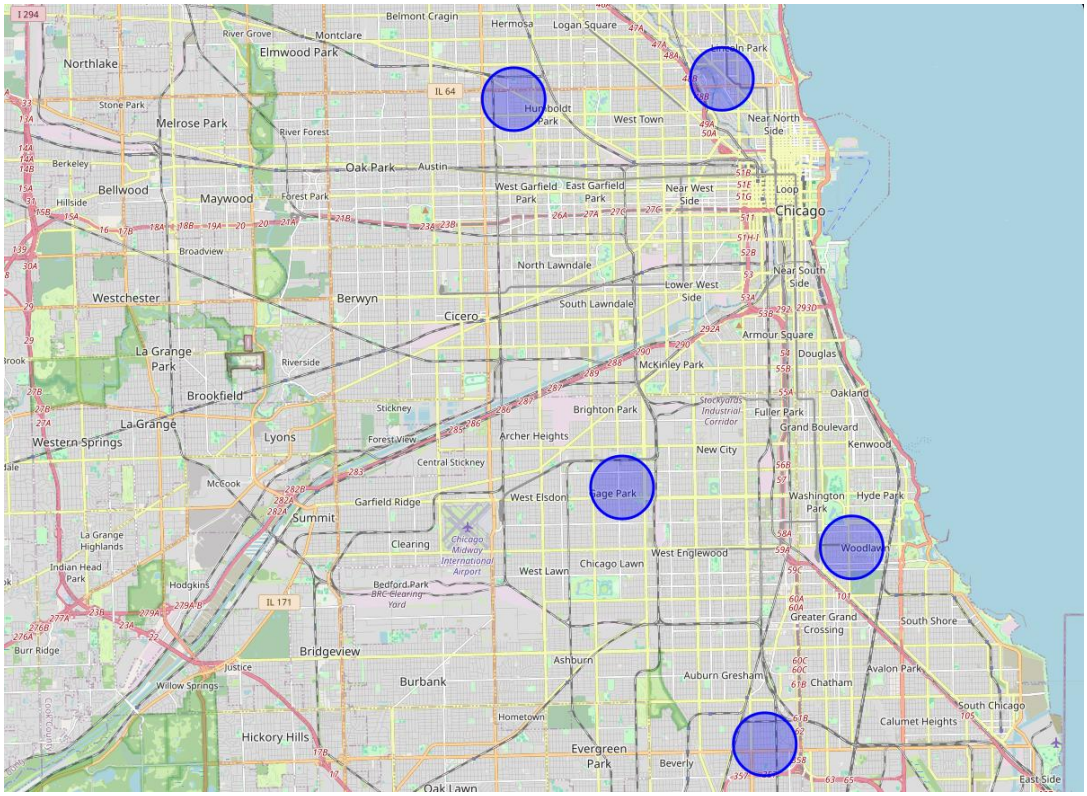

*Obrázek 18:* Sloupcový graf s průměrným počtem zločinů podle měsíců. Zdroj: vlastní.

**Analýza 10** – tato metoda vytváří graf, který zobrazuje četnost zločinů v jednotlivých hodinách během dne. Pro vytvoření grafu je použita knihovna Matplotlib. Vstupními daty je datum a čas zločinu, které jsou uloženy ve sloupci ‘Date’ datasetu ‘dates’. Tento údaj je převeden na datetime objekt a z něj jsou extrahovány informace o hodině. Poté je vypočítán počet záznamů pro každou hodinu. Výstupem je graf, který zobrazuje četnost zločinů pro každou hodinu dne (viz Obrázek 19). Tato analýza umožňuje analyzovat, ve které části dne dochází ke zvýšené kriminalitě.




*Obrázek 19:* Časová křivka s četností kriminalit za jednotlivé hodiny. Zdroj: vlastní.

**Analýza 11** – metoda shlukové analýzy pomocí algoritmu K-means byla zvolena pro identifikaci geografických oblastí s vyšší koncentrací zločinů v Chicagu. Vstupními daty jsou souřadnice zeměpisné šířky a délky, které jsou nejprve očištěny o extrémně odlehlé hodnoty a převedeny do vektorizovaného formátu. Algoritmus K-means rozdělí data do předem definovaného počtu shluků na základě podobnosti jejich polohy. Kvalita modelu je hodnocena pomocí metrik Silhouette Score a Inertia, které poskytují informace o hustotě a vzdálenosti bodů od centroidů shluků. Výstupem analýzy je vizualizace v podobě interaktivní mapy s vyznačenými centroidy shluků, což umožňuje snadnou interpretaci prostorového rozložení kriminality a identifikaci oblastí s vyšším rizikem (viz Obrázek 20).




*Obrázek 20:* Geografické shlukování kriminality. Zdroj: vlastní.

### 4.4.2 Vytvoření testovacího návrhu

V této části je popis, jak by mohl vypadat testovací návrh pro jednotlivé modely, které jsou součástí aplikace. Kódovou část testovacího návrhu obsahuje Příloha V.

**Analýza 1** – nejprve by měla být potvrzena správnost načtení dat pomocí SQL dotazu a ověřeno, že obsahují všechny požadované sloupce, zejména sloupec Description, nezbytný pro tvorbu sloupcového grafu. Dále má být zajištěno, že četnost hodnot v tomto sloupci byla správně spočítána a data byla správně seřazena podle četnosti. Počet vybraných popisů by měl

být omezen na 10. Při tvorbě grafu je třeba zajistit, že je graf bez chyb, popisy zločinů jsou správně přiřazeny jednotlivým sloupcům a titulek grafu je nastaven na „Top 10 nejčastějších popisů zločinů“. Nakonec by mělo být ověřeno, že graf je zobrazen bez problémů.

**Analýza 2** – správnost načtení dat pomocí SQL dotazu a přítomnost sloupce Year, potřebného pro vytvoření sloupcového grafu, má být zkontrolována. Počet hodnot v tomto sloupci má být správně vypočítán. Následně má být sloupcový graf zobrazen bez chyb, přičemž roky musí být správně přiřazeny k osám. Titulek grafu „Četnost hodnot podle roku“ má být nastaven správně a graf by měl být viditelný bez jakýchkoli problémů.

**Analýza 3** – načtení dat má být ověřeno SQL dotazem s potvrzením přítomnosti sloupců Description, Latitude a Longitude, které jsou nezbytné pro tvorbu mapy. Správný počet náhodně vybraných řádků (500) má být zajištěn. Při vytváření mapy je třeba dbát na to, aby byly barvy správně přiřazeny k popisům zločinů, body správně umístěny na mapě a aby se mapa zobrazovala bez problémů.

**Analýza 4** – je třeba ověřit, že data obsahují sloupec Primary_Type, nutný pro vytvoření sloupcového grafu. Po výpočtu četnosti hodnot v tomto sloupci má být graf vytvořen bez chyb a primární typy zločinů musí být správně přiřazeny osám. Titulek „Top 5 typů zločinů podle četnosti“ má být nastaven správně a graf se má zobrazovat bez komplikací.

**Analýza 5** – sloupce Longitude a Latitude, potřebné pro vytvoření mapy, mají být přítomny a četnosti hodnot v těchto sloupcích správně vypočítány. Jednotlivé body na mapě musí být umístěny přesně podle souřadnic. Legenda mapy s názvem „Frequency of Crimes“ má být nastavena správně a zobrazení mapy by nemělo obsahovat chyby.

**Analýza 6** – Data by měla obsahovat sloupce Arrest a Domestic, nezbytné pro analýzu. Po přípravě dat má být sloupcový graf vytvořen bez chyb, přičemž vztah mezi zatčením a domácím násilím má být správně zobrazen. Graf by měl být viditelný bez problémů.

**Analýza 7** – sloupce Arrest a Year mají být přítomny, aby bylo možné vytvořit sloupcový graf. Graf má být vytvořen bez chyb a hodnoty True a False správně přiřazeny jednotlivým rokům. Titulek „Frequency of Arrests per Year“ má být nastaven správně a zobrazení grafu má proběhnout bez problémů.

**Analýza 8** – dataset musí obsahovat sloupec Date, který je nutný pro vytvoření časové křivky. Po přípravě dat má být graf zobrazen bez chyb, přičemž jednotlivé dny v měsíci musí odpovídat bodům na křivce. Konečné zobrazení grafu má proběhnout bez komplikací.

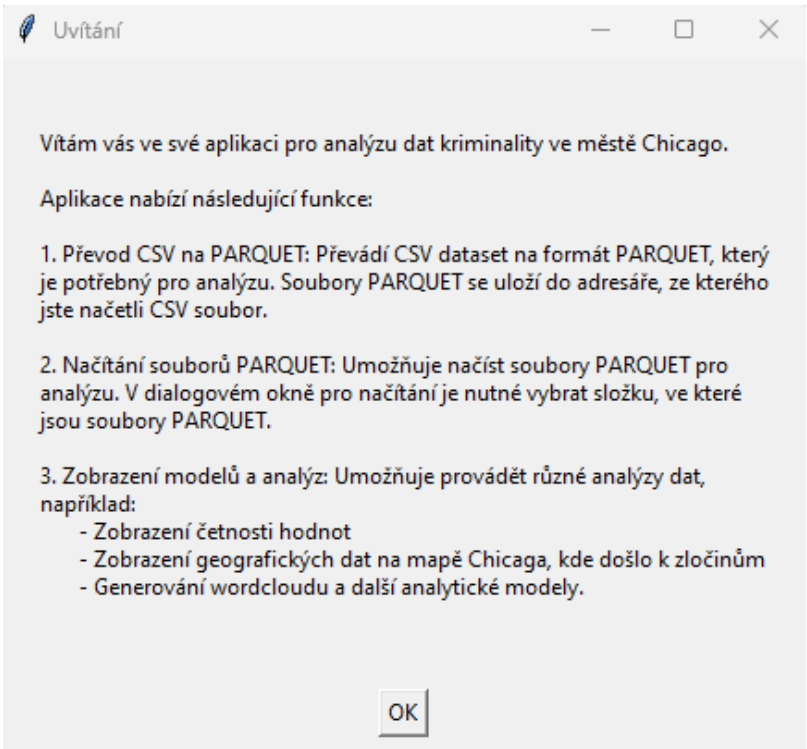
**Analýza 9** – SparkSession má být inicializována a DataFrame dates vytvořen. SQL dotaz na průměrný počet zločinů podle měsíců má vracet správné hodnoty. Počet záznamů a výsledky mají odpovídat očekávání. SparkSession má být na závěr úspěšně uzavřena.

**Analýza 10** – je třeba potvrdit, že SparkSession byla inicializována správně a že dataset obsahuje potřebné časové záznamy. SQL dotaz pro získání četností zločinů podle hodin má vracet správné hodnoty. Graf má být vytvořen, uložen do byte streamu a jeho velikost má být větší než nula, což potvrzuje úspěšnou tvorbu.

**Analýza 11** – dataset musí obsahovat správně naformátované souřadnice Latitude a Longitude, které jsou nutné pro shlukování dat. Po odstranění neplatných a odlehlých hodnot má být DataFrame úspěšně převeden na PySpark formát a připraven pomocí VectorAssembler. Model K-means musí provést shlukování do skupin, přičemž výpočet metrik Silhouette Score a Inertia má proběhnout bez chyb. Výsledné shluky musí být správně přiřazeny datovým bodům. Vizualizace výsledků má být vytvořena a interaktivní mapa být uložena bez komplikací.

### 4.4.3 Uživatelské rozhraní a funkce aplikace

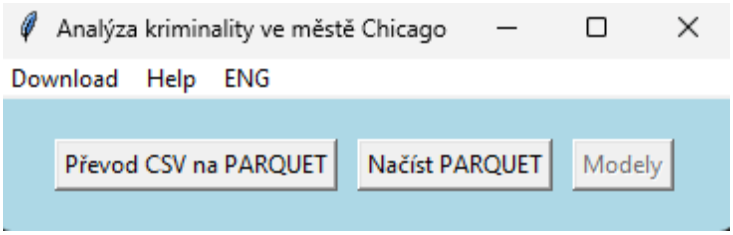
Obrázek 21 zachycuje úvodní grafické rozhraní aplikace, kde je uvítání do aplikace a následný popis, co vše aplikace umožňuje. Toto okno se zobrazí po spuštění aplikace.




*Obrázek 21:* Úvodní grafické rozhraní aplikace. Zdroj: vlastní.

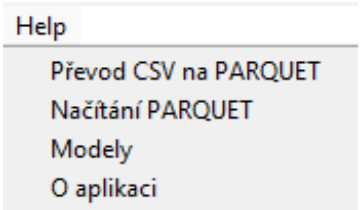
Obrázek 22 zobrazuje okno, které se otevře po kliknutí na tlačítko OK úvodní obrazovky. Jsou zde tři tlačítka, kde první „*Převod CSV na PARQUET*“ převede CSV soubory na soubory PARQUET, které jsou mnohem lepší pro práci s big daty. Po kliknutí na tlačítko se zobrazí průzkumník pro vybrání CSV souboru, který se má převést do nového formátu. Po vybrání CSV se vytvoří nová složka v adresáři, kde je načítaný CSV soubor. Nová složka je nazvána „PARQUET“.

Dalším tlačítkem je „*Načíst PARQUET*“, které slouží pro načtení nového zdroje dat. Je potřeba vybrat celou složku, kde jsou všechny podsložky. V případě, že se načte pouze jedna podsložka, tak nebudou umožněny všechny analýzy, jelikož se načte pouze jedna tabulka. Po úspěšném načtení dat se zobrazí informativní okno, že se data úspěšně načetla. Třetím tlačítkem je tlačítko „*Modely*“. Toto tlačítko slouží pro provádění jednotlivých analýz. V případě, že data nejsou načtena, tak nebude umožněno přejít na další okno po kliknutí na tlačítko „*Modely*“. Aplikace nabízí dvě jazykové varianty: anglickou a českou. Anglická verze je preferována z důvodu jejího globálního použití a datasetu, který je v angličtině. Česká verze byla vytvořena, aby odpovídala požadavkům na jazyk diplomové práce.

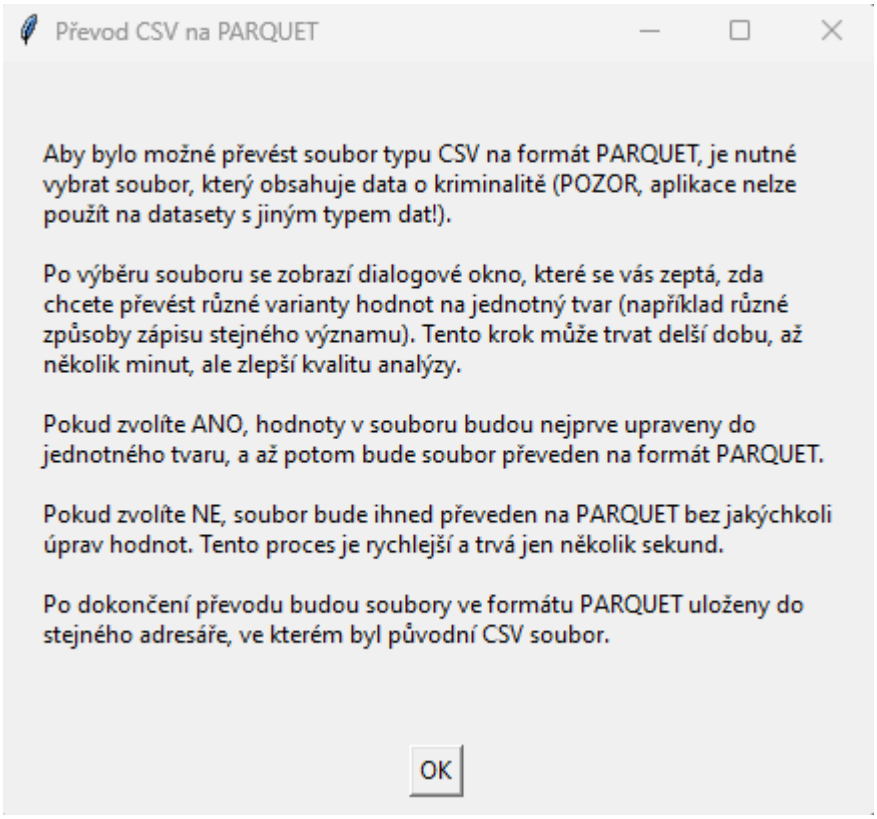



*Obrázek 22:* Okno pro provedení různých akcí. Zdroj: vlastní.

V aplikaci je také možnost v případě problému kliknout na menu „*Help*“ a z nabídky se podívat na nápovědu, kterou v dané chvíli uživatel potřebuje (viz Obrázek 23). Po klepnutí se zobrazí okno s informacemi, které uživatel potřebuje k ovládání aplikace. V okně jsou také pokyny, jak například správně převést soubor a co se stane (viz Obrázek 24).

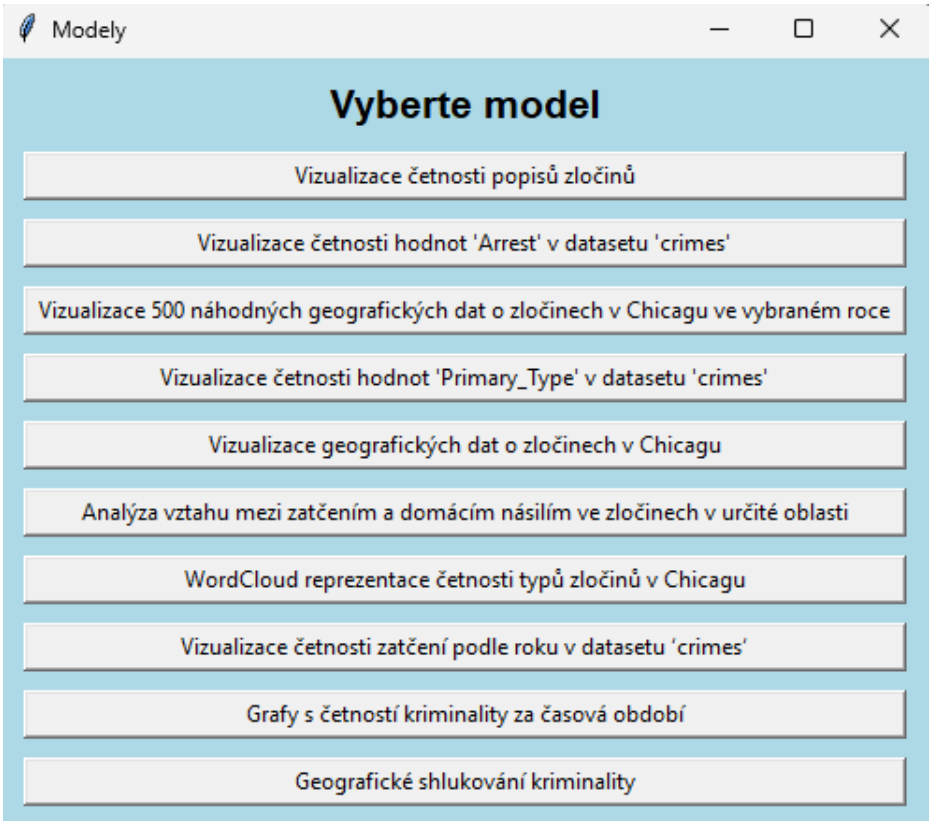



*Obrázek 23:* Nabídka menu Help. Zdroj: vlastní.



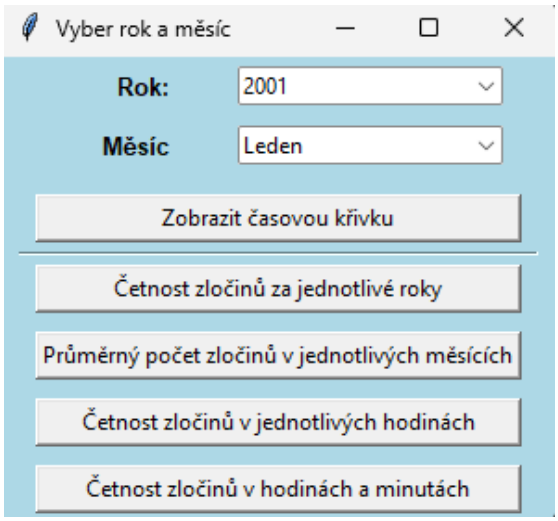

*Obrázek 24:* Nápověda pro Převod CSV na PARQUET. Zdroj: vlastní.

Obrázek 25 zobrazuje okno, ve kterém se vybírají konkrétní modely pro požadovanou analýzu, která uživatele aktuálně zajímá. Po kliknutí na jakékoli tlačítko se v novém okně zobrazí graf či další dialogové okno pro výběr. Pod tlačítky „*Grafické shlukování kriminality*“ se zobrazí podokno, kde je možnost vybrat počet shluků nebo si zobrazit dendrogram. Tlačítko „*Grafy s četností kriminality za časová období*“ otevře nové okno, ve kterém je uživateli umožněn výběr analýzy, která má být zobrazena. Aplikace poskytuje 15 analýz pro hlubší vhled do dat.




*Obrázek 25:* Výběr modelu pro analýzu. Zdroj: vlastní.

V případě, že se klikne na tlačítko „*Grafy s četností kriminality za časová období*“, tak se uživateli otevře nové dialogové okno (viz Obrázek 26), kde uživatel vybírá, zda chce zobrazit časovou křivku podle jím vybraného časového období. Pokud se nepožaduje zobrazit časovou křivku v daném časovém období, tak jsou k dispozici ještě další tlačítka, která zobrazí časovou křivku podle názvu tlačítka. Místo jednoho měsíce je možnost vybrat „*Vše*“, kdy se zobrazí časová křivka s četností za jednotlivé měsíce ve vybraném roce.




*Obrázek 26:* Výběr zobrazení časové křivky. Zdroj: vlastní.

## 4.5 Vyhodnocení výsledků

Kapitola se zaměřuje na analýzu výsledků získaných pomocí vytvořených modelů. Jsou zde popsány metody hodnocení výkonnosti modelů, jejich srovnání s definovanými cíli a také identifikace potenciálních zlepšení. Je zmíněn i návrh dalších kroků pro budoucí analýzy.

### 4.5.1 Zhodnocení výsledků

**Analýza 1** – tato analýza nabízí podrobný pohled na nejběžnější typy zločinů v datasetu a může odhalit, zda některé z nich vykazují výrazně vyšší frekvenci. Tyto informace jsou cenné pro kriminální analytiky a vědce, kteří se zaměřují na pochopení vzorců zločinnosti, a mohou rovněž pomoci orgánům veřejné moci při formulování efektivních strategií pro prevenci a potírání kriminality.

**Analýza 2** – tato analýza nabízí užitečný vhled do změn četnosti zločinů v průběhu let. To by mohlo napomoci k lepšímu porozumění trendům ve zločinnosti a být užitečné pro kriminální analytiky nebo vědce, kteří sledují vývoj zločinnosti v čase. Tato informace by také mohla pomoci orgánům veřejné moci při plánování strategií pro boj proti zločinu.

**Analýza 3** – tato analýza nabízí hlubší vhled do geografické distribuce zločinů v Chicagu v roce 2008. Může například odhalit, zda existují určité oblasti města, které jsou náchylné k určitým typům zločinů. Tato informace může být cenná pro politiky při rozhodování o rozdělení zdrojů, jako je policejní přítomnost nebo sociální služby, a může pomoci identifikovat oblasti, které by mohly profitovat z cílených preventivních opatření.

**Analýza 4** – tato analýza poskytuje hlubší vhled do typů zločinů, které se nejčastěji vyskytují v datasetu. Může například odhalit, zda existují určité typy zločinů, které jsou obzvláště časté. Tato informace může být užitečná pro kriminální analytiky nebo vědce, kteří chtějí porozumět nejčastějším typům zločinů. Právní orgány by mohly využít tyto poznatky při plánování strategií pro boj proti zločinu.

**Analýza 5** – tato analýza poskytuje pohled do geografického rozložení zločinů v Chicagu. Může například odhalit, zda existují určité oblasti města, které jsou náchylné k určitým typům zločinů. Tato informace může být cenná pro politiky při rozhodování o rozdělení zdrojů, jako je policejní přítomnost nebo sociální služby, a může pomoci identifikovat oblasti, které by mohly profitovat z cílených preventivních opatření.

**Analýza 6** – tato analýza poskytuje rychlý přehled o nejčastějších typech zločinů v daném datasetu. To může být užitečné pro kriminální analytiky nebo vědce, kteří zkoumají zločinnost v určitém městě nebo regionu. Právní orgány by mohly využít tyto poznatky při plánování strategií pro boj proti zločinu.

**Analýza 7** – tato analýza poskytuje užitečný vhled do změn četnosti zatčení v průběhu let. To by mohlo napomoci k lepšímu pochopení účinnosti právních opatření a jejich dopadu na zločinnost. Například pokud by se ukázalo, že četnost zatčení klesá, mohlo by to naznačovat potřebu zvýšit policejní přítomnost nebo zlepšit strategie pro vyšetřování zločinů.

**Analýza 8** – tento výstup ilustruje změny v četnosti zločinů během roku 2014. Nižší kriminalita v zimních měsících naznačuje sezónnost, což by mohlo souviset s nižší mírou aktivity v zimních měsících. Tento vzor by mohl pomoci při plánování policejních operací nebo preventivních opatření zaměřených na konkrétní období v roce.

**Analýza 9** – tato analýza v podstatě potvrzuje sezónní trendy z předchozích grafů. Ukazuje průměrnou četnost zločinů v jednotlivých měsících za všechny roky, což poskytuje konzistentnější pohled na sezónnost kriminality. Pokles v zimních měsících je zde jasně viditelný, což může znamenat, že zločiny jsou častější během teplejších měsíců. Tento souhrnný přístup je užitečný pro odhalení dlouhodobých vzorců v datech.

**Analýza 10** – tato analýza ukazuje, že počet zločinů je v ranních hodinách nižší, což je logické, protože během noci je méně aktivit. Nicméně, podezřele vysoké hodnoty v 0:00 a 12:00 naznačují možnou chybu v datech. Je pravděpodobné, že tam, kde nebyl evidován přesný čas zločinu, systém automaticky přiřadil 0:00 nebo 12:00 jako výchozí hodnotu. Tento jev je běžný v systémech, které pracují s neúplnými daty.

**Analýza 11** – v této analýze byla identifikována nejvíce zasažená místa kriminalitou v Chicagu. Pomocí metody K-means bylo město rozděleno do pěti hlavních shluků, které označují oblasti s vyšší koncentrací zločinů. Počet shluků byl určen na základě Wardovy metody hierarchického shlukování, přičemž z dendrogramu bylo možné odhadnout možný počet shluků. Silhouette Score s hodnotou 0,572 ukazuje, že shluky byly vytvořeny poměrně dobře a kriminalita je v některých lokalitách výrazně koncentrována. Inertia s hodnotou 11 099 potvrzuje, že mezi jednotlivými shluky existují znatelné rozdíly v hustotě výskytu zločinů. Na mapě bylo vizualizováno pět hlavních oblastí s nejvyšší četností kriminality. Výsledky této analýzy mohou být využity pro efektivnější plánování policejních hlídek a preventivních opatření v nejrizikovějších částech města.

### 4.5.2 Zhodnocení fází a kroků

Analýza a vizualizace dat o kriminalitě je zajímavým a podnětným procesem, který zahrnuje různé metody vizualizace a statistické analýzy. Každá fáze analýzy byla zaměřena na konkrétní aspekty big datasetu a přinesla hodnotné informace, které byly v souladu s cíli práce. V rámci analýzy byly použity různé metody, např. sloupcové grafy nebo mapové výstupy, přičemž každá z těchto metod poskytla specifické vhledy do struktury a dynamiky kriminality.

Během realizace jednotlivých fází bylo nutné vyřešit několik překážek, zejména s problémem chybějících dat nebo nepřesností ve vstupním datasetu. Tyto problémy do jisté míry ovlivnily výsledky některých vizualizací, například výskyt chybějících hodnot v grafech znázorňujících četnost popisů zločinů. Nejvíce času bylo věnováno přípravě a kontrole kvality dat, aby bylo zajištěno, že vstupní data jsou konzistentní a spolehlivá před jejich následným zpracováním.

Jednou z klíčových analýz byla časová křivka četnosti zločinů podle měsíců v roce 2014, která ukázala, že zimní měsíce zaznamenávají nižší kriminalitu než letní. Tento sezónní jev může být spojen s klimatickými podmínkami nebo zvýšenou policejní aktivitou v určitých obdobích. Další analýzou byla četnost zločinů za jednotlivé roky, jež prokázala celkový pokles zločinů, což naznačuje, že implementované bezpečnostní opatření mohla mít pozitivní efekt na kriminalitu v daném městě. Následující analýza, průměrný počet zločinů v jednotlivých měsících za všechny roky, potvrdila sezónní trend zjištěný v předchozích grafech a prokázala, že sezónní výkyvy nejsou výjimečným jevem pro konkrétní rok, ale stabilní vzorec.

Důležitou součástí analýzy byly i analýzy četnosti zločinů v jednotlivých hodinách a minutách. Tyto ukázaly, že zločiny jsou méně časté v ranních hodinách. Nicméně se v těchto analýzách objevily možné chyby dat, kde četnost kolem půlnoci a poledne vykazovala podezřelý vrchol. Tento problém mohl být způsoben systémovými zaokrouhleními času nebo chybějícími informacemi ve zdrojovém datasetu. Přesto tato analýza ukázala významné problémy spojené s kvalitou dat a zdůraznila, že při práci s časovými daty je nutné být velmi opatrný.

Většina provedených analýz splnila očekávání a poskytla užitečné informace. I přesto, že kvalita dat v některých případech zkomplikovala analýzu, celkový výsledek lze považovat za přínosný. Celkově analýza také podtrhla důležitost kvalitních a přesných dat pro správné interpretace.

Vytvoření aplikace pro analýzu big dat tak představuje interaktivní nástroj, pomocí kterého uživatelé mohou analyzovat vybraný dataset. Kód pro tvorbu aplikace je otevřený a k dispozici dalším zájemcům, kteří ho tak mohou upravit pro analýzu jiných big datasetů.

### 4.5.3 Určení dalších kroků

Na základě přechozích kroků lze tedy definovat další postupy pro přesnější analýzy. Prvním je doplnění chybějících dat, když identifikace a doplnění chybějících nebo neúplných dat v datasetu může vylepšit přesnost analýz. To by mohlo zahrnovat vyhledání chybějících informací nebo nahrazení chybějících hodnot vhodnými postupy. Tento krok by mohl také zahrnovat kontrolu dat za účelem identifikace a opravy chyb, jako jsou nesprávné formáty nebo nesprávné hodnoty. Důkladnější kontrola dat a odstranění nesprávných záznamů zaručí větší spolehlivost výsledků. Toto je však nutné zajistit už při vytváření dat, a tudíž sjednotit postupy tvorby.

Pokud je dataset zastaralý, doporučuje se pravidelně aktualizovat informace o zločinnosti, aby byly analýzy relevantní a aktuální. Pro podrobnější pochopení popisů zločinů, které se nejčastěji vyskytují, lze provést hlubší analýzu jednotlivých popisů a jejich četnosti v různých regionech. To by mohlo zahrnovat použití pokročilých technik textové analýzy nebo strojového učení k identifikaci vzorců nebo trendů v popisech zločinů. Pro strategické plánování je důležité identifikovat oblasti s vysokou mírou zločinnosti a vytvořit strategii pro zlepšení bezpečnosti v těchto oblastech. To by mohlo zahrnovat analýzu geografických dat o zločinnosti a vývoj strategií zaměřených na zlepšení bezpečnosti v oblastech s vysokou mírou zločinnosti.

## 4.6 Využití výsledků

V této části je popsán proces nasazení modelu do praxe. Jsou zde diskutovány plány nasazení aplikace, řízení a správa implementace. Kapitola se věnuje i otázkám udržitelnosti a možnosti rozšíření funkcionality aplikace v budoucnu.

### 4.6.1 Plán nasazení

Analýza četnosti popisů zločinů je široce využívána v analytických odděleních zaměřených na kriminalitu nebo v rámci sociologických výzkumů. Cílem je identifikovat nejčastější typy zločinů podle jejich textových popisů. Ve studii *A Systematic Review of Using Machine Learning and Natural Language Processing in Smart Policing* (Sarzaeim et al., 2023) bylo zdůrazněno, že zpracování přirozeného jazyka může hrát klíčovou roli při analýze popisů

zločinů. Doporučuje se, aby analýzy zahrnovaly i kontextové faktory, například socioekonomické podmínky, které by mohly odhalit hlubší vztahy mezi popisy a typy kriminality. Rovněž by mohlo být začleněno vyhledávání klíčových slov souvisejících s konkrétními zločiny, což by napomohlo predikci rizikových oblastí a umožnilo lepší prevenci.

Analýza četnosti zločinů podle jednotlivých let se využívá pro identifikaci dlouhodobých trendů a časových vzorců. Ve studii bylo zjištěno, že historická data mohou odhalit sezónní a meziroční změny, které ovlivňují kriminalitu (Sarzaeim et al., 2023). Tyto informace se doporučuje využít k porovnání jednotlivých let, což může vést k přesnější identifikaci trendů. Takové porovnání umožňuje plánovat preventivní opatření na základě zjištěných historických vzorců a vývoje kriminality.

Geografická vizualizace zločinů se považuje za důležitý nástroj, který napomáhá identifikovat rizikové oblasti. Tento přístup se často využívá v městském plánování a při tvorbě bezpečnostních strategií. Studie *GIS and Crime Mapping* (Chainey a Ratcliffe, 2013) ukázala, že prostorová data, například hustota obyvatelstva nebo dostupnost veřejných služeb, mohou pomoci odhalit vzorce kriminality. Doporučuje se využití geografických analytických nástrojů spolu s prediktivní analýzou, aby bylo možné lépe plánovat policejní zásahy a efektivně alokovat zdroje v rizikových lokalitách. Geografická vizualizace zločinů v Chicagu je zásadní pro strategické plánování a efektivní využívání policejních zdrojů. Ve studii *Multi-density crime predictor* (Cesario et al., 2024) bylo ukázáno, že kombinace historických dat s údaji o mobilitě obyvatel může výrazně zlepšit predikci rizikových míst. Například vztah mezi pohybem lidí a četností zločinů se doporučuje zohlednit při identifikaci kritických oblastí. Dále se navrhuje zahrnout do vizualizací dynamické prvky, například aktuální data v reálném čase, což by umožnilo pružnější reakce na změny v trendech kriminality.

Při analýze vztahu mezi zatčeními a domácím násilím je zdůrazňována nutnost zahrnout i socioekonomické faktory. Ve studii *Police Responses to Domestic Violence* (Hoyle et al., 2000) bylo uvedeno, že faktory, jako je nezaměstnanost nebo přístup k sociálním službám, mají vliv na výskyt domácího násilí. Doporučuje se, aby analýza zahrnovala širší kontext, což by mohlo pomoci navrhnout účinnější opatření, například prevenci recidivy nebo zlepšení dostupnosti pomoci pro oběti.

Vizualizace časových křivek v rámci jednotlivých měsíců odhaluje sezónní trendy v kriminalitě. Ve studii Corcoran a Zahnow (2022) bylo prokázáno, že teplejší měsíce a delší dny korelují s nárůstem některých typů kriminality, například majetkových a násilných činů.

Doporučuje se zahrnout klimatická data, která by přispěla k přesnějším analýzám sezónních vzorců a souvislostí mezi počasím a kriminalitou. Časové analýzy na úrovni hodin poskytují detailní přehled o kritických obdobích dne. Towers et al. (2018) ve své studii upozornili na zvýšenou kriminalitu během nočních hodin v oblastech s vyšší koncentrací nočních podniků. Doporučuje se propojit data s faktory, jako jsou dny v týdnu nebo místní události, což umožní efektivnější alokaci zdrojů.

Analýza shlukování kriminality pomocí K-means poskytuje nový pohled na koncentraci zločinů v Chicagu. Metoda K-means byla využita k rozdělení města do pěti hlavních shluků, které označují oblasti s nejvyšší četností kriminality. Ve studii *A comparative analysis of selected clustering algorithms for criminal profiling* bylo zdůrazněno, že shlukovací metody mohou pomoci lépe identifikovat klíčové oblasti, kde je nutná zvýšená policejní přítomnost (Adeyiga et al., 2020). Hodnota Silhouette Score (0,572) naznačuje, že vytvořené shluky jsou poměrně dobře oddělené, zatímco hodnota Inertia (11 099) potvrzuje existenci jasných rozdílů v hustotě zločinů mezi jednotlivými oblastmi. Doporučuje se využití těchto shlukovacích metod v kombinaci s prediktivní analýzou, která by mohla pomoci včas identifikovat nové problematické lokality a optimalizovat rozložení policejních sil.

**Celkový cíl nasazení** – prezentované analýzy byly navrženy tak, aby poskytovaly ucelené nástroje podporující strategické rozhodování v oblasti prevence kriminality a bezpečnostního plánování. Tyto analýzy mají sloužit nejen k rychlému přístupu k důležitým informacím, ale také k hlubší analýze, která může odhalit dlouhodobé trendy i skryté vzorce. Význam těchto analýz spočívá rovněž v jejich schopnosti přispět k efektivnějším diskusím mezi odborníky a výzkumníky v oblastech, jako je kriminologie, sociální politika nebo městské plánování.

Bylo identifikováno, že důraz na propojení dat s širšími socioekonomickými faktory, jako jsou demografické změny, kulturní události, svátky nebo klimatické podmínky, může výrazně zlepšit kvalitu analýz. Zvláště v oblasti geografických a časových vizualizací je doporučeno využití dalších kontextových proměnných, například dat o pohybu obyvatel, nezaměstnanosti nebo dostupnosti sociálních služeb. Tyto faktory umožňují lépe pochopit, proč ke kriminalitě dochází v konkrétních oblastech a obdobích, a přispívají tak k navrhování cílenějších a účinnějších bezpečnostních opatření.

Zahrnutí metod detekce anomálií a prediktivní analýzy by mohlo hrát klíčovou roli při předpovědích potenciálně rizikových oblastí nebo období zvýšené kriminality. Například analýza sezónních trendů ukázala, že kriminalita může být ovlivněna faktory, jako je délka dne,

teplota nebo velké události, například festivaly nebo sportovní zápasy. Proto je doporučováno, aby analýzy zahrnovaly klimatická data a makroekonomické indikátory, což může vést k přesnějším predikcím a lepšímu plánování preventivních strategií.

Velký důraz by měl být kladen na kontrolu kvality dat, protože nesrovnalosti nebo chyby v údajích mohou významně ovlivnit přesnost analýz. Jak ukázaly některé studie, využití metod strojového učení k identifikaci a opravě datových nesrovnalostí by mohlo výrazně zvýšit spolehlivost výsledků. Integrace těchto technik by umožnila lepší zpracování velkých datových souborů a vytvoření robustnějších modelů predikce.

Doporučuje se rovněž zahrnutí dynamických prvků, které by umožnily aktualizace dat v reálném čase. Takové řešení by poskytlo rychlejší a flexibilnější reakce na aktuální situaci, například při nárůstu kriminality v určité oblasti. Vizualizace zaměřené na časové vzorce, například analýzy kriminality během jednotlivých hodin nebo dnů, by mohly být dále propojeny s dalšími faktory, jako jsou konkrétní lokální události, pohyb obyvatel během dne nebo dny v týdnu, kdy dochází k nejčastějším trestným činům.

Zvláštní pozornost by měla být věnována analýzám, které kombinují geografická a časová data. Geografické mapy zločinů, například pro města jako Chicago, umožňují identifikaci rizikových lokalit a efektivní alokaci policejních zdrojů. Pokud by tyto analýzy byly rozšířeny o další kontextové faktory, jako jsou socioekonomická data, demografické charakteristiky nebo informace o hustotě obyvatelstva, mohly by být získány ještě podrobnější poznatky.

Výsledné analýzy tak mohou sloužit jako podpora pro lepší strategické plánování, přesnější alokaci policejních sil a optimalizaci preventivních opatření. Výstupy mohou být využity nejen bezpečnostními složkami, ale i dalšími institucemi, jako jsou městské úřady nebo sociální služby, které by mohly tyto poznatky aplikovat při zlepšování kvality života v daných oblastech. Zároveň uživatelé, tzn. zpravidla občané měst, mohou podobné aplikace využívat pro vlastní vyhodnocení přínosů informací, včetně kombinací s jinými zdroji dat.

Co se týká rozšiřitelnosti aplikace, tak je důležité sledovat změny v datové sadě, např. přes API v reálném čase, kdy je daná datová sada na portálu aktualizovaná. Pokud začnou být evidovány nové informace (atributy), případně budou nově zveřejněny ty, které jsou z různých důvodů chráněné, např. po anonymizaci dat, tak je vhodné uvažovat o přizpůsobení aplikace, aby ta reflektovala tyto změny.

### 4.6.2 Řízení a správa plánu

Pro úspěšnou realizaci plánu je považováno za klíčové identifikovat všechny nezbytné zdroje. Mezi tyto zdroje patří nejen lidské kapacity, jako datoví analytici, projektoví manažeři nebo IT specialisté, ale také technické prostředky, například výpočetní výkon a softwarové nástroje pro zpracování dat a vývoj modelů. Tento přístup odpovídá myšlenkám uvedeným v práci Davenport a Harris (2017), kde se zdůrazňuje, že pro efektivní využití datových projektů je nutné zajistit kvalitní analytické schopnosti spolu se správným přidělením kapacit. Již na začátku projektu by měly být zajištěny dostatečné zdroje, aby se maximalizoval přínos a minimalizovala rizika. Právě díky identifikaci zdrojů v rané fázi lze lépe plánovat čas, rozpočet i samotnou realizaci strategie. Náklady na jednotlivé kroky plánu, jako je získávání a příprava dat, vývoj modelů, testování a prezentace výsledků, by měly být pečlivě vyčísleny. Doporučuje se zahrnout nejen technologické náklady, ale také výdaje spojené s výpočetním výkonem a používanými nástroji. Transparentní a detailní rozpočet přispívá k lepšímu řízení projektu.

Detailní harmonogram je považován za nezbytný pro efektivní řízení projektu. Nedostatečná pozornost věnovaná plánování termínů může vést k plýtvání časem i zdroji. Tento závěr byl podpořen autory Saltz a Shamshurin (2016) ve studii *Big Data Team Process Methodologies*. V plánu by proto měly být jasně specifikovány termíny pro všechny fáze projektu – od přípravy dat až po prezentaci výsledků. Jasné rozdělení úkolů mezi členy týmu je považováno za klíčové pro zajištění efektivní spolupráce. Ve studii Saltz a Dewar (2019) je doporučeno, aby každý člen týmu měl definované odpovědnosti, a to od sběru a přípravy dat přes vývoj modelů až po jejich implementaci a prezentaci výsledků. Vyjasnění těchto rolí pomáhá předcházet nedorozuměním a podporuje hladký průběh projektu.

Pro úspěšnou realizaci plánu by měla být identifikována a hodnocena potenciální rizika, jako jsou chybějící data, nesprávné výpočty nebo nefunkční modely. K minimalizaci rizik se doporučuje provádět pravidelné kontroly kvality a zálohování dat. Tyto postupy jsou v souladu s poznatky ze studie Kutsch a Hall (2010) *Deliberate Ignorance in Project Risk Management*, kde je zdůrazněna důležitost prevence a spolehlivosti dat. Průběh projektu by měl být pravidelně sledován a jeho jednotlivé kroky vyhodnocovány. Dynamické sledování umožňuje včas odhalit odchylky a provést potřebné změny.

Za klíčovou je považována i efektivní komunikace mezi členy týmu a vedením. Doporučuje se pravidelně pořádat týmové schůzky a poskytovat aktuální informace o průběhu projektu. Tento

přístup, zdůrazněný ve studii Schäfer et al. (2017) *Talk to Your Crowd*, podporuje spolupráci a předávání informací mezi všemi zúčastněnými stranami. Plán by měl být koncipován tak, aby umožňoval flexibilní reakce na nečekané změny nebo výzvy. K tomu lze využít např. úpravy metodiky CRISP-DM s využitím agilních metod (Amirian et al., 2024).

### 4.6.3 Omezení práce

Jedním z hlavních omezení navržené aplikace je její vysoká náročnost na výkon počítače při práci s velkými soubory dat. I když využívá Apache Spark pro efektivní zpracování big dat, její rychlost a plynulost závisí na dostupném hardware. Pokud uživatel nemá dostatečně výkonný počítač nebo server, analýza stovek milionů záznamů může trvat velmi dlouho a výrazně zatížit operační paměť. V takových případech je vhodné využít cloudové služby nebo výkonnější hardware, což ale může znamenat vyšší náklady. Řešením pak mohou být počítačový clustery, u kterých lze pomocí různých distribuovaných technologií (Neto et al., 2022) a dynamické konfigurace hardware (Grzegorowski et al., 2021) dosáhnout snížení nákladů.

Dalším omezením je zaměření aplikace na specifické technologie – konkrétně Python a Apache Spark. Uživatelé, kteří preferují jiné analytické nástroje, jako je R, Microsoft Power BI nebo Tableau, mohou narazit na problémy s kompatibilitou. Přestože lze některé funkce upravit, aplikace není primárně navržena pro práci v jiných prostředích, což může omezit její širší využití. Na druhou stranu je aplikace založena na otevřených technologiích, které jsou většinou kompatibilní s jinými nástroji pro analýzu big dat, např. Apache Hadoop ekosystém.

Důležitým faktorem, který ovlivňuje přesnost výsledků, je kvalita vstupních dat. Veřejně dostupná open data mohou obsahovat chyby, být neúplná nebo zastaralá. Pokud nejsou správně zpracována a vyčištěna, může dojít ke zkreslení výsledků analýzy. Proto je nutné provádět důkladnou kontrolu dat, což však může být časově náročné.

Aplikace je také navržena pro dávkové zpracování, což znamená, že analyzuje velké množství dat najednou a výsledky poskytuje až po dokončení výpočtů. Není tedy ideální pro situace, kdy je nutné analyzovat data v reálném čase, například při sledování bezpečnostní situace nebo výkyvů na finančních trzích. Pokud by bylo potřeba okamžitě reagovat na nová data, aplikace by musela být upravena tak, aby zvládala streamované zpracování dat.

# ZÁVĚR

Cílem této práce bylo charakterizovat současné přístupy k analýze big dat, porovnat existující nástroje a metody, provést sběr a předzpracování dat, analyzovat data, navrhnout modely, a nakonec implementovat celé řešení prostřednictvím aplikace v Pythonu. Pro jeho splnění byl sestaven metodický postup vedoucí k vytvoření aplikace, viz kapitola 3, který zahrnoval systematickou literární rešerši, popis postupu tvorby aplikace a následně specifikaci celého řešení aplikace podle metodiky CRISP-DM.

V prvních dvou kapitolách byly zpracovány zdroje nalezené pomocí systematické literární rešerše, které pomohly propojit teoretické pozadí data science a big dat s požadavky na jejich analýzu. Poté byly porovnány nástroje, platformy a metody pro big data na základě čehož byly zvoleny vhodné nástroje a platformy pro vytvoření aplikace. Tvorba aplikace je poté popsána v kapitole 3.2 a především v kapitole 4, jako stěžejní kapitoly této práce.

Použití metodiky CRIPS-DM poskytlo komplexní pohled na celý proces vytvoření aplikace, včetně jejích funkcionalit, které jsou ilustrovány na analýze reálného big datasetu. Hlavní přínos aplikace spočívá v kombinaci nástrojů a platforem, které propojením umožňují analýzu big dat – v tomto případě open datasetu, který je volně ke stažení na datovém portálu. Jelikož takovýchto datasetů je na různých datových portálech velké množství, navržená aplikace tak ukazuje cestu, jak tato data analyzovat prostřednictvím uživatelsky přívětivého rozhraní. Další přínosy lze spatřovat v řadě vizuálních výstupů pro analýzu a vývoj kriminality v Chicagu.

**Přínos pro business:** Práce může být velmi užitečná pro různé oblasti businessu, například pro orgány veřejné moci a bezpečnostní agentury, které se snaží pochopit a řešit kriminalitu, nebo pro vědce, kteří analyzují vzorce a trendy v kriminalitě. Také pro politiky, kteří se zaměřují na zlepšení bezpečnosti ve městech. Tato práce jim může poskytnout důležité informace, které jim pomohou lépe pochopit a řešit problémy, se kterými se při práci s big daty setkávají.

**Přínos pro data science:** Práce demonstruje, jak lze využít data science pro analýzu big dat. Ukazuje, jak mohou být využity různé metody a techniky data science, jako je čištění dat, vizualizace dat a modelování, pro získání užitečných poznatků z dat. Tento přístup může sloužit jako inspirace pro další výzkum v oblasti data science. Práce také ukazuje, jak může být data science užitečná v praxi, nejen pro výzkumné účely, ale také pro řešení konkrétních problémů a výzev v oblasti businessu.

# POUŽITÁ LITERATURA


ABADI, Martín, et al. {TensorFlow}: a system for {Large-Scale} machine learning. In: *12th USENIX symposium on operating systems design and implementation (OSDI 16)*. 2016. s. 265-283.

ADEYIGA, J. A.; OLABIYISI, S. O.; OMIDIORA, E. O. A comparative analysis of selected clustering algorithms for criminal profiling. *Nigerian Journal of Technology*, 2020, 39.2: 464-471.

AKTER, Shahriar, et al. How to improve firm performance using big data analytics capability and business strategy alignment?. *International Journal of Production Economics*, 2016, 182: 113-131.

ALI, Abdulalem, et al. Financial fraud detection based on machine learning: a systematic literature review. *Applied Sciences*, 2022, 12.19: 9637.

AMIRIAN, E.; ABDOLLAHZADEH, A.; SULAIMAN, N. Synergizing Hybrid Agile-Scrum and CRISP-DM Approaches in Data Science Project Management. In: *SPE Canadian Energy Technology Conference*. SPE, 2024. s. D021S017R001.

CESARIO, Eugenio; LINDIA, Paolo; VINCI, Andrea. Multi-density crime predictor: an approach to forecast criminal activities in multi-density crime hotspots. *Journal of Big Data*, 2024, 11.1: 75.

CIELEN, Davy; MEYSMAN, Arno D. B.; ALI, Mohamed. *Introducing Data Science: Big data, machine learning, and more, using Python tools*. Shelter Island: Manning, 2016.

CITY OF CHICAGO. Crimes - 2001 to present [Dataset]. Data.gov. [online]. 2025. Dostupné z: https://catalog.data.gov/dataset/crimes-2001-to-present

CORCORAN, Jonathan; ZAHNOW, Renee. Weather and crime: a systematic review of the empirical literature. *Crime Science*, 2022, 11.1: 16.

DAVENPORT, Thomas. *Big data at work: dispelling the myths, uncovering the opportunities*. Harvard Business Review Press, 2014.

DAVENPORT, Thomas; HARRIS, Jeanne. *Competing on analytics: Updated, with a new introduction: The new science of winning*. Harvard Business Press, 2017.

DEAN, Jeffrey; GHEMAWAT, Sanjay. MapReduce: simplified data processing on large clusters. *Communications of the ACM*, 2008, 51.1: 107-113.

DUVVURI, Srinivas; SINGHAL, Bikramaditya. *Spark for Data Science*. Packt Publishing Ltd, 2016.

FAYYAD, Usama; PIATETSKY-SHAPIRO, Gregory; SMYTH, Padhraic. From data mining to knowledge discovery in databases. *AI Magazine*, 1996, 17.3: 37-54.

FLOWERS, Ronald B. *Demographics and criminality: The characteristics of crime in America*. New York: Greenwood Press, 1989.

GANDOMI, Amir; HAIDER, Murtaza. Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management*, 2015, 35.2: 137-144.

GOKMENOGLU, Korhan K.; YILDIZ, Bünyamin Fuat; KAAKEH, Mohamad. Examining the impact of socioeconomic factors on crime rates: A panel study. In: *Annual Conference on Finance and Accounting*. Cham: Springer International Publishing, 2020. s. 409-420.

HASHEM, Ibrahim Abaker Targio, et al. The rise of "big data" on cloud computing: Review and open research issues. *Information Systems*, 2015, 47: 98-115.

HEBABAZE, Salah Eddine, et al. From Micro-benchmarks to Machine Learning: Unveiling the Efficiency and Scalability of Hadoop and Spark. *International Journal of Interactive Mobile Technologies*, 2024, 18.17: 46.

HENDL, Jan. *Big data: věda o datech – základy a aplikace*. Praha: Grada Publishing, 2021.

HOYLE, Carolyn; SANDERS, Andrew. Police response to domestic violence. *British Journal of Criminology*, 2000, 40.1: 14-36.

CHAINEY, Spencer; RATCLIFFE, Jerry. *GIS and crime mapping*. John Wiley & Sons, 2013.

CHAWDA, Rahul Kumar; THAKUR, Ghanshyam. Big data and advanced analytics tools. In: *2016 symposium on colossal data analysis and networking (CDAN)*. IEEE, 2016. s. 1-8.

CHEN, Hsinchun; CHIANG, Roger HL; STOREY, Veda C. Business intelligence and analytics: From big data to big impact. *MIS Quarterly*, 2012, 36.4: 1165-1188.

CHEN, Li; COULIBALY, Lala Aicha. Data Science and Big Data Practice Using Apache Spark and Python. In: *Intelligent Analytics With Advanced Multi-Industry Applications*. IGI Global, 2021. p. 67-95.

GRZEGOROWSKI, Marek, et al. Cost optimization for big data workloads based on dynamic scheduling and cluster-size tuning. *Big Data Research*, 2021, 25: 100203.

JIN, Xiaolong, et al. Significance and challenges of big data research. *Big Data Research*, 2015, 2.2: 59-64.

JOHNSON, Jeff S.; FRIEND, Scott B.; LEE, Hannah S. Big data facilitation, utilization, and monetization: Exploring the 3Vs in a new product development process. *Journal of Product Innovation Management*, 2017, 34.5: 640-658.

KINI, Sampath; PAI, Karthik. Exploring Real-Time Data Processing Using Big Data Frameworks. *Communications on Applied Nonlinear Analysis*, 2024, 31.8: 620-634.

KITCHENHAM, Barbara, et al. Systematic literature reviews in software engineering–a systematic literature review. *Information and Software Technology*, 2009, 51.1: 7-15.

KLUYVER, Thomas, et al. Jupyter Notebooks – a publishing format for reproducible computational workflows. In: *Positioning and Power in Academic Publishing: Players, Agents and Agendas*. IOS press, 2016. s. 87-90.

KUTSCH, Elmar; HALL, Mark. Deliberate ignorance in project risk management. *International Journal of Project Management*, 2010, 28.3: 245-255.

LANDSET, Sara, et al. A survey of open source tools for machine learning with big data in the Hadoop ecosystem. *Journal of Big Data*, 2015, 2: 1-36.

LEVY, Eran. *What is the Parquet file format? Use cases & benefits* [online]. 2023. Dostupné z: https://www.upsolver.com/blog/apache-parquet-why-use

LUTZ, Mark. *Learning python: Powerful object-oriented programming*. O'Reilly Media, Inc., 2013.

MCAFEE, Andrew, et al. Big data: the management revolution. *Harvard Business Review*, 2012, 90.10: 60-68.

MENG, Xiangrui, et al. Mllib: Machine learning in apache spark. *Journal of Machine Learning Research*, 2016, 17.34: 1-7.

MERHI, Mohammad I.; BREGU, Klajdi. Effective and efficient usage of big data analytics in public sector. *Transforming Government: People, Process and Policy*, 2020, 14.4: 605-622.

MCKINNEY, Wes. *Python for data analysis: data wrangling with Pandas, NumPy, and IPython*. Second edition. Sebastopol: O'Reilly, 2017.

MEZZOUDJ, Saliha; KHELIFA, Meriem; SAADNA, Yasmina. A Comparative Study of Parallel Processing, Distributed Storage Techniques, and Technologies: A Survey on Big Data Analytics. *International Journal of Data Science and Analysis*, 2024, 10.5: 86-99.

MUKHOPADHYAY, Ayan, et al. Optimal allocation of police patrol resources using a continuous-time crime model. In: *Decision and Game Theory for Security: 7th International Conference, GameSec 2016*. Springer International Publishing, 2016. s. 139-158.

NATIVIDADE JOERGENSEN, Pedro; ZAGGL, Michael. The role of data science and data analytics for innovation: a literature review. *Journal of Business Analytics*, 2024, 1-17.

NETO, Antônio José Alves; NETO, José Aprígio Carneiro; MORENO, Edward David. The development of a low-cost big data cluster using Apache Hadoop and Raspberry Pi. A complete guide. *Computers and Electrical Engineering*, 2022, 104: 108403.

PERKEL, Jeffrey M. Why Jupyter is data scientists' computational notebook of choice. *Nature*, 2018, 563.7732: 145-147.

PERRY, Walt L., et al. *Predictive policing: The role of crime forecasting in law enforcement operations*. Rand Corporation, 2013.

PROVOST, Foster; FAWCETT, Tom. Data science and its relationship to big data and data-driven decision making. *Big data*, 2013, 1.1: 51-59.

RAGAN-KELLEY, Min, et al. The Jupyter/IPython architecture: a unified view of computational research, from interactive exploration to communication and publication. In: *AGU Fall Meeting Abstracts*. 2014. s. H44D-07.

RAO, T. Ramalingeswara, et al. The big data system, components, tools, and technologies: a survey. *Knowledge and Information Systems*, 2019, 60: 1165-1245.

ROŽANEC, Jože M., et al. Automotive OEM demand forecasting: A comparative study of forecasting algorithms and strategies. *Applied Sciences*, 2021, 11.15: 6787.

SALTZ, Jeffrey S. CRISP-DM for data science: strengths, weaknesses and potential next steps. In: *2021 IEEE International Conference on Big Data (Big Data)*. IEEE, 2021. s. 2337-2344.

SALTZ, Jeffrey S.; DEWAR, Neil. Data science ethical considerations: a systematic literature review and proposed project framework. *Ethics and Information Technology*, 2019, 21: 197-208.

SALTZ, Jeffrey S.; SHAMSHURIN, Ivan. Big data team process methodologies: A literature review and the identification of key factors for a project's success. In: *2016 IEEE International Conference on Big Data (Big Data)*. IEEE, 2016. s. 2872-2879.

SARKER, Iqbal H. Data science and analytics: an overview from data-driven smart computing, decision-making and applications perspective. *SN Computer Science*, 2021, 2.5: 377.

SARZAEIM, Paria, et al. A systematic review of using machine learning and natural language processing in smart policing. *Computers*, 2023, 12.12: 255.

SHIMAOKA, André Massahiro; FERREIRA, Renato Cordeiro; GOLDMAN, Alfredo. The evolution of CRISP-DM for Data Science: Methods, Processes and Frameworks. *SBC Reviews on Computer Science*, 2024, 4.1: 28-43.

SINGH, Dilpreet; REDDY, Chandan K. A survey on platforms for big data analytics. *Journal of Big Data*, 2015, 2: 1-20.

SCHÄFER, Sebastian, et al. Talk to Your Crowd: Principles for Effective Communication in Crowdsourcing A few key principles for communicating with solvers can help contest sponsors maintain and grow their base of participants. *Research-Technology Management*, 2017, 60.4: 33-42.

SNOWFLAKE. *What is Parquet? A guide to the columnar storage format* [online]. 2024. Dostupné z: https://www.snowflake.com/guides/what-parquet

SUN, Zhaohao; STRANG, Kenneth David; PAMBEL, Francisca. Privacy and security in the big data paradigm. *Journal of Computer Information Systems*, 2020, 60.2: 146-155.

TOSI, Davide; KOKAJ, Redon; ROCCETTI, Marco. 15 years of Big Data: a systematic literature review. *Journal of Big Data*, 2024, 11.1: 73.

TOWERS, Sherry, et al. Factors influencing temporal patterns in crime in a large American city: A predictive analytics perspective. *PLoS one*, 2018, 13.10: e0205151.

VAN DER AALST, Wil M. P. Data scientist: The engineer of the future. In: *Enterprise interoperability VI: Interoperability for agility, resilience and plasticity of collaborations*. Springer International Publishing, 2014. s. 13-26.

VARGAS, Jason. The impact of socioeconomic factors on crime rates. *Addiction and Criminology*, 2023, 6.4: 161.

VIJAYARAJ, J., et al. A comprehensive survey on big data analytics tools. In: *2016 Online International Conference on Green Engineering and Technologies (IC-GET)*. IEEE, 2016. s. 1-6.

WASSOUF, Wissam Nazeer, et al. Predictive analytics using big data for increased customer loyalty: Syriatel Telecom Company case study. *Journal of Big Data*, 2020, 7.1: 29.

WHITE, Tom. *Hadoop: The definitive guide*. O'Reilly Media, Inc., 2012.

WIRTH, Rüdiger; HIPP, Jochen. CRISP-DM: Towards a standard process model for data mining. In: *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*. 2000. s. 29-39.

WOOD, Matt. Algorithm predicts crime a week in advance, but reveals bias in police response. University of Chicago [online]. 2022. Dostupné z: https://biologicalsciences.uchicago.edu/news/algorithm-predicts-crime-police-bias

YENDURI, Lakshmana Kumar. Performance Evaluation of Apache Hadoop, Spark, and Flink for Batch Processing of Big Data: A Comparative Analysis. In: *2024 Third International Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT)*. IEEE, 2024. s. 1-5.

ZAHARIA, Matei, et al. Apache spark: a unified engine for big data processing. *Communications of the ACM*, 2016, 59.11: 56-65.

ZAHARIA, Matei, et al. Resilient distributed datasets: A {Fault-Tolerant} abstraction for {In-Memory} cluster computing. In: *9th USENIX symposium on networked systems design and implementation (NSDI 12)*. 2012. s. 15-28.

ZAHARIA, Matei, et al. Spark: Cluster computing with working sets. In: *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*. 2010. s. 1-10.

ZHOU, Lina, et al. Machine learning on big data: Opportunities and challenges. *Neurocomputing*, 2017, 237: 350-361.

# SEZNAM PŘÍLOH

## *Příloha I: Instalace knihoven*

```
pip install Numpy
pip install SciPy
pip install Matplotlib
pip install Pandas
pip install Statsmodels
pip install Plotly
pip install Scikit-learn
pip install openpyxl
pip install spacy
pip install seaborn
pip install pgmpy
pip install apyori
pip install mlxtend
pip install networkx
pip install pySpark
pip install findSpark
pip install jupyter
pip install notebook
pip install folium geopandas plotly
pip install ipywidgets
```

## *Příloha II: Datový slovník*

Kompletní datový slovník. Zdroj: City of Chicago (2025).

| | |
|---|---|
| **ID** | Jedinečný identifikátor pro záznam. |
| **Case Number** | Číslo RD Chicago Police Department (číslo oddělení záznamů), které je pro tento incident jedinečné. |
| **Date** | Datum, kdy k incidentu došlo. to je někdy nejlepší odhad. |
| **Block** | Částečně upravená adresa, kde k incidentu došlo, umístěním do stejného bloku jako skutečná adresa. |
| **IUCR** | Kód Illinois Unifrom pro hlášení zločinu. To je přímo spojeno s primárním typem a popisem. Podívejte se na seznam kódů IUCR na https://data.cityofchicago.org/d/c7ck-438e. |
| **Primary Type** | Primární popis kódu IUCR. |
| **Description** | Sekundární popis kódu IUCR, podkategorie primárního popisu. |
| **Location Description** | Popis místa, kde k incidentu došlo. |
| **Arrest** | Označuje, zda došlo k zatčení. |
| **Domestic** | Označuje, zda se incident týkal domácího prostředí, jak je definováno zákonem Illinois Domestic Violence Act. |
| **Beat** | Označuje dobu, ve které k incidentu došlo. Beat je nejmenší policejní geografická oblast – každá doba má vyhrazené policejní auto. Tři až pět taktů tvoří policejní sektor a tři sektory tvoří policejní obvod. Chicagské policejní oddělení má 22 policejních obvodů. Podívejte se na beaty na https://data.cityofchicago.org/d/aerh-rz74. |
| **District** | Označuje policejní obvod, kde k incidentu došlo. Podívejte se na okresy na https://data.cityofchicago.org/d/fthy-xz3r. |
| **Ward** | Oddělení (okres městské rady), kde k incidentu došlo. Podívejte se na oddělení na https://data.cityofchicago.org/d/sp34-6z76. |
| **Community Area** | Označuje oblast komunity, kde k incidentu došlo. Chicago má 77 komunitních oblastí. Podívejte se na komunitní oblasti na https://data.cityofchicago.org/d/cauq-8yn6. |

| | |
|---|---|
| **FBI Code** | Označuje klasifikaci zločinu, jak je uvedeno v národním systému hlášení incidentů (NIBRS) FBI. Podívejte se na seznam těchto klasifikací policejního oddělení v Chicagu na http://gis.chicagopolice.org/clearmap_crime_sums/crime_types.html. |
| **X Coordinate** | Souřadnice x místa, kde k incidentu došlo v projekci State Plane Illinois East NAD 1983. Toto umístění je posunuto od skutečného umístění pro částečnou úpravu, ale spadá do stejného bloku. |
| **Y Coordinate** | Souřadnice Y místa, kde k incidentu došlo v projekci State Plane Illinois East NAD 1983. Toto umístění je posunuto od skutečného umístění pro částečnou úpravu, ale spadá do stejného bloku. |
| **Year** | Rok, kdy k incidentu došlo. |
| **Updated On** | Datum a čas poslední aktualizace záznamu. |
| **Latitude** | Zeměpisná šířka místa, kde k incidentu došlo. Toto umístění je posunuto od skutečného umístění pro částečnou úpravu, ale spadá do stejného bloku. |
| **Longitude** | Zeměpisná délka místa, kde k incidentu došlo. Toto umístění je posunuto od skutečného umístění pro částečnou úpravu, ale spadá do stejného bloku. |
| **Location** | Místo, kde k incidentu došlo, ve formátu, který umožňuje vytváření map a dalších geografických operací na tomto datovém portálu. Toto umístění je posunuto od skutečného umístění pro částečnou úpravu, ale spadá do stejného bloku. |

## *Příloha III: Převod CSV na PARQUET*

1) Načtení CSV datasetu a následný převod na PARQUET:

```
# Načtení CSV souboru a následné vytvoření tabulek
from pyspark.sql import SparkSession
from pyspark.sql.functions import monotonically_increasing_id

# Inicializace Spark Session
spark = SparkSession.builder \
    .appName('Crimes_Database') \
    .getOrCreate()

# Načtení souboru CSV
df = spark.read.format('csv').option('header',
'true').load('D:/Python/Crimes.csv')

# Přejmenování sloupců
df = df.withColumnRenamed('X Coordinate', 'X_Coordinates') \
    .withColumnRenamed('Y Coordinate', 'Y_Coordinates') \
    .withColumnRenamed('Case Number', 'Case_Number') \
    .withColumnRenamed('Date', 'Date') \
    .withColumnRenamed('IUCR', 'IUCR') \
    .withColumnRenamed('Updated On', 'Updated_On') \
    .withColumnRenamed('Primary Type', 'Primary_Type') \
    .withColumnRenamed('Description', 'Description') \
    .withColumnRenamed('Location Description', 'Location_Description') \
    .withColumnRenamed('Arrest', 'Arrest') \
    .withColumnRenamed('Beat', 'Beat') \
    .withColumnRenamed('District', 'District') \
    .withColumnRenamed('Ward', 'Ward') \
    .withColumnRenamed('Community Area', 'Community_Areas') \
    .withColumnRenamed('FBI Code', 'FBI_Code') \
    .withColumnRenamed('Latitude', 'Latitude') \
    .withColumnRenamed('Longitude', 'Longitude') \
    .withColumnRenamed('Domestic', 'Domestic') \
    .withColumnRenamed('Location', 'Location')

# Přidání sloupců s unikátními ID do DataFrame
```

```
df = df.withColumn('Date_ID_Records', monotonically_increasing_id()+1)
df = df.withColumn('Crime_ID_Records', monotonically_increasing_id()+1)
df = df.withColumn('Case_designation_ID_Records',
monotonically_increasing_id()+1)
df = df.withColumn('Coordinates_ID_Records', monotonically_increasing_id()+1)

# Vytvoření DataFramů pro každou tabulku na základě struktury v obrázku
coordinates_df = df.select('X_Coordinates', 'Y_Coordinates', 'Latitude',
'Longitude', 'Location')
case_designations_df = df.select('Beat', 'District', 'Ward',
'Community_Areas', 'FBI_Code')
records_df = df.select('Case_Number', 'Block', 'Date_ID_Records',
'Crime_ID_Records', 'Case_designation_ID_Records', 'Coordinates_ID_Records')
dates_df = df.select('Date', 'Year', 'Updated_On')
crimes_df = df.select('IUCR', 'Primary_Type', 'Description',
'Location_Description', 'Arrest', 'Domestic')

# Přidání sloupců s unikátními ID
coordinates_df = coordinates_df.withColumn('Coordinate_ID',
monotonically_increasing_id()+1)
case_designations_df = case_designations_df.withColumn('Case_designation_ID',
monotonically_increasing_id()+1)
dates_df = dates_df.withColumn('Date_ID', monotonically_increasing_id()+1)
crimes_df = crimes_df.withColumn('Crime_ID', monotonically_increasing_id()+1)

# Uložení DataFramu do souboru PARQUET s případným přepsáním existujícího
souboru
records_df.write.format('parquet').mode('overwrite').save('D:\Python\database\
DatabaseCrimesRecords.parquet')
case_designations_df.write.format('parquet').mode('overwrite').save('D:\Python
\database\DatabaseCaseDesignationsRecords.parquet')
dates_df.write.format('parquet').mode('overwrite').save('D:\Python\database\Da
tabaseCrimesDates.parquet')
crimes_df.write.format('parquet').mode('overwrite').save('D:\Python\database\D
atabaseCrimesCrimes.parquet')
coordinates_df.write.format('parquet').mode('overwrite').save('D:\Python\datab
ase\DatabaseCrimesCoordinates.parquet')
```

2) Načtení PARQUET souborů

```
from pyspark.sql import SparkSession
```

```
spark = SparkSession.builder \
    .appName("YourAppName") \
    .config("spark.executor.memory", "6g") \
    .getOrCreate()

# Načtení souborů Parquet do DataFrame
records_df =
spark.read.parquet('D:\Python\PARQUET\DatabaseCrimesRecords.parquet')
case_designations_df =
spark.read.parquet('D:\Python\PARQUET\DatabaseCaseDesignationsRecords.parquet'
)
dates_df = spark.read.parquet('D:\Python\PARQUET\DatabaseCrimesDates.parquet')
crimes_df =
spark.read.parquet('D:\Python\PARQUET\DatabaseCrimesCrimes.parquet')
coordinates_df =
spark.read.parquet('D:\Python\PARQUET\DatabaseCrimesCoordinates.parquet')

# Registrace DataFrame jako dočasné tabulky
records_df.createOrReplaceTempView('records')
dates_df.createOrReplaceTempView('dates')
crimes_df.createOrReplaceTempView('crimes')
coordinates_df.createOrReplaceTempView('coordinates')
```

## *Příloha IV: Vytváření modelu*

**Analýza 1:** Vizualizace četnosti popisů zločinů

```
import matplotlib.pyplot as plt
# Předpokládejme, že 'result' obsahuje výstup z vašeho Spark SQL dotazu
result = spark.sql("SELECT * FROM crimes")
# Přepočítání četnosti hodnot v sloupci 'Description'
description_frequency = result.groupBy('Description').count().toPandas()
# Seřazení dat podle četnosti a výběr prvních 10
description_frequency = description_frequency.sort_values('count', ascending=False).head(10)
# Vytvoření sloupcového grafu
plt.figure(figsize=(10,8))
plt.bar(description_frequency['Description'], description_frequency['count'], color='#1f77b4')  # modrá barva
plt.xlabel('Description')
plt.ylabel('Četnost')
plt.title('Top 10 nejčastějších popisů zločinů – Description')
# Zobrazení hodnot nad sloupci s větším posunem
for i, count in enumerate(description_frequency['count']):
    plt.text(i, count + 10000, str(count), ha='center')  # posun o 2 % výšky sloupce
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

**Analýza 2:** Vizualizace četnosti hodnot ‘Year’ v datasetu ‘dates’

```
import matplotlib.pyplot as plt
# Původní SQL dotaz (v kódu zůstává stejný)
result = spark.sql("""
    SELECT * FROM dates
""")
# Vypočítání četnosti hodnot v sloupci 'Year'
year_counts = result.groupBy('Year').count().toPandas()
# Seřazení podle roku
year_counts = year_counts.sort_values(by='Year')
# Vytvoření sloupcového grafu
plt.figure(figsize=(10, 8))
```

```
plt.bar(year_counts['Year'], year_counts['count'], color='#1f77b4')
plt.xlabel('Rok')
plt.ylabel('Četnost')
plt.title('Četnost hodnot roku')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

**Analýza 3:** Vizualizace geografických dat o zločinech v Chicagu v roce 2008

```
import numpy as np
import folium
import matplotlib.pyplot as plt
import matplotlib.colors as mcolors
from pyspark.sql import Row
result = spark.sql("""
    SELECT cr.Description, r.Case_Number, d.Date, d.Year, c.Latitude,
c.Longitude
    FROM records r
    LEFT JOIN dates d ON r.Date_ID_Records = d.Date_ID
    LEFT JOIN coordinates c ON r.Coordinates_ID_Records = c.Coordinate_ID
    LEFT JOIN crimes cr ON r.Crime_ID_Records = cr.Crime_ID
    WHERE d.Year = 2008
""")
# Výběr 500 náhodných řádků
result_sample = spark.createDataFrame(result.rdd.takeSample(False, 500))
# Převedení výsledku dotazu na pandas dataframe
pdf = result_sample.toPandas()
# Inicializace objektu mapy s centrem v Chicagu
m = folium.Map(location=[41.8781, -87.6298], zoom_start=11)
# Vytvoření barevné mapy
colors = plt.cm.hsv(np.linspace(0, 1, len(pdf['Description'].unique())))
cmap = mcolors.ListedColormap(colors)
# Iterace přes řádky a přidání značek do objektu mapy
for index, row in pdf.iterrows():
    if row['Latitude'] is not None and row['Longitude'] is not None:
        color =
mcolors.rgb2hex(cmap(pdf['Description'].unique().tolist().index(row['Descripti
on'])))
```

```
        folium.CircleMarker([row['Latitude'], row['Longitude']], color=color,
fill=True, fill_color=color, radius=5,
popup=f"{row['Description']}").add_to(m)


# Zobrazení mapy
m
```

**Analýza 4:** Vizualizace četnosti hodnot ‘Primary_Type’ v datasetu ‘crimes’

```
import matplotlib.pyplot as plt
# Vytvoření DataFrame s četností hodnoty 'Primary_Type'
primary_type_frequency = spark.sql("""
    SELECT Primary_Type, COUNT(*) as Frequency
    FROM crimes
    GROUP BY Primary_Type
    ORDER BY Frequency DESC
    LIMIT 5  -- Zde omezíme výběr na prvních 5 záznamů
""")

# Převod DataFrame do Pandas pro vizualizaci
primary_type_frequency_pandas = primary_type_frequency.toPandas()

# Vytvoření sloupcového grafu
plt.figure(figsize=(10, 6))
bars = plt.bar(primary_type_frequency_pandas['Primary_Type'],
primary_type_frequency_pandas['Frequency'], color='#1f77b4')

# Přidání hodnot nad sloupce
for i, v in enumerate(primary_type_frequency_pandas['Frequency']):
    plt.text(i, v + 10, str(v), ha='center', va='bottom')

plt.xlabel('Typ zločinu')
plt.ylabel('Četnost')
plt.title('Top 5 Typů zločinu podle četnosti')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

**Analýza 5:** Vizualizace geografických dat o zločinech v Chicagu

```
import folium
import branca
# Vytvoření DataFrame s četností hodnot 'Longitude' a 'Latitude'
location_frequency = spark.sql("""
    SELECT c.Longitude, c.Latitude, COUNT(*) as Frequency
    FROM records r
    LEFT JOIN coordinates c ON r.Coordinates_ID_Records = c.Coordinate_ID
WHERE c.Longitude IS NOT NULL
    GROUP BY c.Longitude, c.Latitude
    ORDER BY Frequency DESC
    LIMIT 20
""")
# Převod DataFrame do Pandas pro vizualizaci
location_frequency_pandas = location_frequency.toPandas()
# Vytvoření mapy Chicaga
chicago_map = folium.Map(location=[41.8781, -87.6298], zoom_start=10)
# Definice barevné škály
colorscale =
branca.colormap.linear.YlOrRd_09.scale(location_frequency_pandas['Frequency'].
min(), location_frequency_pandas['Frequency'].max())
# Přidání bodů na mapu
for index, row in location_frequency_pandas.iterrows():
    folium.CircleMarker(
        location=[row['Latitude'], row['Longitude']],
        radius=5,
        color="black",  # Barva ohraničení
        weight=1,  # Tloušťka ohraničení
        fill=True,
        fill_color=colorscale(row['Frequency']),  # Barva výplně
        fill_opacity=0.6,
        popup=f"Frequency: {row['Frequency']}"
    ).add_to(chicago_map)

# Přidání legendy
colorscale.caption = 'Frequency of Crimes'
chicago_map.add_child(colorscale)
# Zobrazení mapy
```

```
chicago_map
```

**Analýza 6:** Analýza vztahu mezi zatčením a domácím násilím ve zločinech v určité oblasti

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from tkinter import messagebox
import tkinter as tk
from tkinter import ttk
import calendar
from pyspark.sql import SparkSession

# Načtení dat pomocí SQL
sampled_data = spark.sql("""
    SELECT Arrest, Domestic
    FROM crimes
""")

# Rozdělení dat na menší části a převod do Pandas DataFrame
chunk_size = 100000  # Počet řádků, které budeme načítat najednou
num_rows = sampled_data.count()  # Celkový počet řádků
sampled_data_df = pd.DataFrame()  # Prázdný DataFrame pro výsledky
for i in range(0, num_rows, chunk_size):
    chunk = sampled_data.limit(chunk_size).toPandas()  # Načtení další části dat
    sampled_data_df = pd.concat([sampled_data_df, chunk], ignore_index=True) # Přidání části k výslednému DataFrame

# Vykreslení grafu
plt.figure(figsize=(10, 6))
sns.countplot(x='Arrest', hue='Domestic', data=sampled_data_df, palette=['#1f77b4', '#ff7f0e'])
plt.title(' Zatčení vs. Domácí násilí')
plt.xlabel('Zatčení')
plt.ylabel('Četnost (miliony)')
plt.xticks(rotation=45)
#plt.grid(axis='y', linestyle='--', alpha=0.5)
plt.legend(title='Domácí násilí', loc='upper right')
```

```
plt.tight_layout()
plt.show()
```

**Analýza 7:** Vizualizace četnosti zatčení podle roku v datasetu ‘crimes‘

```
import matplotlib.pyplot as plt
import numpy as np
primary_type_frequency = spark.sql("""
SELECT c.Arrest, d.Year
FROM crimes c
JOIN records r ON r.Crime_ID_Records = c.Crime_ID
LEFT JOIN dates d ON r.Date_ID_Records = d.Date_ID
""")
# Převod DataFrame do Pandas pro vizualizaci
arrest_frequency_pandas = primary_type_frequency.toPandas()
# Příprava dat pro graf
data = arrest_frequency_pandas.groupby(['Year',
'Arrest']).size().unstack(fill_value=0)
# Vytvoření sloupcového grafu
data.plot(kind='bar', stacked=False, figsize=(10,5))
plt.xlabel('Rok')
plt.ylabel('Četnost')
plt.title('Porovnání počtu zatčení a nezatčení podle let')
plt.legend(title='Zatčení', loc='upper right')
plt.show()
```

**Analýza 8:** Časová křivka četnosti zločinů podle měsíce v roce v datasetu ‘dates’

```
from datetime import datetime
import matplotlib.pyplot as plt
import pandas as pd
from pyspark.sql.functions import to_date, year, month
from ipywidgets import interact
import calendar  # Import modulu calendar pro práci s měsíci
# Nastavení spark.sql.legacy.timeParserPolicy na LEGACY
spark.conf.set("spark.sql.legacy.timeParserPolicy", "LEGACY")
# Váš SQL kód pro načtení dat
primary_type_frequency = spark.sql("""
SELECT Date FROM dates
```

```
""")
# Převod řetězců na datetime objekty a extrakce roku, měsíce a dne
primary_type_frequency = primary_type_frequency.withColumn('Date', to_date('Date', 'MM/dd/yyyy'))
# Výpočet počtu záznamů pro každý den
date_counts = primary_type_frequency.groupBy(year('Date').alias('Year'), month('Date').alias('Month')).count()
# Seřazení podle roku a měsíce
date_counts = date_counts.sort(['Year', 'Month'])
# Převod Spark DataFrame na Pandas DataFrame
pdf = date_counts.toPandas()
# Převod sloupce 'Date' na datetime
pdf['Date'] = pd.to_datetime(pdf[['Year', 'Month']].assign(day=1))
# Funkce pro vytvoření časové křivky pro vybraný rok
def plot_year(year):
    # Filtrace dat pro vybraný rok
    year_data = pdf[pdf['Year'] == year]

    # Vytvoření časové křivky
    plt.figure(figsize=(10, 6))  # Zvětšení grafu
    plt.plot(year_data['Date'].dt.month, year_data['count'], marker='o')
    plt.xticks(range(1, 13), calendar.month_name[1:13], rotation=45)  # Nastavení popisků měsíců
    plt.xlabel('Month')
    plt.ylabel('Count')
    plt.title(f'Count by Month in {year}')
    plt.grid(True)
    plt.show()

# Vytvoření widgetu pro výběr roku
interact(plot_year, year=(pdf['Year'].min(), pdf['Year'].max()))
```

**Analýza 9:** Průměrný počet zločinů v jednotlivých měsících za všechny roky

```
import pandas as pd
import matplotlib.pyplot as plt
from pyspark.sql import SparkSession
from tkinter import messagebox
# Dotaz na průměrný počet zločinů za jednotlivé měsíce
```

```
monthly_data = spark.sql("""
    SELECT MONTH(TO_TIMESTAMP(Date, 'MM/dd/yyyy hh:mm:ss a')) as Month, COUNT(*) as Count
    FROM dates d
    LEFT JOIN records r ON r.Date_ID_Records = d.Date_ID
    GROUP BY Month
    ORDER BY Month
""")
# Převod Spark DataFrame na Pandas DataFrame
pdf = monthly_data.toPandas()
# Přidání prázdných řádků pro měsíce, které nemají žádné záznamy
all_months = pd.DataFrame({'Month': range(1, 13), 'Count': [0]*12})
pdf = pd.merge(all_months, pdf, on='Month', how='left').fillna(0)
# Výpočet průměrného počtu zločinů za jednotlivé měsíce
pdf['Average'] = pdf['Count_y'] / len(pdf)  # Zde můžete upravit, pokud máte více než jeden rok
# Vytvoření sloupcového grafu
plt.figure(figsize=(10, 6))
plt.bar(pdf['Month'], pdf['Average'], color='blue')
plt.xticks(pdf['Month'], ['Leden', 'Únor', 'Březen', 'Duben', 'Květen', 'Červen',
           'Červenec', 'Srpen', 'Září', 'Říjen', 'Listopad', 'Prosinec'], rotation=45)
plt.xlabel('Měsíc')
plt.ylabel('Průměrný počet zločinů')
plt.title('Průměrný počet zločinů v jednotlivých měsících za všechny roky')
plt.grid(axis='y')
plt.tight_layout()  # Automatické nastavení rozložení
plt.show()
```

**Analýza 10:** Četnost zločinů v jednotlivých hodinách

```
import pandas as pd
import matplotlib.pyplot as plt
from pyspark.sql import SparkSession
from tkinter import messagebox
# Dotaz na počet zločinů za jednotlivé hodiny
hourly_data = spark.sql("""
    SELECT HOUR(TO_TIMESTAMP(Date, 'MM/dd/yyyy hh:mm:ss a')) as Hour, COUNT(*) as Count
```

```
    FROM dates d
    LEFT JOIN records r ON r.Date_ID_Records = d.Date_ID
    GROUP BY Hour
    ORDER BY Hour
""")
# Převod Spark DataFrame na Pandas DataFrame
pdf = hourly_data.toPandas()
# Přidání prázdných řádků pro hodiny, které nemají žádné záznamy
all_hours = pd.DataFrame({'Hour': range(24), 'Count': [0]*24})
pdf = pd.merge(all_hours, pdf, on='Hour', how='left').fillna(0)
# Zajištění, že 'Count' je typu int
pdf['Count'] = pdf['Count_y'].astype(int)  # nebo můžete použít .fillna(0) a
.astype(int)
# Vytvoření časové křivky pro jednotlivé hodiny
plt.figure(figsize=(10, 6))
plt.plot(pdf['Hour'], pdf['Count'], marker='o', linestyle='-', color='blue')
plt.xticks(pdf['Hour'], rotation=45)  # Otočení popisků na x-ové ose
plt.xlabel('Hodina')
plt.ylabel('Počet zločinů')
plt.title('Četnost zločinů v jednotlivých hodinách')
plt.grid()
plt.tight_layout()  # Automatické nastavení rozložení
plt.show()
```

**Analýza 11:** Geografické shlukování kriminality

```
from pyspark.sql import SparkSession
from pyspark.ml.clustering import KMeans
from pyspark.ml.evaluation import ClusteringEvaluator
from pyspark.ml.feature import VectorAssembler
import pandas as pd
import folium

# SQL dotaz pro získání dat se správnými souřadnicemi
query = """
    SELECT c.Primary_Type, d.Year, co.Latitude, co.Longitude
    FROM records r
    JOIN crimes c ON r.Crime_ID_Records = c.Crime_ID
```

```
    JOIN dates d ON r.Date_ID_Records = d.Date_ID
    JOIN coordinates co ON r.Coordinates_ID_Records = co.Coordinate_ID
    WHERE co.Latitude IS NOT NULL AND co.Longitude IS NOT NULL
"""
df = spark.sql(query).toPandas()

# Převod souřadnic na čísla a odstranění neplatných hodnot
df['Latitude'] = pd.to_numeric(df['Latitude'], errors='coerce')
df['Longitude'] = pd.to_numeric(df['Longitude'], errors='coerce')

# Odstranění řádků, kde jsou NaN hodnoty po konverzi
df = df.dropna(subset=['Latitude', 'Longitude'])

# Odstranění odlehlých hodnot
if not df.empty:
    df = df[(df['Latitude'].between(df['Latitude'].quantile(0.02),
df['Latitude'].quantile(0.98))) &
            (df['Longitude'].between(df['Longitude'].quantile(0.02),
df['Longitude'].quantile(0.98)))]
else:
    print("Chyba: DataFrame je prázdný po SQL dotazu.")

# Převod Pandas DataFrame zpět na PySpark DataFrame
df_spark = spark.createDataFrame(df)

# Použití VectorAssembleru
assembler = VectorAssembler(inputCols=['Latitude', 'Longitude'],
outputCol='features')
df_spark = assembler.transform(df_spark)

# Přerozdělení dat do 100 partitionů pro efektivnější zpracování
df_spark = df_spark.repartition(100)

# Použití Spark MLlib K-Means
kmeans = KMeans(k=5, seed=42, featuresCol='features')
model = kmeans.fit(df_spark)
df_spark = model.transform(df_spark)
```

```
# Výpočet metrik kvality shlukování
evaluator = ClusteringEvaluator(featuresCol='features')
silhouette_avg = evaluator.evaluate(df_spark)

# Výpis metrik kvality modelu
print(f"Silhouette Score: {silhouette_avg}")
print(f"Inertia (Sum of squared distances to cluster centers):
{model.summary.trainingCost}")

# Výpis velikostí shluků
cluster_sizes = df_spark.groupBy("prediction").count().toPandas()
print("Cluster sizes:")
print(cluster_sizes)

# Vizualizace na mapě
cluster_centers = model.clusterCenters()
if not df.empty:
    map_center = [df['Latitude'].mean(), df['Longitude'].mean()]
    crime_map = folium.Map(location=map_center, zoom_start=11)
    for idx, center in enumerate(cluster_centers):
        folium.Circle(
            location=[center[0], center[1]],
            radius=1000,  # Zvýšený poloměr pro lepší vizualizaci
            color='blue',
            fill=True,
            fill_color='blue',
            fill_opacity=0.3,
            popup=f'Cluster {idx}'
        ).add_to(crime_map)
    # Uložení mapy do souboru
    crime_map.save("D:\\Python\\crime_clusters_map.html")
    print("Mapa shluků kriminality byla uložena jako crime_clusters_map.html")
else:
    print("Chyba: DataFrame je prázdný, vizualizace nebyla provedena.")
```

## *Příloha V: Testovací návrh*

### Testovací návrh Analýzy 1:

```
import matplotlib.pyplot as plt
from pyspark.sql import SparkSession

# Inicializace SparkSession
spark = SparkSession.builder.getOrCreate()

# Kontrola načtení dat
result = spark.sql("""SELECT * FROM crimes""")
assert 'Description' in result.columns, "Chybí sloupec 'Description'."

# Vypočítání četnosti hodnot v sloupci 'Description'
description_frequency = result.groupBy('Description').count().toPandas()

# Seřazení dat podle četnosti a výběr prvních 10
description_frequency = description_frequency.sort_values('count', ascending=False).head(10)
assert len(description_frequency) == 10, "Nebylo vybráno správné množství popisů."

# Vytvoření sloupcového grafu
plt.figure(figsize=(10,8))
plt.bar(description_frequency['Description'], description_frequency['count'], color='#1f77b4')  # modrá barva
plt.xlabel('Popis')
plt.ylabel('Počet')
plt.title('Top 10 nejčastějších popisů zločinů')

# Zobrazení hodnot nad sloupci s vyšším posunem
for i, count in enumerate(description_frequency['count']):
    plt.text(i, count + 0.05 * count, str(count), ha='center')  # posun o 5 % výšky sloupce

# Nastavení titulu grafu
plt.title('Top 10 nejčastějších popisů zločinů')
assert plt.title, "Titul grafu není nastavený."
```

```
# Zobrazení grafu
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

**Testovací návrh Analýzy 2:**

```
import matplotlib.pyplot as plt
from pyspark.sql import SparkSession
# Inicializace SparkSession
spark = SparkSession.builder.getOrCreate()
# Původní SQL dotaz (v kódu zůstává stejný)
result = spark.sql("""
    SELECT * FROM dates
""")
# Vypočítání četnosti hodnot v sloupci 'Year'
year_counts = result.groupBy('Year').count().toPandas()
# Seřazení podle roku
year_counts = year_counts.sort_values(by='Year')
# Vytvoření sloupcového grafu
plt.figure(figsize=(10, 8))
plt.bar(year_counts['Year'],   year_counts['count'],   color='#1f77b4')      #
nastavení barvy na #1f77b4
plt.xlabel('Rok')
plt.ylabel('Četnost')
plt.title('Četnost hodnot roku')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

**Testovací návrh Analýzy 3:**

```
import numpy as np
import folium
import matplotlib.pyplot as plt
import matplotlib.colors as mcolors
from pyspark.sql import SparkSession, Row
# Inicializace SparkSession
spark = SparkSession.builder.getOrCreate()
```

```
# Kontrola načtení dat
result = spark.sql("""
    SELECT cr.Description, r.Case_Number, d.Date, d.Year, c.Latitude, c.Longitude
    FROM records r
    LEFT JOIN dates d ON r.Date_ID_Records = d.Date_ID
    LEFT JOIN coordinates c ON r.Coordinates_ID_Records = c.Coordinate_ID
    LEFT JOIN crimes cr ON r.Crime_ID_Records = cr.Crime_ID
    WHERE d.Year = 2008
""")
assert 'Description' in result.columns, "Chybí sloupec 'Description'."
assert 'Latitude' in result.columns, "Chybí sloupec 'Latitude'."
assert 'Longitude' in result.columns, "Chybí sloupec 'Longitude'."
# Výběr 500 náhodných řádků
result_sample = spark.createDataFrame(result.rdd.takeSample(False, 500))
assert result_sample.count() == 500, "Nebylo vybráno správné množství náhodných řádků."
# Převedení výsledku dotazu na pandas dataframe
pdf = result_sample.toPandas()
# Inicializace objektu mapy s centrem v Chicagu
m = folium.Map(location=[41.8781, -87.6298], zoom_start=11)
# Vytvoření barevné mapy
colors = plt.cm.hsv(np.linspace(0, 1, len(pdf['Description'].unique())))
cmap = mcolors.ListedColormap(colors)
# Iterace přes řádky a přidání značek do objektu mapy
for index, row in pdf.iterrows():
    if row['Latitude'] is not None and row['Longitude'] is not None:
        color = mcolors.rgb2hex(cmap(pdf['Description'].unique().tolist().index(row['Description'])))
        folium.CircleMarker([row['Latitude'], row['Longitude']], color=color, fill=True, fill_color=color, radius=5, popup=f"{row['Description']}").add_to(m)

# Zobrazení mapy
m
```

**Testovací návrh Analýzy 4:**

```
import matplotlib.pyplot as plt
```

```
from pyspark.sql import SparkSession
# Inicializace SparkSession
spark = SparkSession.builder.getOrCreate()
# Původní SQL dotaz (v kódu zůstává stejný)
primary_type_frequency = spark.sql("""
    SELECT Primary_Type, COUNT(*) as Frequency
    FROM crimes
    GROUP BY Primary_Type
    ORDER BY Frequency DESC
    LIMIT 5  -- Zde omezíme výběr na prvních 5 záznamů
""")
# Převod DataFrame do Pandas pro vizualizaci
primary_type_frequency_pandas = primary_type_frequency.toPandas()
# Vytvoření sloupcového grafu
plt.figure(figsize=(10, 6))
bars = plt.bar(primary_type_frequency_pandas['Primary_Type'],
primary_type_frequency_pandas['Frequency'], color='#1f77b4')
# Přidání hodnot nad sloupce
for i, v in enumerate(primary_type_frequency_pandas['Frequency']):
    plt.text(i, v + 10, str(v), ha='center', va='bottom')

plt.xlabel('Typ zločinu')
plt.ylabel('Četnost')
plt.title('Top 5 Typů zločinu podle četnosti')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

**Testovací návrh Analýzy 5:**

```
import folium
import branca
from pyspark.sql import SparkSession
# Inicializace SparkSession
spark = SparkSession.builder.getOrCreate()
# Původní SQL dotaz (v kódu zůstává stejný)
location_frequency = spark.sql("""
    SELECT c.Longitude, c.Latitude, COUNT(*) as Frequency
```

```
    FROM records r
    LEFT JOIN coordinates c ON r.Coordinates_ID_Records = c.Coordinate_ID
    WHERE c.Longitude IS NOT NULL
    GROUP BY c.Longitude, c.Latitude
    ORDER BY Frequency DESC
    LIMIT 20
""")
# Převod DataFrame do Pandas pro vizualizaci
location_frequency_pandas = location_frequency.toPandas()
# Vytvoření mapy Chicaga
chicago_map = folium.Map(location=[41.8781, -87.6298], zoom_start=10)
# Definice barevné škály
colorscale                                                                    =
branca.colormap.linear.YlOrRd_09.scale(location_frequency_pandas['Frequency'].
min(), location_frequency_pandas['Frequency'].max())
# Přidání bodů na mapu
for index, row in location_frequency_pandas.iterrows():
    folium.CircleMarker(
        location=[row['Latitude'], row['Longitude']],
        radius=5,
        color="black",  # Barva ohraničení
        weight=1,  # Tloušťka ohraničení
        fill=True,
        fill_color=colorscale(row['Frequency']),  # Barva výplně
        fill_opacity=0.6,
        popup=f"Frequency: {row['Frequency']}"
    ).add_to(chicago_map)

# Přidání legendy
colorscale.caption = 'Frequency of Crimes'
chicago_map.add_child(colorscale)
# Zobrazení mapy
chicago_map
```

**Testovací návrh Analýzy 6:**

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
from pyspark.sql import SparkSession
# Inicializace SparkSession
spark = SparkSession.builder.getOrCreate()
# Načtení dat pomocí SQL
sampled_data = spark.sql("""
    SELECT Arrest, Domestic
    FROM crimes
""")
# Rozdělení dat na menší části a převod do Pandas DataFrame
chunk_size = 100000  # Počet řádků, které budeme načítat najednou
num_rows = sampled_data.count()  # Celkový počet řádků
sampled_data_df = pd.DataFrame()  # Prázdný DataFrame pro výsledky
for i in range(0, num_rows, chunk_size):
    chunk = sampled_data.limit(chunk_size).toPandas()  # Načtení další části dat
    sampled_data_df = pd.concat([sampled_data_df, chunk], ignore_index=True)  # Přidání části k výslednému DataFrame

# Vykreslení sloupcového grafu
plt.figure(figsize=(10, 6))
sns.countplot(x='Arrest', hue='Domestic', data=sampled_data_df, palette=['#1f77b4', '#ff7f0e'])
plt.title('Zatčení vs. Domácí násilí')
plt.xlabel('Zatčení')
plt.ylabel('Četnost (miliony)')
plt.xticks(rotation=45)
plt.legend(title='Domácí násilí', loc='upper right')
plt.tight_layout()
plt.show()
```

**Testovací návrh Analýzy 7:**

```
import matplotlib.pyplot as plt
import numpy as np
from pyspark.sql import SparkSession

# Inicializace SparkSession
spark = SparkSession.builder.getOrCreate()
```

```
# Kontrola načtení dat
result = spark.sql("""
SELECT c.Arrest, d.Year
FROM crimes c
JOIN records r ON r.Crime_ID_Records = c.Crime_ID
LEFT JOIN dates d ON r.Date_ID_Records = d.Date_ID
""")
assert 'Arrest' in result.columns, "Chybí sloupec 'Arrest'."
assert 'Year' in result.columns, "Chybí sloupec 'Year'."
# Převod DataFrame do Pandas pro vizualizaci
arrest_frequency_pandas = result.toPandas()
# Příprava dat pro graf
data = arrest_frequency_pandas.groupby(['Year', 'Arrest']).size().unstack(fill_value=0)

# Vytvoření sloupcového grafu
data.plot(kind='bar', stacked=False, figsize=(10,5))
plt.xlabel('Year')
plt.ylabel('Frequency')
plt.title('Frequency of Arrests per Year')
# Zobrazení grafu
plt.show()
```

**Testovací návrh Analýzy 8:**

```
from datetime import datetime
import matplotlib.pyplot as plt
import pandas as pd
from pyspark.sql import SparkSession
from pyspark.sql.functions import to_date
from ipywidgets import interact

# Inicializace SparkSession
spark = SparkSession.builder \
    .appName("Crime Data Analysis") \
    .getOrCreate()

# Nastavení spark.sql.legacy.timeParserPolicy na LEGACY
```

```
spark.conf.set("spark.sql.legacy.timeParserPolicy", "LEGACY")

# Kontrola načtení dat
result = spark.sql("SELECT Date FROM dates")
assert 'Date' in result.columns, "Chybí sloupec 'Date'."

# Převod řetězců na datetime objekty
result = result.withColumn('Date', to_date('Date', 'MM/dd/yyyy'))

# Výpočet počtu záznamů pro každý den
date_counts = result.groupBy('Date').count()

# Seřazení podle data
date_counts = date_counts.sort('Date')

# Převod Spark DataFrame na Pandas DataFrame
pdf = date_counts.toPandas()

# Zajištění, že sloupec 'Date' je typu datetime
pdf['Date'] = pd.to_datetime(pdf['Date'])

# Funkce pro vytvoření časové křivky pro vybraný rok a měsíc
def plot_year_month(year, month):
    # Filtrace dat pro vybraný rok a měsíc
    year_month_data = pdf[(pdf['Date'].dt.year == year) & (pdf['Date'].dt.month
== month)]

    # Ověření, zda existují záznamy pro daný rok a měsíc
    if year_month_data.empty:
        print(f"Žádné záznamy pro {year} a měsíc {month}.")
        return

    # Vytvoření časové křivky
    plt.figure(figsize=(10, 6))  # Zvětšení grafu
    plt.plot(year_month_data['Date'].dt.day,          year_month_data['count'],
marker='o')
    plt.xticks(rotation=45)  # Otočení popisků na x-ové ose
    plt.xlabel('Den v měsíci')
```

```
    plt.ylabel('Počet zločinů')
    plt.title(f'Četnost zločinů v {year} - Měsíc {month}')
    plt.grid(True)
    plt.show()

# Vytvoření widgetu pro výběr roku a měsíce
interact(plot_year_month, year=(2001, 2024), month=(1, 12))
```

**Testovací návrh Analýzy 9:**

```
import unittest
from pyspark.sql import SparkSession

class TestCrimeAnalysis(unittest.TestCase):

    @classmethod
    def setUpClass(cls):
        cls.spark = SparkSession.builder \
            .appName("Test Crime Data Analysis") \
            .getOrCreate()
        data = [
            ('01/01/2022 01:00:00 AM',),
            ('01/01/2022 02:00:00 AM',),
            ('01/02/2022 01:00:00 AM',),
            ('02/01/2022 01:00:00 AM',),
            ('02/01/2022 02:00:00 AM',),
            ('02/02/2022 01:00:00 AM',),
            ('03/01/2022 01:00:00 AM',),
            ('03/01/2022 02:00:00 AM',),
            ('03/01/2022 03:00:00 AM',),
            ('03/01/2022 03:00:00 AM',),
        ]
        cls.df = cls.spark.createDataFrame(data, ['Date'])
        cls.df.createOrReplaceTempView("dates")

    def test_monthly_data(self):
        monthly_data = self.spark.sql("""
```

```
            SELECT MONTH(TO_TIMESTAMP(Date, 'MM/dd/yyyy hh:mm:ss a')) as Month,
COUNT(*) as Count
            FROM dates GROUP BY Month ORDER BY Month
        """)
        pdf = monthly_data.toPandas()
        self.assertEqual(pdf['Month'].tolist(), [1, 2, 3])
        expected_counts = {1: 2, 2: 3, 3: 4}
        for month in expected_counts.keys():
            self.assertEqual(pdf[pdf['Month']   ==   month]['Count'].values[0],
expected_counts[month])

    @classmethod
    def tearDownClass(cls):
        cls.spark.stop()

if __name__ == "__main__":
    unittest.main()
```

**Testovací návrh Analýzy 10:**

```
import unittest
import pandas as pd
from pyspark.sql import SparkSession
import matplotlib.pyplot as plt
from io import BytesIO

class TestCrimeHourMinuteAnalysis(unittest.TestCase):

    @classmethod
    def setUpClass(cls):
        cls.spark = SparkSession.builder \
            .appName("Test Crime Hour Minute Analysis") \
            .getOrCreate()
        data = [
            ('01/01/2022 01:00:00 AM',),
            ('01/01/2022 01:15:00 AM',),
            ('01/01/2022 01:30:00 AM',),
            ('01/01/2022 01:45:00 AM',),
            ('01/01/2022 02:00:00 AM',),
```

```
            ('01/01/2022 02:15:00 AM',),
            ('01/01/2022 02:30:00 AM',),
            ('01/01/2022 02:45:00 AM',),
        ]
        cls.df = cls.spark.createDataFrame(data, ['Date'])
        cls.df.createOrReplaceTempView("dates")
        records_data = [(i,) for i in range(1, 9)]
        cls.records_df = cls.spark.createDataFrame(records_data, ['Date_ID_Records'])
        cls.records_df.createOrReplaceTempView("records")

    def test_hour_minute_data(self):
        hour_minute_data = self.spark.sql("""
            SELECT HOUR(TO_TIMESTAMP(Date, 'MM/dd/yyyy hh:mm:ss a')) AS Hour,
                   MINUTE(TO_TIMESTAMP(Date, 'MM/dd/yyyy hh:mm:ss a')) AS Minute,
                   COUNT(*) AS Count
            FROM dates d
            LEFT JOIN records r ON r.Date_ID_Records = d.Date_ID
            GROUP BY Hour, Minute
            ORDER BY Hour, Minute
        """)
        pdf = hour_minute_data.toPandas()
        self.assertEqual(len(pdf), 8)
        expected_counts = {(1, 0): 1, (1, 15): 1, (1, 30): 1, (1, 45): 1,
                           (2, 0): 1, (2, 15): 1, (2, 30): 1, (2, 45): 1}
        for (hour, minute), count in expected_counts.items():
            actual_count = pdf[(pdf['Hour'] == hour) & (pdf['Minute'] == minute)]['Count'].values[0]
            self.assertEqual(actual_count, count)

    def test_plotting(self):
        hour_minute_data = self.spark.sql("""
            SELECT HOUR(TO_TIMESTAMP(Date, 'MM/dd/yyyy hh:mm:ss a')) AS Hour,
                   MINUTE(TO_TIMESTAMP(Date, 'MM/dd/yyyy hh:mm:ss a')) AS Minute,
                   COUNT(*) AS Count
            FROM dates d
```

```
            LEFT JOIN records r ON r.Date_ID_Records = d.Date_ID
            GROUP BY Hour, Minute
            ORDER BY Hour, Minute
        """)
        pdf = hour_minute_data.toPandas()
        pdf.dropna(subset=['Hour', 'Minute', 'Count'], inplace=True)
        pdf['Hour'] = pdf['Hour'].astype(int)
        pdf['Minute'] = pdf['Minute'].astype(int)
        pdf['Count'] = pdf['Count'].astype(int)
        plt.figure(figsize=(12, 6))
        plt.plot(pdf['Hour'] + pdf['Minute'] / 60, pdf['Count'], marker='o',
linestyle='-', color='blue')
        byte_stream = BytesIO()
        plt.savefig(byte_stream, format='png')
        plt.close()
        self.assertGreater(byte_stream.tell(), 0, "Graf nebyl vytvořen.")

    @classmethod
    def tearDownClass(cls):
        cls.spark.stop()

if __name__ == "__main__":
    unittest.main()
```

**Testovací návrh Analýzy 11:**

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import col
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.clustering import KMeans
from pyspark.ml.evaluation import ClusteringEvaluator
import pandas as pd
import folium

# Inicializace Spark session
spark = SparkSession.builder.appName("Crime Clustering Test").getOrCreate()

# Ověření správnosti SQL dotazu
```

```
query = """
    SELECT c.Primary_Type, d.Year, co.Latitude, co.Longitude
    FROM records r
    JOIN crimes c ON r.Crime_ID_Records = c.Crime_ID
    JOIN dates d ON r.Date_ID_Records = d.Date_ID
    JOIN coordinates co ON r.Coordinates_ID_Records = co.Coordinate_ID
    WHERE co.Latitude IS NOT NULL AND co.Longitude IS NOT NULL
"""
df = spark.sql(query).toPandas()
assert not df.empty, "Chyba: SQL dotaz nevrátil žádná data."
assert all(col in df.columns for col in ["Primary_Type", "Year", "Latitude",
"Longitude"]), "Chyba: Chybí očekávané sloupce."

# Ověření konverze datových typů
df['Latitude'] = pd.to_numeric(df['Latitude'], errors='coerce')
df['Longitude'] = pd.to_numeric(df['Longitude'], errors='coerce')
assert df['Latitude'].dtype == 'float64' and df['Longitude'].dtype ==
'float64', "Chyba: Konverze souřadnic selhala."

# Ověření odstranění NaN hodnot
df = df.dropna(subset=['Latitude', 'Longitude'])
assert not df.isnull().values.any(), "Chyba: DataFrame stále obsahuje NaN
hodnoty."

# Ověření odstranění odlehlých hodnot
if not df.empty:
    lat_range = (df['Latitude'].quantile(0.02), df['Latitude'].quantile(0.98))
    lon_range = (df['Longitude'].quantile(0.02),
df['Longitude'].quantile(0.98))
    df = df[(df['Latitude'].between(*lat_range)) &
(df['Longitude'].between(*lon_range))]
    assert not df.empty, "Chyba: Odstranění odlehlých hodnot vedlo k prázdnému
datasetu."

# Ověření shlukování
df_spark = spark.createDataFrame(df)
assembler = VectorAssembler(inputCols=['Latitude', 'Longitude'],
outputCol='features')
df_spark = assembler.transform(df_spark)
```

```
df_spark = df_spark.repartition(100)
kmeans = KMeans(k=5, seed=42, featuresCol='features')
model = kmeans.fit(df_spark)
df_spark = model.transform(df_spark)

# Ověření metrik kvality shlukování
evaluator = ClusteringEvaluator(featuresCol='features')
silhouette_avg = evaluator.evaluate(df_spark)
assert silhouette_avg > 0, "Chyba: Silhouette Score je neplatné."
assert model.summary.trainingCost > 0, "Chyba: Inertia je neplatná."

# Ověření velikostí shluků
cluster_sizes = df_spark.groupBy("prediction").count().toPandas()
assert len(cluster_sizes) == 5, "Chyba: Počet shluků není 5."

# Ověření vizualizace
cluster_centers = model.clusterCenters()
if not df.empty:
    map_center = [df['Latitude'].mean(), df['Longitude'].mean()]
    crime_map = folium.Map(location=map_center, zoom_start=11)
    for idx, center in enumerate(cluster_centers):
        folium.Circle(
            location=[center[0], center[1]],
            radius=1000,            color='blue',
            fill=True,
            fill_color='blue',
            fill_opacity=0.3,
            popup=f'Cluster {idx}'
        ).add_to(crime_map)
    crime_map.save("D:\\Python\\crime_clusters_map_test.html")
    assert crime_map, "Chyba: Vizualizace mapy selhala."
else:
    raise ValueError("Chyba: DataFrame je prázdný, vizualizace nebyla
provedena.")

print("Všechny testy proběhly úspěšně!")
```