TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Algoritmus pro detekci jízdních režimů

Diplomová práce

Studijní program:	N2612 Elektrotechnika a informatika
Studijní obor:	Informační technologie
Autor práce:	**Bc. Kryštof Brzák**
Vedoucí práce:	doc. Ing. Otto Severýn, Ph.D. Ústav mechatroniky a technické informatiky

Liberec 2020

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Zadání diplomové práce

Algoritmus pro detekci jízdních režimů

Jméno a příjmení: **Bc. Kryštof Brzák**
Osobní číslo: M17000122
Studijní program: N2612 Elektrotechnika a informatika
Studijní obor: Informační technologie
Zadávající katedra: Ústav mechatroniky a technické informatiky
Akademický rok: **2019/2020**

Zásady pro vypracování:

1. Seznamte se s formátem dat aplikace OneApp Škoda-auto a.s., které obsahují jízdní data (rychlost, zrychlení atd.) a popisují jednotlivé jízdy služebními vozy.
2. Navrhněte softwarové řešení pro předzpracování a uložení dat do odpovídajícího datového úložiště – předpokládá se dokumentově orientovaná NoSQL databáze, která umožní efektivní vyhledávání a agregaci takovýchto dat.
3. Navrhněte, implementujte a otestujte statistický či heuristický model, který bude na základě jízdních dat schopen detekovat jízdní režimy (např. neekonomické, nebezpečné, jízdu v koloně, případně další).
4. Vyhodnoťte výsledky modelu a případně navrhněte jeho další možná zlepšení.

Rozsah grafických prací: dle potřeby dokumentace
Rozsah pracovní zprávy: 40–50 stran
Forma zpracování práce: tištěná/elektronická
Jazyk práce: Čeština

Seznam odborné literatury:

[1] MCKINNEY, Wes. *Python for data analysis: data wrangling with pandas, NumPy, and IPython*. Second edition. Sebastopol, California: O'Reilly Media, 2018. ISBN 978-1491957660.
[2] KUBO, Noboru, et al. Vehicle behavior analysis system. U.S. Patent No 7,676,306, 2010.
[3] TAYLOR, Jeffrey, et al. Method for gathering, processing, and analyzing data to determine the risk associated with driving behavior. U.S. Patent Application No 13/508,941, 2013.
[4] WANG, Bo, et al. Driver identification using vehicle telematics data. SAE Technical Paper, 2017.

Vedoucí práce: doc. Ing. Otto Severýn, Ph.D.
Ústav mechatroniky a technické informatiky

Datum zadání práce: 10. října 2019
Předpokládaný termín odevzdání: 18. května 2020

L.S.

prof. Ing. Zdeněk Plíva, Ph.D.
děkan

doc. Ing. Milan Kolář, CSc.
vedoucí ústavu

V Liberci dne 10. října 2019

Prohlášení

Prohlašuji, že svou diplomovou práci jsem vypracoval samostatně jako původní dílo s použitím uvedené literatury a na základě konzultací s vedoucím mé diplomové práce a konzultantem.

Jsem si vědom toho, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci nezasahuje do mých autorských práv užitím mé diplomové práce pro vnitřní potřebu Technické univerzity v Liberci.

Užiji-li diplomovou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti Technickou univerzitu v Liberci; v tomto případě má Technická univerzita v Liberci právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Současně čestně prohlašuji, že text elektronické podoby práce vložený do IS/STAG se shoduje s textem tištěné podoby práce.

Beru na vědomí, že má diplomová práce bude zveřejněna Technickou univerzitou v Liberci 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.

Jsem si vědom následků, které podle zákona o vysokých školách mohou vyplývat z porušení tohoto prohlášení.

29. května 2020 Bc. Kryštof Brzák

Poděkování

Tímto děkuji vedoucímu práce doc. Ing. Ottu Severýnovi Ph.D. za trpělivost a svědomité a konstruktivní vedení diplomové práce. Poděkování patří také kolegům Ing. Štěpánovi Moníkovi za rady zkušeného programátora, Ing. Davidovi Židovi za veškeré informace ohledně vývoje aplikace OneApp.

Abstrakt

Tato práce pojednává o zpracování jízdních dat z automobilů značky Škoda. Stručně popisuje mobilní aplikaci OneApp, jejímž prostřednictvím jsou data sbírána. Pro jízdní data je navrženo uložiště v dokumentové databázi. Práce klade důraz na vyhodnocení kvality a úplnosti dat. Analýze předchází složitá příprava včetně převzorkování. V kontextu komerčního řešení v oblasti pojišťovnictví je navržen klasifikátor založený na algoritmu k-means. Program detekuje vzory v chování řidičů a třídí jednotlivé jízdy podle stylu řízení. Práce se zabývá problematikou sběru, přípravy a zpracování telematických dat.

Klíčová slova

Automobil, telemetrie, data mining, k-means

Abstract

Thesis deals with an analysis of driving data generated from Škoda cars. Mobile application OneApp, by through the data is collected, is briefly described. A document database repository is drafted for the driving data. Thesis puts emphasis on evaluation of data quality and completeness. In a context of commercial utility on the field of insurance, a k-means based classifier is proposed. Program detects driver behavior patterns and divides drives into clusters. Thesis is concerned with data collection, preparation and processing of telematic data.

Keywords

Automobile, telemetry, data mining, k-means

Obsah

Seznam tabulek

Seznam grafů

Definice pojmů a zkratek

Add-on	Nástavbový zásuvní modul zajišťující samostatnou funkci. Add-on je nezávislý na jádru a ostatních add-on
Widget	Výstup add-on ve formě dat zobrazených v grafickém uživatelském rozhraní
MIB modul	Modul zajišťující jízdní data z auta
Infotainment	Sloučenina slov information a entertainment, tj. zábava z informací, je jakékoli vestavěné technické řešení fungující jako služba zákazníkovi.
UBI	Usage based insurance – pojištění na základě používání
PAYD	Pay as you drive – model, podle něhož se výše pojištění rozhoduje na základě délky doby, ve které je automobil v provozu
MHYD	Manage how you drive – model pro správu klientů nebo vozů a stanovování výše pojištění na základě stylu řízení
PHYD	Pay how you drive – model, podle něhož se výše pojištění stanovuje na základě stylu řízení

Právní ujednání

Praktická část diplomové práce se zabývá zpracováním citlivých, zákaznických dat a je proto nezbytné deklarovat, jak je možné tato data využívat. Použití se v prvé řadě řídí podle nařízení Evropského parlamentu a rady Evropské unie ze dne 26.budna 2016 o ochraně osobních údajů (GDPR). Data poskytuje společnost Škoda Auto a.s. a způsob jejich využití se odvíjí také od smlouvy o zpracování závěrečné práce, kterou autor se společností uzavřel. Z důvodů zachování obchodního tajemství společnosti a bezpečnosti zákaznických dat, je tato práce neveřejná. Zákaznická data použitá v práci společnost klasifikuje jako důvěrná a podle pravidel o zpracování závěrečné práce nesmí být použita nebo zveřejňována. Data ale lze pozměnit a anonymizovat tak, aby splňovala legislativu i pravidla společnosti. Za tímto účelem budou podniknuty tyto kroky.

- Unikátní identifikátor vztahující se k uživatelskému profilu zákazníka bude nevratně nahrazen nesouvisejícím identifikátorem
- Identifikační číslo vozidla (VIN) bude nevratně změněno a nahrazeno nesouvisejícím identifikátorem
- V případě prezentace zeměpisné polohy budou odstraněny všechny záznamy, podle kterých je potenciálně možné určit identitu zákazníka
- V případě prezentace dat v čase dojde k zaokrouhlení časových hodnot do dvouhodinových intervalů, aby nebylo možné potenciálně určit identitu zákazníka v časoprostoru
- Po dokončení analýzy budou data odstraněna

Úvod

Tato práce vznikla na podnět vedoucího mé stáže Martina Sahuly BBA ve společnosti Škoda Auto. Úkolem bylo analyzovat jízdní data automobilů a vyhodnotit, zda lze data využít k obchodním účelům. V případě využitelnosti navrhnout systém jejich zpracování. Práce se soustředí na obchodní využití v oblasti pojišťovnictví vozů na základě jejich využívání.

Data, která jsou v této práci předmětem analýzy, jsou generována vestavěnými senzory automobilu. Jejich sběr probíhá prostřednictvím mobilní aplikace OneApp, která musí během jízdy komunikovat s MIB modulem automobilu. Aplikace slouží jako služba koncovým uživatelům a současně shromažďuje data z několika senzorů. Data jsou odesílána mobilním telefonem a ukládána do databáze. Pro analýzu byl použit datový set, exportovaný ze zmíněné databáze.

Cílem práce je navrhnout klasifikační algoritmus, který bude schopen rozeznat jízdní styly. Uvažuji dvě primární třídy; ekonomická a neekonomická jízda. Součástí řešení je návrh vlastní databáze pro snadnější manipulaci s daty. Z kompetenčních důvodů je program v této práci oddělen od aplikační architektury uvnitř společnosti.

Postup řešení je soustředěn ve čtyřech bodech. Nejprve je vytvořena krátká rešerše vědeckých publikací, které se vztahují k tématu analýzy jízdních dat. V druhém kroku jsou vybrány a popsány nástroje, kterými je práce provedena. Jde o typ databázovému systému a programovacího jazyka. Třetím krokem je ukládání dat do zvolené databáze a příprava pro analýzu. Posledním krokem je analýza samotná a její vyhodnocení.

1 Teoretická část

V druhé dekádě 21. století můžeme zaznamenat významný nárůst počtu zařízení připojených k internetu. Stále častěji se skloňován termín „internet věcí". Od mobilních telefonů se trend rozšířil do většiny oblastí průmyslu včetně automobilového. Vývoj umožňuje nasazení nových služeb s novými formami zisku. Nová odvětví informatiky a příbuzných oborů nabízí nespočet nových příležitostí od pojišťovnictví založeném na využívání vozu po autonomní řízení. Stupňuje se tlak na automobilky, aby byly schopné udržet krok s globální transformací směrem k digitálním službám a digitální ekonomice.

Spolupráce automobilek s IT a jeho začlenění do vlastních struktur byly jedny z primárních předpokladů k úspěšné transformaci. Doba si však žádá i další marketingové a strategické změny. Chápání mobility jako služby je významný trend ve společnosti a různé firmy se už této poptávce začaly přizpůsobovat. Například služby sdílených aut nebo spolujízdy už fungují několik let. V současnosti už jsou všechny vyráběné vozy opatřeny technologiemi pro nepřetržité připojení a komunikaci přes internet. Automobilky rovněž nasazují služby nazývané „infotainment", které poskytují zákazníkům služby založené mimo jiné i na jízdních datech (např. kniha jízd). Jednou z těchto služeb je i aplikace OneApp, jejímž prostřednictvím byla shromažďována data pro analýzu v této práci. Zvyšující se počty senzorů a čidel v automobilech z nich vytváří hodnotné generátory obrovských dat. Jedná se pravděpodobně o jeden z nejvýznamnějších objektů v síti internetu věcí. Vývoji nahrává i rozvoj mobilních aplikací a telekomunikačních sítí.

Dalším důležitým aspektem, jenž je rovněž zmíněn v praktické části práce, je kvalita telekomunikační infrastruktury a schopnost efektivně ukládat a zpracovávat data enormních rozměrů. Sběr a zpracování takových dat je jeden z nejnáročnějších úkolů v tomto průmyslovém odvětví. Je potřeba správně vyhodnotit, jaká data jsou relevantní pro konkrétní obchodní řešení. Jednotlivé aplikace se poptávají pouze po relevantních datech. Navzdory těmto opatřením i investicím do lepšího pokrytí, je v současnosti telekomunikační síť velice vytížena. Nelze předpokládat, že bude bezdrátová síť v blízké budoucnosti schopna zajistit plynulý přenos dat v takovém rozsahu. Nové systémy si vyžadují moderní, efektivní komunikační protokol schopný zajistit komunikaci mezi zařízeními i v případě náhlého výpadku spojení. Tradiční internetové komunikační protokoly jako http nejsou pro tyto systémy vhodné. Současně jsou kladeny vysoké nároky na ochranu dat a bezpečnost.

Následující kapitoly nastíní v současnosti nejrelevantnější byznysová řešení, která se v tomto odvětví rozvíjí a pomalu objevují na trhu.

1.1 Využití jízdních dat v pojišťovnictví

Populární využívání velkých dat nejen v automobilovém průmyslu je v pojišťovnictví. V článku (Arumugam and Bhargavi, 2019) je definován případ využití UBI – usage based insurance, neboli pojištění založené na způsobu využívání. V rámci UBI je popsán model PAYD – pay as you drive, tedy plať podle toho, jak často řídíš. V návaznosti na něj vznikl i model PHYD – pay how you drive, plať podle toho, jak řídíš. I tento model postoupil dál k MHYD – manage how you drive, tedy spravuj podle toho, jak řídíš. Jde o proaktivní management, který automaticky varuje řidiče v průběhu jejich jízd. Tyto tři modely jsou hlavními pilíři pojištění na základě využívání vozů. Zrychlující se nasazení modelů typu PHYD a MHYD napříč celým světem způsobuje adaptaci zákazníků na nový systém. Monitorování jejich řízení zvýšilo bezpečnost. Mezi zákazníky a pojišťovateli exponenciálně narostl objem přeposlaných dat a pojišťovny stále více potřebují schopnost data analyzovat.

Přestože jsou dopravní nehody v naprosté většině případů neúmyslné, předpokládá se, že lidský faktor je nejčastější příčinou jejich výskytu. Dalšími příčinami dopravních nehod jsou poruchy vozidel a externí vlivy jako je počasí nebo stav vozovky. Ztráta soustředění, jízda v podnapilém stavu, překračování povolené rychlosti, jízda na červenou, bezohlednost, agresivní styl jízdy nebo únava jsou hlavní lidské faktory, které potenciálně mohou způsobovat dopravní nehody. Autonomní řízení sice odstraní vliv lidských faktorů, ale jeho široké nasazení je otázkou mnoha následujících dekád (Arumugam and Bhargavi, 2019). Mezitím se počty registrací nových vozů stále zvyšují a zvyšuje se rovněž intenzita dopravy. Úměrně tomu roste i nehodovost, která se na vytížených komunikacích často objevuje a způsobuje ztráty lidské, materiální a časové.

Ve zmíněném článku (Arumugam and Bhargavi, 2019) jsou lidské faktory děleny na psychologické a behaviorální. Psychologické jsou způsobené především únavou, zatímco behaviorální mohou vypadat různě od roztržitosti po agresivního řízení. Agresivita za volantem není vzácným jevem a potenciálně vede k dopravním kolizím nebo jiným incidentům fyzického násilí. Agresivní styl řízení zahrnuje zmíněné překračování rychlosti, nebezpečné a rychlé střídání jízdních pruhů, nerespektování semaforů a dopravního značení. Méně často se vyskytuje hněv za volantem, jenž provází i výhružná gesta nebo zastrašující a agresivní chování.

Hlavní důvod zavedení UBI je schopnost detekovat a spolehlivě změřit takové incidenty. Výše pojistného se tedy bude přímo odvíjet od chování řidičů. Oproti modelu PAYD, který počítá

výši pojistného podle počtu ujetých kilometrů, je nový princip přesnější. Nárůst připojených zařízení umožňuje vyhodnocovat různé vzorce chování a personalizovat výši pojistného.

1.2 Metody sběru dat

Metody sběru dat, sbírané parametry i frekvence se neustále mění. Existuje několik způsobů, jak data od uživatelů získat.

- Černá skříňka – elektronické zařízení původně využívané pouze k ukládání informací vztahující se k dopravní nehodě, umožňuje komunikaci pouze v jednom směru
- Hardwarový klíč (dongl) – dodatečně připojené hardwarové zařízení zajišťující jednosměrnou komunikaci se serverem
- Vestavěný čip – v současnosti automobiloví výrobci běžně osazují své vozy vestavěnými mikročipy pro účely infotainmentu, dálkové diagnostiky vozidla nebo navigace
- Smartphone – zatím nejnovější telematické řešení, telefony fungují samostatně nebo jsou propojeny s vestavěným čipem automobilu pro komunikaci a výměnu jízdních dat

Senzory samotných telefonů, jako jsou akcelerometry nebo gyroskopy mohou poskytnout užitečné informace o řidičově chování s minimálními náklady. Nevýhodou je však nestálá pozice i rotace telefonu uvnitř vozu. Globální polohovací systém je dalším důležitým zdrojem informací. Systém GPS je dnes nasazen v nespočtu komerčních i veřejných aplikacích. Pomocí GPS lze určit přesnou zeměpisnou polohu, nadmořskou výšku, směr pohybu a při použití správných algoritmů i rychlost a míru zrychlení nebo zpomalení. Hodnoty nejsou příliš závislé na pozici nebo rotaci telefonu. Z poměrně malého množství dat tak lze potenciálně získat velké množství informací. Nevýhodou však je velká výpočetní náročnost.

V mnoha výzkumech se uvažuje o kombinaci obou přístupů. GPS souřadnice od uživatelů doplní nedostatky senzorů. Jakmile uživatel začne jízdu, v pravidelných intervalech se na server odesílají senzorová i poziční data. V jeden okamžik tak od všech uživatelů přijde obrovské množství dat, které tradiční servery nebudou schopny přijmout a zpracovat. Nové technologie v oblasti zpracování velkých dat umožňují dostatečně rychlé a efektivní přijímání, zpracování a komprimaci. Dále se pomocí metod strojového učení identifikují události během řízení. Na základě výskytu těchto událostí je poté stanovena výše pojistného.

1.3 Dosavadní využití velkých jízdních dat v pojišťovnictví

Firmy se snaží využít velká data k udržení nebo zlepšení konkurenceschopnosti. Současné možnosti výpočetní techniky jsou schopné nejen vyhodnocovat ale i úspěšně predikovat na základě historie a současného uživatelského profilu. V případě připojení například záznamů

dopravní policie, lze rozhodování o výši pojistného určovat s vysokou přesností. Velký potenciál je rovněž v komerční sféře, především pak u pojištění vozových parků firem.

Řada pojišťovacích firem už v praxi nasadila systémy založené na GPS k rozeznávání a vyhodnocování vzorů v chování řidičů. Firma IBM vyvinula službu pro řidiče v komerční sféře ("IBM Analytics - Insurance," 2015), která upozorňuje uživatele, pokud překračují některý ze stanovených limitů. Parametry jsou prudké zrychlování a zpomalování, rychlost a prudké projíždění zatáček. Allstate (vedoucí pojišťovací společnost ve Spojených státech) nasadila službu ("Drivewise from Allstate - Good Driver Discount," n.d.), která oceňuje a zvýhodňuje řidiče za bezpečné chování za volantem. Systém zahrnuje kromě rychlosti a prudkého brždění také počet najetých kilometrů a počet jízd za den. TD insurance, která je lídrem v pojišťovnictví v Kanadě, zrealizovala řešení ("TD MyAdvantage - Safe Driving Discount | TD Insurance," n.d.), během kterého je každé uskutečněné jízdě přiřazeno skóre. Výše pojištění je pak založena na celkovém skóre ze všech jízd. Řešení používá stejné proměnné jako předchozí zmíněné firmy a v současné době upozorňuje řidiče pouze při překračování rychlosti. Podobné služby už nabídly i další společnosti.

1.4 Rešerše dostupných publikací

Systémy založené na senzorech mobilního telefonu jsou zatím pouze předmětem vědeckých publikací. Výzkumníci (Yu et al., 2017) navrhli systém detekující abnormální chování řidičů v reálném čase. Tým shromažďoval reálná provozní data za 6 měsíců. Za použití parametrů prudkého přetížení a bořního zrychlení v zatáčkách, dosáhl tým velice přesných výsledků v klasifikaci. Při použití dat v delším časovém horizontu se může přesnost ještě zvýšit. Tým (Hu et al., 2017) provedl výzkum pomocí neuronových sítí. Tým využil nashromážděná reálná jízdní data k vyhodnocení abnormálního chování řidičů. Parametry jsou rychlost, zrychlování a brždění. Další práce zmiňuje i význam externích vlivů jako je hustota dopravy nebo dopravní regulace.

Další výzkumné týmy se zabývají diagnózou přímo na palubě vozu (On Board Diagnosis). Tým (Bergasa et al., 2014) vytvořil mobilní aplikaci pro iPhone, která využívá mobilních senzorů i senzorů vozu a analyzuje tato vstupní data v reálném čase. Program rozpozná neobvyklé chování a oznámí to uživateli. Celkově program rozeznává pouze normální a abnormální chování. V budoucnu hodlá tým navrhnout pojišťovací systém založený kromě jízdních dat i na zdravotním stavu klienta.

Různí výzkumníci přicházení s širokou škálou řešení, jak monitorovat a analyzovat vzory v chování řidičů. Nejčastěji využívanými proměnnými jsou rychlost, akcelerace, brždění a

boční přetížení. Použité algoritmy se různí stejně jako jejich přednosti a nedostatky. Nicméně klasifikační model stále není jednoznačně vyřešenou úlohou, přestože různé přístupy vykazují dobré výsledky.

Hlavní výzvou je princip, podle kterého se bude výše pojištění stanovovat. Existují dva hlavní druhy klasifikace.

V reálném čase – proud dat analyzován okamžitě stejně jako výše pojištění

Z uložených dat – nejprve dojde ke shromáždění dat za časové období a poté k jejich analýze

Toto je výčet pouze několika vědeckých publikací a komerčních aplikací, které se zabývají analýzou telemetrie a jízdních dat uživatelů. V současnosti jde o velký trend a tato práce jde tomuto trendu vstříc. V praktické části této práce dojde k přípravě a analýze telemetrických dat. Behaviorální faktor bude jediným zkoumaným faktorem. Cíl bude podobný, jako u zmíněných výzkumných prací; detekce vzorů v chování řidičů.

2 Nástroje

2.1 Stručný popis aplikace OneApp

Mobilní, multiplatformní aplikace OneApp, vyvinutá pro Škodu auto, měla ve své době za cíl sjednotit několik menších aplikací a dodatečných funkcí do jedné. OneApp nabídla uživatelům praktický infotainmentový nástroj a současně shromažďovala telematická data za účelem analýz. Uživatel musí souhlasit se zpracováním osobních údajů, pokud chce hlavní funkce používat. Předchozí dílčí funkce sice plnily úlohu infotainmentu, ale každá z nich pracovala samostatně. Nově navržená architektura umožňuje snadné přidávání nových funkcí, aby nebylo potřeba vyvíjet nové aplikace. Mobilní telefon lze připojit k palubnímu rozhraní pomocí USB sběrnice nebo technologie bluetooth.

Koncoví uživatelé (řidiči vozů) mají tímto k dispozici nový nástroj pro vytváření knihy jízd. Z uživatelského hlediska jde tedy o doplňující zařízení k původnímu infotainmentu palubního displeje. Aplikace může zobrazovat jízdní data na telefonu nebo přímo na palubním displeji pomocí sdíleného obrazu. Z hlediska vývojářského jde o řešení sběru jízdních dat i z těch automobilů, které nemají potřebnou výbavu k nepřetržitému připojení k internetu.

Celkové řešení ale přesahuje rámec aplikace OneApp. Jelikož jsou funkce aplikace dostupné až po přihlášení, je součástí řešení připojení k systému uživatelských účtů. Vazba na konkrétního uživatele (nikoliv na vůz) je zásadní pro klasifikaci řidičů podle jízdních vlastností. Součástí řešení je i back-end databázová aplikace pro shromažďování jízdních dat. Back-end slouží jak uživatelům pro vizualizaci předchozích jízd, tak firmě pro analýzu.

Aplikace funguje ve dvou módech. App mode je dostupný i bez propojení s autem a slouží ke sledování dat z již ukončených jízd. Uživatel je může vizualizovat různými způsoby v grafech nebo na mapovém podkladu. Car mode je dostupný pouze po propojení s vozem a jeho hlavním úkolem je komunikace s MIB modulem.

Aplikace samotná se skládá z jádra (core) a doplňujících nástaveb (add-on a widget). Jádro umožňuje vkládání nástaveb a zajišťuje komunikaci s externími zdroji jako je MIB, uživatelská a jízdní databáze. Pro vytváření dalších nástaveb bylo vyvinuto nástavbové API nad jádrem aplikace OneApp. Aplikace používá několik výchozích nástaveb, mezi které patří především sběr jízdních dat v reálném čase. Různé nástavby mohou sbírat jiná jízdní data a v jiných frekvencích. Nástavby mají rovněž přístup k čidlům přímo nesouvisející s jízdou (například stav nádrže). Nástavba typu widget je rozšíření uživatelského rozhraní aplikace. Umožňuje vizualizaci různých jízdních veličin.

Data jsou generována jako signály o vývoji jednotlivých veličin pomocí čidel rozmístěných v automobilu. Přenos dat zajišťuje sběrnice CAN. Data se shromažďují a zpracovávají v MIB modulu uvnitř automobilu. Jádro aplikace OneApp komunikuje s modulem a dotazuje se na měřené veličiny. Konkrétní nástavba definuje, které veličiny a v jaké frekvenci jsou dotazovány. Analýza se soustředí na nástavbu *drives*, která běží plošně u všech uživatelů. Kvůli potenciální náročnosti na datovou síť odešle jádro data na back-end až ve chvíli kdy je dostupné připojení k Wi-Fi.

2.2 Nástroje pro analýzu

Příprava a zpracování dat bude prováděna v programovacím jazyce Python 3.7 vyvíjeném jako open source projekt. Python je multiplatformní, vysokoúrovňový jazyk, u kterého je kladen důraz na čitelnost kódu a obecnou uživatelskou přívětivost pro člověka. Je založený na jádře jazyka C. Python podporuje mnoho programovacích paradigmat nejčastěji však objektově orientované. Filosofie jazyka předpokládala jednoduché jádro a vysokou míru rozšiřitelnosti. Díky tomu pronikl Python, za téměř 30 let své existence, do většiny programovacích sfér od webových aplikací po zpracování signálů a strojové učení. V současné době existuje celá řada rozšíření (modulů) pro datovou analýzu, statistiku a vytváření datových modelů.

Základní předpoklad používání je instalace interpreta zajišťujícího správné vykonávání programu. Konzole interpretu slouží k zadávání příkazů přímo. Interpret automaticky provede libovolný Python kód nebo spustí libovolný soubor s příponou py. Základní interpret obsahuje pouze několik desítek výchozích knihoven. Python sám nemá žádné vývojářské rozhraní.

Pro snadnější psaní aplikace jsem doplnil interpreta o vývojové prostředí PyCharm od firmy JetBrains s.r.o.. Firma nabízí studentskou licenci zdarma po poskytnutí platného univerzitního emailu (“Free Educational Licenses - Community Support,” n.d.). Edice professional verze 2018.3 je v tomto případě určena pouze pro vzdělávací účely. PyCharm lze provázat s jedním nebo více interprety. Přestože hlavní účel produktu PyCharm je vývoj webových aplikací, nalezne využití i pro vývoj aplikací desktopových. Nabízí široké možnosti nápovědy a korektury kódu, přehlednou správu projektů, skriptů a dalších souborů, snadnou instalaci knihoven a sadu nástrojů podporující vědecké analýzy. Především dynamické zobrazování dat a vizualizace pomocí grafů přímo uvnitř studia je velice praktické.

Nástroje pro datovou analýzu jsou implementovány v několika modulech. Modul NumPy (“NumPy Documentation,” n.d.) přidává podporu pro práci s vícerozměrnými daty a maticemi. Obsahuje velké množství matematických, logických, statistických a algebraických operací.

V přímé návaznosti na NumPy je důležitý modul SciPy ("SciPy — SciPy v1.4.1 Reference Guide," n.d.), který obsahuje větší množství funkcí v oblasti datových věd, matematické analýzy, zpracování signálů a počítačového vidění. Modul Scikit-learn ("sklearn.cluster.KMeans — scikit-learn 0.22.2 documentation," n.d.) pracuje s oběma knihovnami současně a výrazně zjednodušuje vytváření statistických modelů. Jeho prostřednictvím se vytváří klasifikační, regresní a shlukovací modely. Knihovna matplotlib ("Overview — Matplotlib 3.2.1 documentation," n.d.) slouží ke generování grafů a je rovněž integrovaná do předchozích knihoven. V průběhu práce budou použity jen některé funkce. Použití každé matematické funkce bude provázet název knihovny, ze které je volána a vzorec popisující matematický vztah, jenž reprezentuje.

2.3 Databáze

Přestože struktura dat není příliš složitá, a lze jí převést do relačního databázového modelu, byla vybrána NoSQL databáze, která zachová původní strukturu dat. MongoDB ("MongoDB Documentation," n.d.) je multiplatformní, dokumentový, databázový program. Je dostupný zdarma jako open source software v licenci GNU Affero General Public Licence. Oproti relační databázi umožňuje ukládat a pracovat s dokumenty formátu BSON. BSON je binární reprezentace formátu JSON ("BSON Types — MongoDB Manual," n.d.) a tudíž lze ukládat dokumenty aplikace OneApp v nezměněné struktuře.

Dokumentově orientovaný datový model představuje přirozenou datovou strukturu a zachová přehlednost dat. Dokumenty jsou flexibilní, lze je snadno upravovat v závislosti na potřebách aplikace i v situacích které by u SQL vyžadovaly přepracování relačního modelu. Dokumenty se v rámci jedné databáze třídí do kolekcí. V rámci kolekce není nezbytně nutné zachovávat schéma a umožňují libovolné rozšíření o páry klíč: hodnota nebo celá datová pole. Dokumenty umožňují snadné procházení hierarchií a hledání dat. Relační model by pro provedení i triviálních úloh vyžadoval operaci JOIN.

MongoDB je v základu desktopový program ale je dostupný i ve formě webové služby. V tomto případě bude sloužit pouze jako uložiště pro desktopovou aplikaci. Program lze instruovat pomocí příkazů do konzole nebo některého z dodatečných grafických rozhraní. Přesto, že není nezbytné, doinstaloval jsem rozhraní Compass. Dotazování pomocí rozhraní se však ukázalo jako pomalejší než pomocí konzole. Proto bude sloužit pouze k vizualizaci a přehledu.

MongoDB používá vlastní dotazovací a manipulační jazyk. Dovoluje užívání sekundárních indexů i agregačních funkcí. Lze tedy psát široké spektrum dotazů od jednoduchých po velice

složité. Rozhodl jsem se pro kompromis a používat dotazování k hledání, filtrování a základním operacím, zatímco složitější statistické modely budou prováděny v jazyce Python.

Doporučený způsob práce s Mongem pomocí jazyku python je modulem PyMongo. Modul obsahuje všechny důležité nástroje. Spuštěním mongod.exe se vytvoří instance MongoDB, která musí běžet na pozadí. Python se ní připojí pomocí výchozího portu.

3 Popis dat

3.1 Popis výchozího souboru

Pro diplomovou práci poskytla firma SKODA Auto a.s. datový soubor exportovaný z databáze OneApp o velikosti 5,67 gigabajtu v nekomprimovaném stavu. Data jsou organizována v hierarchické struktuře formátu JSON a vypovídají o jedné tisícovce uživatelů. Text je kódován pomocí utf-8. Soubor obsahuje 39 818 řádek, přičemž každý z nich je kompletní JSON objekt. Počet objektů je tedy roven počtu řádků. Objekty se dělí podle typu nástavby která je vygenerovala. Hlavními typy nástaveb jsou settings, cars, drives. Objekty settings vypovídají o nastavení uživatelského rozhraní a nebudou k analýze využity. Objekty cars specifikují k aplikaci připojený vůz nebo data, která se k němu vztahují (např. záznamy o tankování). Objekty drives jsou hlavním zdrojem dat. Obsahují naměřené vzorky a obecné statistické veličiny v rámci jízdy. Celkem soubor obsahuje 2083 objektů settings, 4333 specifikací vozů a 33 261 jízd.

JSONy sdílí velice podobnou strukturu, která se liší jen v atributech na nižších úrovních hierarchie. Mateřský objekt (neboli pole dvojic klíč: hodnota) obsahuje unikátní identifikátor, identifikátor nástavby a identifikátor uživatele. Unikátní id typu string je univerzální pro všechny typy JSONů. Id nástavby je rovněž string a zkratkou popisující stát, pro který je verze aplikace navržena, platformu, na které aplikace běží a název nástavby. Dalšími klíči jsou; platforma samotná, verze aplikace, identifikátor dokumentu, částečný uživatelský klíč, čas poslední úpravy dokumentu a identifikátor uživatele. Id uživatele se váže k systému uživatelských účtů nad rámec aplikace OneApp. Příklad vypadá takto:

```
{
  {"id":"6...6#cz.eman.android.oneapp.lib.addon.drives#92233704
  70471679791",
  "addonId":"cz.eman.android.oneapp.lib.addon.drives",
  "platform":"Android",
  "version":{"$numberInt":1},
  "itemId":"92233704704716791791",
  "partialUserKey":"6...6#cz.eman.android.oneapp.lib.addon.driv
  es#",
  "updated":{"$numberLong":1566383096016},
  "userId":"6...6"
    "content": {…
     "data": {…}
      }
}
```


Klíč content odkazuje na první vnořenou úroveň. Na této úrovni nejsou žádné atributy společné pro všechny typy nástaveb. Výjimkou je klíč data, který obsahuje vlastní hodnoty jednotlivých nástaveb.

3.1.1 Specifikace vozů

Objekt generovaný doplňkem cars popisuje auto v několika proměnných. Dokumenty tohoto typu nejsou pro každý vůz unikátní. Generují se na základě uživatelských vstupů. Pokud uživatel do aplikace zadá informace o tankování nebo technické kontrole, vytvoří se nový dokument. Klíče popisuje následující tabulka.

Název klíče	Popis
`engineTypePrimary`	Primární typ paliva motoru
`engineTypeSecondary`	Sekundární typ paliva pro vozy upravené na CNG
`maxPower`	Maximální výkon v kilowattech
`serviceInspectionDistance`	Počet kilometrů do další technické kontroly
`serviceInspectionDistanceState`	Stav, zda motor vyžaduje technickou kontrolu podle ujeté vzdálenosti (podle typu vozu každých 30 nebo 50 tisíc kilometrů)
`serviceInspectionTime`	Čas do další technické kontroly
`serviceInspectionTimeState`	Stav, zda motor vyžaduje technickou kontrolu podle času od poslední kontroly
`serviceOilDistance`	Počet kilometrů do výměny oleje
`serviceOilDistanceState`	Stav, zda motor vyžaduje výměnu oleje podle ujeté vzdálenosti
`serviceOilTime`	Čas do další výměny oleje
`serviceOilTimeState`	Stav, zda motor vyžaduje výměnu oleje podle času od poslední výměny
`tankLevelPrimary`	Stav nádrže
`totalDistance`	Stav tachometru v kilometrech
`vehicleType`	Model automobilu
`vin`	Identifikační číslo vozidla

Tabulka 1: Klíče a jejich popis v dokumentech cars na úrovni data

Hodnoty v uvozovkách jsou datového typu string. Časy a vzdálenosti a výkony jsou typu integer. Čas je vždy formátu UNIX timestamp. Vzdálenosti jsou v kilometrech. Stav nádrže je typu float mezi 0.0 (prázdná) a 1.0 (plná). Příklad obsahu cars vypadá následovně:

```
"data": {
  "engineTypePrimary": "PETROLGASOLINE",
  "engineTypeSecondary": "NOTINSTALLED",
  "maxPower": 92,
  "serviceInspectionDistance": 0,
  "serviceInspectionDistanceState": "NO_DATA",
  "serviceInspectionTime": 0,
  "serviceInspectionTimeState": "NO_DATA",
  "serviceOilDistance": 0,
  "serviceOilDistanceState": "NO_DATA",
  "serviceOilTime": 0,
  "serviceOilTimeState": "NO_DATA",
  "tankLevelPrimary": 0.5700000000000001,
  "totalDistance": 46667,
  "vehicleType": "RAPIDSPACEBACK",
  "vin": "T...9",
  "visible": true
},
```


Primární funkcí doplňku cars je informování uživatele o stavu vozu a případně ho upozorňovat na servisní kontroly. Jelikož je funkce závislá na uživatelských vstupech, pro spolehlivou analýzu se nehodí. Objekty poslouží pouze ke klasifikaci jízd podle typů motorů a modelů.

3.1.2 Jízdy

Objekty drives jsou hlavním nositelem informace o průběhu jízdy. Jsou řádově mnohem větší než cars a tvoří přes 80% souboru. Zbytek tvoří dokumenty typu cars a settings. Průměrná velikost souboru je 150,6 KB. Pole pod klíčem data obsahuje větší množství proměnných popisující jízdu agregovanými hodnotami. Jedna skupina proměnných udává průměrné hodnoty za jízdu, další nejvyšší naměřené za jízdu. Skupiny doplňuje několik samostatných proměnných. Čas je vždy ve formátu UNIX, boční a dopředná zrychlení jsou vždy násobky gravitačního zrychlení, rychlost je v kilometrech za sekundu.

Název klíče	Popis
avgConsumptionPrimary	Průměrná spotřeba l/100 km
avgConsumptionSecondary	Průměrná spotřeba kg/100 km pro CNG motory
avgEngineSpeed	Průměrné otáčky za minutu
avgLeftG	Průměrné přetížení doleva (násobek gravitačního zrychlení)
avgOutputPower	Průměrný výkon v kilowattech
avgRightG	Průměrné přetížení doprava (násobek gravitačního zrychlení)
avgVehicleSpeed	Průměrná rychlost v km/s
comment	komentář
defaultType	Výchozí typ jízdy: služební/soukromá
driveCostPrimary	Cena jízdy je součástí služby pro uživatele. Je dopočítána na základě spotřeby a ceny natankovaného paliva, pokud je uživatel zadá.
driveCostSecondary	Cena jízdy pro CNG motory (měna jakou uživatel nastaví)
driveTime	Doba jízdy (počet vteřin)
endLocation	Adresa a země kde byla jízda ukončena
endTime	Čas ukončení jízdy (UNIX)
isMib	Potvrzení přítomnosti a připojení MIB
maxConsumption	Maximální spotřeba l/100 km
maxConsumptionTime	Čas, ve kterém spotřeba dosáhla maxima
maxEngineSpeed	Maximální otáčky za minutu
maxEngineSpeedTime	Čas kdy otáčky dosáhly maxima
maxFrontAcc	Maximální přetížení (násobek gravitačního zrychlení)
maxFrontAccTime	Čas kdy přetížení dosáhlo maxima

Název klíče	Popis
maxLeftAcc	Maximální boční přetížení (násobek gravitačního zrychlení)
maxLeftAccTime	Čas kdy boční přetížení vlevo dosáhlo maxima
maxOutputPower	Maximální výkon v kilowattech
maxOutputPowerTime	Čas, ve kterém motor dosáhl maximálního výkonu
maxRearAcc	Maximální retardace (násobek gravitačního zrychlení)
maxRearAccTime	Čas kdy retardace dosáhla maxima
maxRightAcc	Maximální boční zrychlení vpravo
maxRightAccTime	Čas kdy boční přetížení vpravo dosáhlo maxima (násobek gravitačního zrychlení)
maxVehicleSpeed	Maximální rychlost km/s
maxVehicleSpeedTime	Čas, ve kterém vůz dosáhl maximální rychlosti
startLocation	Adresa a země kde byla jízda zahájena
startTime	Čas zahájení jízdy
totalDistance	Ujetá vzdálenost v km
totalDistanceLast	Stav tachometru po jízdě v km
totalDistanceStart	Stav tachometru před jízdou v km
type	Manuálně zvolený typ jízdy: služební/soukromá
vin	Unikátní identifikátor vozu

Tabulka 2: Klíče a jejich popis v dokumentech drives na úrovni data

Klíč mapData, umístěn na stejné úrovni, obsahuje pole naměřených vzorků seřazených podle času. V každém vzorku se měří třináct proměnných.

Proměnná	Popis
consumption	Spotřeba v l/100 km
efficiency	Efektivita (0–100)
fontAcc[1]	Zrychlení (G – násobek gravitačního zrychlení)
latitude	Zeměpisná šířka
leftG	Boční zrychlení doleva (G – násobek gravitačního zrychlení)
longitude	Zeměpisná délka
outputPower	Výkon (kW)
refuel	Indikace tankování (true/false)
rightG	Boční zrychlení doprava (G – násobek gravitačního zrychlení)
rpm	Otáčky za minutu
speed	Rychlost (km/s)
stopped	Vůz ve stavu klidu (true/false)
time	Čas ve formátu UNIX

Tabulka 3: Klíče a jejich popis v dokumentech drives na úrovni mapData

Proměnná efektivita je produktem matematického vzorce, jenž využívá vícero signálů včetně těch, které se do databáze neukládají. Tento vzorec ale Škoda Auto a.s. neposkytla, ani není popsán v dokumentaci. Během konzultací jsem zjistil jen několik informací. Efektivita nemůže dosáhnout maxima při rychlosti nad 80 km/h. Proměnná zahrnuje plynulost sešlápnutí pedálu a dodržování rychlostního stupně doporučené palubním počítačem. Jelikož nelze přesně určit význam této proměnné, nebude v analýze zahrnuta.

3.1.3 Sběr dat v průběhu jízdy

V průběhu jízdy se jádro aplikace OneApp dotazuje MIB jednotky na okamžité hodnoty proměnných. Interval dotazování není pokaždé stejný. Sbíraná data tedy nemají stabilní vzorkovací interval. Na podnět vedoucího vývoje aplikace byl vzorkovací interval upravena tak, aby se přizpůsobila rychlosti jízdy vozu. Důvodem bylo vytváření nadměrně velkých

[1] U akcelerace pravděpodobně došlo k překlepu; namísto „font" budu používat „front", tedy dopředná akcelerace.

souborů s vysokou redundancí. Každý takový soubor byl uložen v paměti telefonu, dokud nebylo dostupné připojení k Wi-Fi a docházelo tak k vyžádání velké části paměti. Úprava spočívala v implementaci proměnlivého intervalu přímo úměrné rychlosti. Tato hypotéza bude dále v práci ověřena. Je-li vůz v klidovém stavu, tj. motor běží, ale vůz stojí, měří se vzorek pravidelně každých 10 sekund. V nižších rychlostech (a pravděpodobně při průjezdu městem) je zvyšuje frekvence periodických jevů jako je brždění a opětovné rozjíždění, vytáčení motoru a řazení. Alespoň částečné zaznamenání těchto jevů si vyžaduje krátkou vzorkovací periodu. Naopak ve vyšších rychlostech (na dálnicích a rychlostních silnicích) je jízda plynulejší a krátký interval měření je nadbytečný.

Pro správné zahájení jízdy platí tyto předpoklady:

- Nainstalovaná aplikace OneApp
- Registrace a přihlášení uživatele do aplikace
- Zapnuté připojení přes Wi-Fi
- Propojený telefon s autem (modulem MIB) pomocí USB nebo bluetooth
- Uživatel má zapnuté sledování polohy pomocí GPS

Měření jízdy je zahájeno nastartováním vozu a přijetím příslušného signálu z MIB. Není-li dostupný, je pro zahájení jízdy použit první signál o změně rychlosti. K automatickému ukončení měření dojde po vypnutí motoru příslušným signálem nebo posledním signálem o rychlosti. Jízda se ukončí také dvě hodiny po ztrátě signálu, vypnutí telefonu nebo přerušení spojení s MIB. V těchto případech může docházet k nekonzistenci dat z MIB. Pokud uživatel ukončí Car mode, záznam se rovněž ukončí. V určitých situacích dochází k dočasnému přerušení měření. Dochází k tomu při jakémkoli přerušení spojení mezi aplikací a MIB. Při přestávkách kratších než 10 minut dojde k automatickému navázání na předchozí jízdu. Pokud je jízda přerušena na méně než dvě hodiny, aplikace zobrazí dialog a dotáže se uživatele, jestli jí chce ukončit nebo pokračovat ve předchozí. Pokud trvá přerušení déle než dvě hodiny, automaticky dojde k ukončení jízdy a zahájení nové.

4 Příprava dat

Pro jakoukoliv analýzu je nezbytné data upravit tak, aby umožnila spolehlivé nasazení statistických nástrojů a následné vytváření modelů. Dokument před vložením do databáze projde základní přípravou v aplikační vrstvě. Další přípravné kroky jsou provedeny uvnitř databáze buďto přímo dotazovacím jazykem MongoDB nebo jsou dokumenty načteny do aplikační vrstvy, upraveny a poté aktualizovány v databázi.

4.1 Příprava před vložením do databáze

Aby nemusel být celý dokument nahráván do operační paměti, postupuje se při jeho čtení iterativně po řádcích. Obsah řádku se vždy dekóduje pomocí modulu *json*. Jde o pomalou ale nezbytnou operaci. Objekt je tímto převeden do datové struktury dict (slovník). Nejdřív je potřeba rozlišit doplňky, jimž objekty náleží. Aplikace se výchozím portem připojí k databázi pomocí modulu PyMongo. Po připojení se vytvoří výchozí databáze s názvem „OneApp“ obsahující dvě kolekce „Drives“ a „CarSpec“. První bude využita k ukládání jízd, druhá k ukládání specifikací vozů. Po přípravných operacích funkce *pymongo.insert_one()* automaticky převede formát dict do formátu BSON a uloží dokument do příslušné kolekce.

4.1.1 Specifikace

Do databáze se uloží pouze dokumenty s unikátním VIN kódem a duplikace budou zahozeny. Z kapitoly popisující formát dat je zřejmé, že dokumenty obsahují spoustu redundancí. Přípravné kroky jsou prováděny pomocí funkce *prepare_json_spec(line)*. Klíče partialUserKey, addonId jsou odstraněny stejně jako klíče vztahující se ke stavu oleje, nádrže nebo vozidla. Rovněž verze aplikace je odstraněna, jelikož je u všech jízd totožná. Obsah klíče itemId je použit jako primární identifikátor. Ke klíči identifikátoru je přidáno podtržítko. MongoDB ho tak rozpozná jako primární identifikátor. Některé klíče obsahují namísto hodnoty další vnořený klíč, popisující datový typ. Například:

```
"driveTime":{"$numberInt":797617}
"endTime":{"$numberLong":1544780981277}
```

MongoDB neumožnuje klíčům začínat znakem “$” a jelikož mám díky dokumentaci přehled o datových typech, je nadbytečné tento název klíče uchovávat. Rovněž tím dojte ke sjednocení všech dokumentů cars na jednotný formát.

4.1.2 Jízdy

Přípravu dokumentů typu drives provádí funkce funkce *prepare_json(line)*. Stejně jako u specifikací, je obsah proměnné itemId přesunut do primárního klíče (id jízdy) a původní klíč je odstraněn. Proměnné addonId a partialUserKey jsou také odstraněny. I v těchto dokumentech se objevují nadbytečné klíče definující datový typ. Pokud se v úrovni data objeví, jsou odstraněny.

Podle stejného principu se ošetřují a upravují proměnné v jednotlivých vzorcích na úrovni mapData. Mnoho vzorků neobsahuje všechny klíče a hodnoty. V rámci stejné for smyčky, v níž probíhá i odstranění názvů datových typů, dochází i k doplnění vzorku na všech třináct proměnných. Výchozí hodnota je prozatím „null“. Doplnění klíčů má dva hlavní přínosy. Lze snadno určit počet validních vzorků v jízdě. Tedy takových, které mají všechny hodnoty rozdílné od „null“. Pokud jsou hodnoty vyjmuty a uloženy do polí, zůstane délka polí identická. Mnoho matematických a vizualizačních funkcí předpokládá stejnou délku vstupních polí (vektorů).

Rychlost je vynásobena konstantou 3600 (počet sekund za hodinu) a dojde tak k převodu z km/s na km/h. Rychlost je následně zaokrouhlena na dvě desetinná místa. Hodnoty času jsou všechny typu long. Hodnota unixového času nabyla desátého řádu v roce 2001. Současný unixový čas má rovněž deset řádů. Čas v souboru má třináct, tudíž je třeba ho převést na double a dělit tisícem. Přesnost na desetinná místa je u vzorkování podstatná.

Celkem bylo upraveno a vloženo do databáze 682 unikátních specifikací vozů a 33 260 jízd. Celková velikost specifikací je 338 kB, průměrná velikost činí 495,6 B. V úvodní fázi přípravy byl pouze jeden dokument jízdy vyhodnocen jako nevalidní. Celková velikost všech jízdních souborů je 4,3 GB, průměrná velikost je 129,4 KB. Od původního souboru tedy došlo díky odstranění klíčů k úspoře 1,3 GB.

4.2 Příprava dat uvnitř databáze

Po vložení do databáze je anonymizace dat snadno proveditelná. Jelikož jsou specifikace a jízdy ve vzájemné relaci, je potřeba pracovat s oběma kolekcemi současně. Z každé specifikace je vybrán VIN kód vozidla a podle něj jsou v kolekci jízd nalezeny všechny jízdy, podniknuté tímto vozem. Uvnitř všech vybraných dokumentů je pak VIN kód přepsán. Jako náhradu jsem použil číslování od nuly. Stejným postupem jsou přepsány i identifikátory uživatelů. V práci budu místo VIN kódu používat spojení „identifikátor vozu“.

V průběhu příprav docházelo k postupnému hromadění nezbytných operací. Tyto operace je potřeba provádět atomicky a ve správném pořadí. Nebude zde popsán celý průběh hromadění chyb a jejich napravování. Kapitola pouze popíše všechny funkce a odůvodnění jejich použití. Filtrace a příprava je soustředěna v těchto bodech:

- Celkový počet vzorků a počet validních vzorků v jízdě
- Vzorkovací frekvence a prodlevy v měření
- Převzorkování

Jednodušší operace uvnitř databáze je vhodné provádět pomocí MongoDB dotazovacího jazyka. Tento způsob je jednodušší a rychlejší oproti převádění dokumentů zpět do Pythonu. Krátká, relevantní, testovací jízda o délce 20 kilometrů, a zahrnující jízdu městem, mimo město i po dálnici, se skládá z několika stovek vzorků. Jízdy v řádu desítek vzorků tedy nejsou relevantní a budou z databáze odstraněny. Počet dokumentů s požadovanou délkou je 27148. Počet se získá jednoduchým dotazem:

```
collectionDrives.count_documents(
{"content.data.mapData.99"}, {"$exists": "true"})
```

Jejich odstranění z databáze provede následující skript:

```
collectionDrives.delete_many(
{"$where": "this.content.data.mapData<100"})
```

6112 jízd je odstraněno. Podobným dotazem bylo potvrzeno, že všechny jízdy jsou měřeny pomocí MIB modulu. Mnoho přípravných operací je založeno na závislostech předchozích a následujících indexů v datových polích. Tyto operace předpokládají seřazení vzorků podle času. V průběhu příprav se však ukázalo, že ne vždy tomu tak u všech vzorků je. Je proto vhodné seřadit vzorky na úplném počátku příprav tímto skriptem:

```
collectionDrives.update_many({},
{"$push":
        {"content.data.mapData":{"$each":[],
                                 "$sort": {"time":1}
                                 }
        }
})
```

4.2.1 Ošetření chybějících hodnot

Jak bylo popsáno výše, každý vzorek je doplněn o klíče, které mu chybí s výchozí hodnotou "null". Nejprve je pro přehled spočteno, jak vysoký podíl "null" hodnot jednotlivé jízdy mají. Následující dvě tabulky obsahují počty jízd, ve kterých se hodnoty "null" podílejí na vzorcích deseti a dvaceti procenty.

Počet proměnných, kde hodnota "null" přesahuje podíl 10%	Počet jízd
0	16 816
1	3030
2	7544
3	1360
4	815
5	471
6	519
7	163
8	117
9	0
10	78
11	0
12	0
13	0

Tabulka 4: Počty jízd a proměnných s podílem „null" hodnot nad 10%

Počet proměnných, kde hodnota “null” přesahuje podíl 20%	Počet jízd
0	18 647
1	2469
2	7641
3	805
4	748
5	152
6	266
7	66
8	52
9	0
10	67
11	0
12	0
13	0

Tabulka 5: Počty jízd a proměnných s podílem „null“ hodnot nad 20%

Celá polovina jízd obsahuje 90% validních hodnot u všech proměnných. Takové jízdy bude snadné ošetřit. Mezi standardní metody ošetřování chybějících hodnot patří doplňování mediánem nebo aritmetickým průměrem, okopírování hodnoty z jiného záznamu a použití prediktivního modelu. První dvě metody ale nelze použít, neboť veličiny mají v realitě spojitý průběh a vzájemnou závislost. Doplnění hodnot průměrem nebo mediánem by způsobilo nežádoucí zkreslení. Model, který by na základě ostatních hodnot vzorku našel jiný, podobný vzorek a použil jeho hodnotu, lze považovat za relevantní. Pokud by však docházelo k používání vzorků napříč jízdami, je třeba vybírat ze stejných typů vozů a motorizací. Další možností je lineární interpolace, za předpokladu, že předchozí i následující hodnoty jsou známé.

Počty validních jízd se u obou tolerancí zásadně neliší. Jízdy, které mají u pěti a více proměnných podíl “null” hodnot větší než 20 % jsou odstraněny. Takových jízd je pouze pár set, obvykle mají jen nízký počet vzorků. Zeměpisná poloha chybí především u jízd, kde mají nedostatek vzorků právě dvě proměnné. Přestože mobilní telefon nesdílí svou polohu, ostatní veličiny se měří. Pokud kromě polohy nebudou žádné jiné veličiny postrádat významnou část vzorků, lze tyto jízdy do analýzy zahrnout.

Na základě podmíněného filtrování a vizualizace jsem vypozoroval několik opakujících se trendů.

- 8187 jízd neobsahuje zeměpisnou polohu
- Spotřeba nabývá hodnoty "null", pokud se vozidlo nepohybuje
- Vypne-li uživatel motor, nabývají hodnoty "null" proměnné spotřeba, efektivita, výkon, akcelerace a otáčky

Kromě zeměpisné polohy často chybí také okamžitý výkon motoru. Význam této veličiny v klasifikačním modelu ještě není úplně jasný, největší význam předpokládám u rychlosti, spotřeby, otáček a přetížení. Dokumentace se o tomto naprogramování nezmiňuje, přesto se tento trend projevuje fakticky ve všech jízdách. S přihlédnutím k nim je možné navrhnout metody ošetření dat bez velkých datových ztrát. První funkce *delete_begin_end(document)* odstraní všechny vzorky se spotřebou "null" na počátku a na konci. Jde tedy o dvě *while* smyčky, které mají za úkol odstranit vzorky, které byly naměřené před tím, než se vozidlo poprvé začalo pohybovat, a naopak od chvíle kdy se naposledy zastavilo. Funkce po úpravě předá dokument následující funkci.

V rámci zjednodušení a úspor paměti i operací, je provedeno sjednocení bočních přetížení do jedné proměnné. U většiny dokumentů je přetížení vlevo vždy záporná hodnota a vpravo vždy kladná. V takovém případě je hodnota nové proměnné "sideG" rovna součtu obou původních. Významnější hodnoty může nabývat pouze jedna ze stran a druhá je tudíž nepodstatným šumem. Existují ale i dokumenty, kde hodnoty nabývá pouze jedna strana a druhá má automaticky hodnotu "null". I v těchto dokumentech se zachovává znaménko (tedy směr) a stačí tedy použít jednu nebo druhou.

Následující funkce *handle_nulls_in_motion(document)* lineárně interpoluje všechny hodnoty ve všech vzorcích, pokud splňují patřičné podmínky. Ke každému vzorku se přistupuje jednotlivě. V prvé řadě nesmí být předchozí ani následující vzorek vzdálenější v čase než 10 sekund (dopočteno rozdílem časů). Pro každou z proměnných je nastavena hranice, jak velký rozdíl může být mezi předchozím a následujícím vzorkem. Funkce tedy ošetřuje jen velice plynule probíhající jízdy. Rozdíl efektivity může být maximálně 20, rozdíl otáček 400 otáček za minutu, rozdíl rychlosti 30 km/h , rozdíl dopředného přetížení 0,3 G a bočního 0,2 G, rozdíl výkonu 50 kW, rozdíl spotřeby 5 litrů na 100 km. Jsou-li podmínky splněny, je volána funkce *interp1d* z modulu *scipy.interpolate*, která lineární interpolaci provede. Funkce opět předává dokument další.

Veličiny měřené pomocí senzorů a čidel nabývají často extrémních hodnot. Například dopředné přetížení v realitě u masově vyráběného vozu nemůže překročit 1,5 G. Avšak kvůli citlivosti akcelerometru dochází k občasnému zaznamenávání extrémně vysokých hodnot např. 20 G. K

podobným extrémům dochází rovněž u bočního přetížení a spotřeby. Pro dopředné i boční přetížení je nastaven přípustný interval od –1,7 do 1,7 G. Spotřeba může být maximálně 70 litrů na 100 kilometrů. Pokud hodnota překračuje tyto stropy, je nahrazena nejvyšší naměřenou hodnotou v jízdě, která není extrém.

Dále je třeba ošetřit chybějící spotřebu během stání. Lineární interpolace může doplnit jen jednotlivé vzorky. Pokud však automobil zastaví například na semaforu na delší dobu, může se naměřit větší počet vzorků bez spotřeby. Funkce *handle_consumption_nulls(document)* se nejprve ujistí že chybí pouze spotřeba a rychlost není vyšší než 3 km/h. Pokud ano, je spotřeba doplněna pseudonáhodným číslem v intervalu od 1 do 2 včetně, které odpovídá spotřebě motorů na volnoběh.

Pokud se budu držet předpokladů popsaných výše, lze v této fázi všechny vzorky, u kterých chybí současně a pouze celá pětice proměnných, odstranit nebo přepsat na nuly. Jejich účel v analýze je ale bezvýznamný a smazání je logičtější. Úprava probíhá opět iterativně po jednotlivých dokumentech pomocí příkazu update.

4.2.2 Vzorkovací interval a prodlevy v měření

Pro výpočet vzorkovacího intervalu jsem použil proměnnou time. V rámci jízdy je mezi každými dvěma vzorky spočten rozdíl časů a uložen do pole. Z pole je pomocí funkcí *mean()* a *stdev()* z knihovny *statistics* dopočten aritmetický průměr a směrodatná odchylka. Průměr intervalů u testovací jízdy je 3,7 s. Pokud vyjmu prodlevy rovny 10 (vůz stojí), pak je průměr 3.4 s. Směrodatná odchylka je v prvním případě 4,0 s a v druhém 2,1 s.

Vzorkování a prodlevy v měření jsou zásadním problémem a druhým hlavním předmětem přípravy dat. Prodlevy, ať už způsobené výpadkem signálu nebo záměrným navázáním měření po pauze, potenciálně způsobí značná zkreslení v datovém modelu. Současně brání nezbytnému převzorkování. Čisté jízdy bez prodlev ale tvoří pouze třetinu z celkového počtu jízd. Tudíž je zapotřebí upravit jízdy tak, aby bylo možné využít relevantní počet.

První možnost spočívá v rozdělení jedné jízdy na dvě za předpokladu, že obě dílčí jízdy mají dostatečný počet vzorků (tj. 100 a více). Druhou možností je odstranění části vzorků oddělených od zbytku měření delší časovou prodlevou. Jako ideální se jeví kombinace obou metod.

Tyto operace jsou prováděny v jazyce Python. Jde o pomalejší ale snazší a flexibilnější variantu oproti dotazovacímu jazyku MongoDB. Nejprve je vhodné pro přehled zjistit počet dokumentů s prodlevami a počty prodlev v nich. Standardní prodleva obvykle nepřekračuje deset vteřin.

Může však dojít k drobným odchylkám a jako dolní práh prodlevy je tedy nastaveno 15 sekund. Pro tyto operace je nadefinována funkce *delete_frequency_gaps(drive)*. Funkce zpracovává jednu jízdu a je volána pro každou zvlášť. Nejprve jsou spočteny prodlevy mezi vzorky a počet prodlev nad patnáct vteřin. Funkce vrací počet prodlev, které jízdu rozdělují, a počty vzorků v dílčích jízdách. Mateřský skript volá postupně dokumenty z databáze a pomocí výstupů ze zmíněné funkce sčítá dokumenty se stejným počtem prodlev a počty použitelných dílčích jízd (tj. nad 100 vzorků). K přehledu slouží následující tabulka.

Počet prodlev nad 15 sekund	Počet jízd	% z celkového počtu jízd nad 100 vzorků
Bez prodlevy	9292	34,2
1	4577	16,8
2	6046	22,2
3	2194	8,0
4	1896	6,9
5	949	3,4

Tabulka 6: Počty prodlev, jízd a jejich procentuální podíl na počtu všech jízd nad 100 vzorků

Jízdy s jednou a dvěma prodlevami a jízdy bez prodlev dohromady tvoří více než dvě třetiny všech jízd. Pokud by došlo k ošetření i jízd se třemi prodlevami, zůstalo by zachováno 81% původního počtu jízd. Dílčí jízdy musí mít rovněž alespoň 100 vzorků. Další tabulka je vyplněna počty vhodných, dílčích jízd.

Počet prodlev	Počet dílčích jízd nad 100 vzorků					
1	3240	2439				
2	4926	1014	1366			
3	1298	893	507	559		
4	1112	329	942	244	333	
5	532	248	308	383	165	200

Tabulka 7: Počty prodlev a dílčích jízd nad 100 vzorků

Rozdělením dojde k odstranění časově odlehlých a ojedinělých měření, která by způsobila chyby během převzorkování. Rovněž další, nerelevantní krátké jízdy (přeparkování, popojíždění) budou odstraněny. Zbytek jízd bude uloženo v oddělených dokumentech. Identifikátor dokumentu bude pozměněn tak, aby bylo možné zpětně dohledat, které dílčí jízdy tvoří jízdu původní. Ujistil jsem se, že třináctý řád identifikátoru převedené na integer je u všech jízd nulový. Tento řád se u rozdělených jízd doplní řadovým číslem. Identifikátory tak zůstanou unikátní a na třináctém řádu budou obsahovat číslovku popisující, o kolikátou dílčí jízdu se jedná. Nejvyšší číslo ale neodpovídá počtu dílčích dokumentů, které vzešly z jedné jízdy. Pokud délka dílčí jízdy nebyla dostatečná, dokument byl zahozen.

Většina obecných hodnot pro jízdu je u dílčích jízd vynulována. Tyto hodnoty nebudou předmětem analýz. Je důležité zachovat pouze unikátní identifikátor vozu. Rovněž typ jízdy (soukromá, služební), platforma a identifikátor uživatele jsou jednoznačně totožné.

Počet oddělených dokumentů se zvýšil na 30 881, nicméně celková velikost dat klesla na 3,1 GB. Rovněž průměrná velikost dokumentu klesla ze 156 KB na 99 KB. V kolekci převažují kratší jízdy. Třetina jízd obsahuje méně než 200 vzorků. To odpovídá krátkým jízdám na vzdálenosti několika kilometrů. Druhá třetina jízd obsahuje mezi 200 a 500 vzorky, tedy přibližně nižší desítky kilometrů. Třetí třetina jízd se pohybuje v rozmezí 500 a více vzorků. Poslední interval odpovídá vyšším desítkám až stovkám kilometrů.

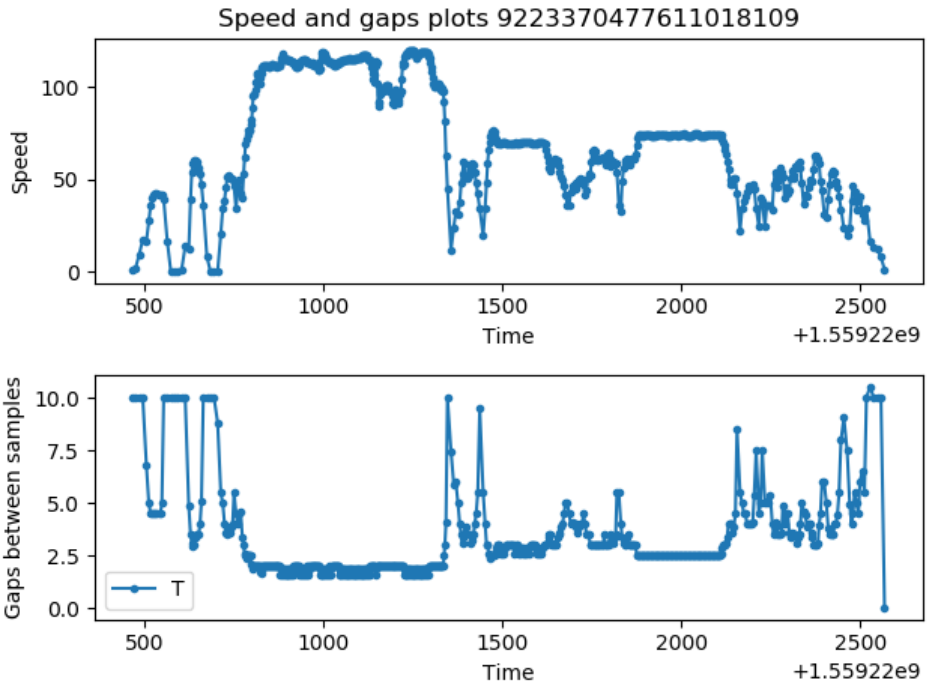
Oddělené dokumenty jsou uloženy do nové kolekce collectionDrivesDivided. Je důležité zajistit, aby šlo v případě chybování nebo ztráty dat, rychle vygenerovat nové jízdy z původní kolekce.

4.2.3 Hypotéza o nepřímé úměrnosti intervalu vzorkování a rychlosti

Během konzultace s Ing. Davidem Židem byl přijat návrh na otestování hypotézy o vzorkovacím intervalu. Jak již bylo zmíněno, vzorkovací interval by měl klesat se snižující rychlostí a naopak. K testování bude využito pole rychlosti a pole intervalů doplněné o jednu nulu na konci tak, aby byly délky polí stejné. Porovnávání hodnot vzorků a intervalů mezi nimi sice není nejpřesnější, ale v tomto případě nejjednodušší a dostačující. Pearsonův korelační koeficient napoví, zda mezi jednotlivými veličinami existuje korelace. Jeho vzorec vypadá následovně:

$$\rho_{X,Y} = \frac{E(XY) - E(X)E(Y)}{\sqrt{E(X^2) - E^2(X)}\sqrt{E(Y^2) - E^2(Y)}}$$

Kde E(X) je střední hodnota veličiny. Za předpokladu že jsou druhé mocniny náhodných veličin X a Y konečné, potom je korelace definována jako kovariance vydělená směrodatnými odchylkami obou proměnných. Hodnota korelačního koeficientu 1 značí přímou závislost (korelaci) mezi veličinami. Hodnota –1 značí nepřímou závislost (antikorelaci). K posouzení bude vybráno několik náhodných jízd bez velkých prodlev v měření. Pearsonův korelační koeficient je spočten pomocí funkce *pearsonr* z knihovny *scipy.stats.stats*. Jako parametry jsou funkci předány obě zmíněná pole.



Graf 1: Rychlost a vzorkovací interval v čase

U jízdy, která je vizualizována jako příklad, nabývá pearsonův korelační koeficient hodnoty

-0.8. Vypovídá o přesvědčivé antikorelaci mezi rychlostí a délkami intervalů mezi měřeními. Délka intervalu a rychlost jsou v nepřímé úměrnosti. V grafu lze rovněž pozorovat kratší intervaly současně s vyšší rychlostí. Okolo 50 km/h se ale začínají intervaly prodlužovat a přibližovat deseti vteřinám. Devět dalších, náhodně vybraných jízd bylo podrobeno stejnému testu. U pěti z devíti jízd nabývá koeficient hodnoty rovné nebo vyšší -0.8. Koeficienty u zbylých čtyřech jízd se pohybují mezi -0,67 a -0,76. Hypotéza je tímto vyvrácena.

Vyvrácení hypotézy má několik důsledků, které je třeba popsat, než bude přikročeno k dalším fázím přípravy. Z dat nelze spolehlivě zjistit, jakým stylem se uživatel rozjíždí a jaké jsou jízdní vlastnosti v nízkých rychlostech. V přímé návaznosti je i druhý důsledek; jízdy nelze převzorkovat na několikanásobně nižší interval bez výrazného zkreslení dat v nižších rychlostech.

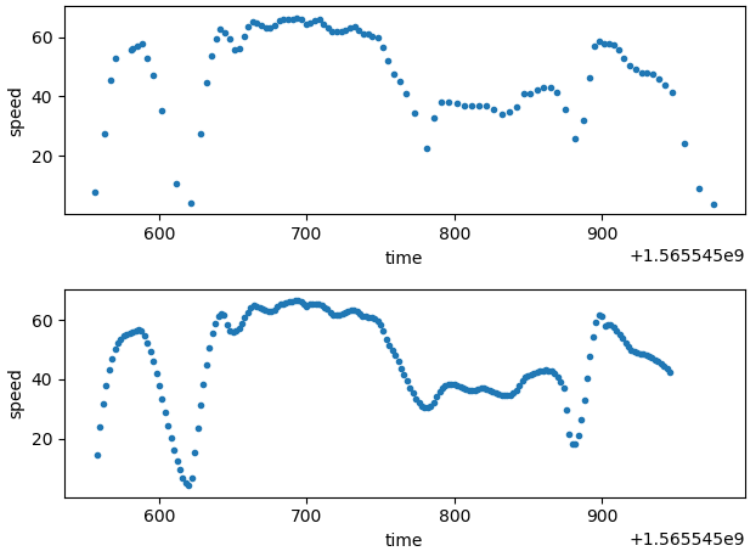
4.2.4 Kubická interpolace a převzorkování

Pro úplné narovnání podmínek pro všechny validní jízdy, je nezbytné převzorkování na totožnou vzorkovací periodu. Nejprve je pro čtveřici vzorků nalezena polynomiální funkce protínající všechny body. Do výsledné funkce je poté dosazeno a jsou dopočteny nové vzorky. Funke se nazývá *def resampling()* a vyžaduje dokument jako vstupní parametr. Interpolaci je nezbytné provést u všech proměnných vždy v relaci s časovou osou. Jízda je nejprve očištěna o hodnoty "null". Očištěná data se uloží do oddělených datových polí. Jelikož jde o kubickou interpolaci, tj. proložení křivkou, postupuje se iterativně po čtyřech vzorcích s překryvem jednoho vzorku. Pro zachování kontinuity je tedy poslední vzorek první čtveřice současně prvním vzorkem čtveřice následující.

Původní interval se v rychlostech nad 50 km/h pohybuje okolo 2,5 vteřiny (frekvence 0,4 Hz). K převzorkování byl tedy vybrán interval dvou vteřin. Z každé čtveřice časů je vybrána první a poslední hodnota. Pomocí operátoru modulo (zbytek po dělení) jsou uvnitř for smyčky uloženy do samostatného pole všechny hodnoty timetamp dělitelné dvěma. Toto platí pro všechny hodnoty uvnitř zvoleného intervalu včetně první hodnoty a kromě poslední. Nežádoucí vkládání stejných časů kvůli překryvu je tak ošetřeno. Pole cílových časů bude opakovaně použito s každou proměnnou zvlášť.

Dále je volána funkce *interpolation* se třemi parametry; pole časů, pole hodnot příslušné proměnné a pole cílových časů. Interpolace se dělí na dvě funkce z knihovny *scipy.interpolate.interpolate*. Nejdříve je pomocí funkce *splrep* a původních polí nalezena polynomiální funkce třetího řádu. Funkce je principiálně založena na metodě nejmenších čtverců. Funkce tedy nemusí přesně procházet všemi body a zahlazuje extrémy. Poté jsou

pomocí funkce *splev*, vyžadující jako parametry zmíněný polynom a pole cílových časů, dopočteny nové hodnoty jednoduchým dosazováním cílových časů. Výsledkem mateřské funkce *interpolation* je pole cílových časů (s periodou dvou vteřin) a pole nových, odvozených hodnot cílové proměnné.



Graf 2: Průběh rychlosti v čase u krátké jízdy před a po interpolaci.

Nová pole jednotlivých proměnných jsou vložena do nově vygenerovaného slovníku (dict). V databázi je celý obsah úrovně mapData nahrazen novým slovníkem stejného formátu a pouze jiné délky, resp. počtu vzorků. Přestože dochází k případům podvzorkování i nadvzorkování, obecně se počet vzorků zvyšuje.

5 Analýza dat

V zásadách pro zpracování práce je návrh takového statistického modelu, jež je schopen detekovat různé jízdní režimy. Podle zadání lze klasifikovat celé jízdy nebo jejich části. Metod pro takovou klasifikaci je celá řada, protože neexistuje trénovací datový set, a jeho vytvoření by bylo časově náročné, nezbývá než využít některé z metod klasifikace bez učitele. Princip bude spočívat ve dvojici klasifikátorů, z nichž první bude mít za cíl klasifikovat jednotlivé vzorky (nezávisle na tom, do které jízdy patří) a označit je podle příslušných tříd. Druhý klasifikátor pracuje s oddělenými jízdami a označenými vzorky s cílem nalézt hlavní struktury a rozdíly v jízdách.

Klasifikátorům předchází přípravná procedura PCA neboli dekompozice do hlavních komponent. Nejmocnějšími a nejvíce užívanými jsou shlukovací algoritmy. Předmětem clusterování jsou všechny proměnné s výjimkou efektivity a rychlosti. Rychlost poslouží k ověření správnosti klasifikátoru. Překračování nejvyšší povolené rychlosti je hlavním parametrem. K jejímu zjištění použiji dotazovací nástroj OpenStreetMap.

5.1 Výběr testovacího vzorku

Při analýze je potřeba brát ohledy na jednotlivé motorizace. Zvlášť mezi naftovými a benzínovými motory jsou veliké provozní rozdíly. Naftové motory obecně běží na nižších otáčkách a také spotřeby se mohou významně lišit. Totéž platí pro výkonnostní kategorie. Pro lepší přehled v motorizacích a typech vozů jsem navrhl následující dotaz:

```
collectionSpec.aggregate([
    {"$group":
            {"_id": {"type": "$content.data.vehicleType",
                     "fuel": "$content.data.engineTypePrimary",
                     "power": "$content.data.maxPower"},
                     "count": {"$sum": 1}
                    }
            },
    {"$sort": bson.SON([("count", -1), ("id", -1)])}
])
```

Klíčová slova “aggregate” a “group” mají v tomto dotazu stejný účel jako má klíčové slovo “group by” v SQL. Dotaz tedy nalezne počet dokumentů (tedy vozů) s identickým typem (např. Octavia), stejným palivem a výkonem. Výsledek uloží do jednoduché struktury binárního JSONu. Tato operace je provedena pro každou nalezenou unikátní motorizaci. Konečným výsledkem je tedy kursor před všemi tyto dokumenty uloženými v paměti. Lze je vytisknout do konzole jednoduchou *for* smyčkou. Výsledek vypadá takto:

```
{'_id': {'type': 'KODIAQ', 'fuel': 'PETROLGASOLINE', 'power': 110}, 'count': 53}
{'_id': {'type': 'KODIAQ', 'fuel': 'PETROLDIESEL', 'power': 110}, 'count': 47}
{'_id': {'type': 'OCTAVIACOMBI', 'fuel': 'PETROLDIESEL', 'power': 110}, 'count': 42}
{'_id': {'type': 'OCTAVIACOMBI', 'fuel': 'PETROLGASOLINE', 'power': 110}, 'count': 42}
{'_id': {'type': 'OCTAVIACOMBI', 'fuel': 'PETROLDIESEL', 'power': 85}, 'count': 26}
{'_id': {'type': 'SUPERBCOMBI', 'fuel': 'PETROLDIESEL', 'power': 110}, 'count': 25}
...
```

Těchto unikátních kombinací motorizací je v kolekci 93. Pro zjednodušení lze pro analýzy sjednotit vozidla typu sedan a kombi. Z výsledku dotazu je zřejmé, že hlavním předmětem výzkumu budou vozy Kodiaq, Octavia a Superb. Analýza i testování správnosti algoritmu jsou potřeba provádět na co největším vzorku jízd. Z tohoto důvodu jsem se rozhodl zanedbat vozy, které se nevyskytují v dostatečném počtu.

Pro nalezení všech jízd provedených skupinou automobilů bylo potřeba napsat dotaz pracující s více kolekcemi. Já jsem se rozhodl provést tuto operaci v aplikační vrstvě Python. Skript projde všechny specifikace a vybere VIN kódy těch, které odpovídají zkoumané skupině. Skript zahrne sedany i kombi. Následně najde v kolekci jízd všechny dokumenty s příslušnými VIN kódy.

Jako první je vždy analyzována skupina typ: Octavia, palivo: benzin, maximální výkon: 110 kW. Skupina však obsahuje 4259 jízd a 2 643 978 vzorků což přesahuje možnosti většiny

klasifikačních algoritmů, pokud bereme v úvahu hardware dostupný na osobních počítačích. Ze seznamu jízd je třeba několik jízd vybrat. K pseudonáhodnému výběru bez opakování používám funkci *random.sample*. Abych pokryl dostatečně pokryl různorodost jízd, vybral jsem 100 dokumentů. Vzorky ze všech dokumentů se uloží do jedné datové struktury. Datová struktura první skupiny má 7 proměnných a 67 429 měření.

5.2 Analýza hlavních komponent

Analýza hlavních komponent (Principal component analysis – PCA) je přípravná statistická procedura nebo transformace sloužící k dekorelaci dat a redukci vstupních dimenzí při zachování většiny informací. Pokud se v datech objevují významně korelující proměnné, touto procedurou je možné počet proměnných zredukovat. PCA je průmět dat do jiné souřadné soustavy lineárně dekorelovaných dat zvaných hlavní komponenty.

Mějme kupříkladu dvě nekorelující proměnné A a B, jejichž do roviny promítnuté vzorky tvoří výplň elipsy. Potom osy pomyslné elipsy (rovnoběžné s osami proměnných) reprezentují rozptyl dat proměnných A, B. Čím větší bude rozptyl proměnné A, tím větší bude její obsah informace oproti proměnné B. Při poměru obsahu informace např. 10/1 ve prospěch A, můžeme proměnnou B z analýzy odstranit při ztrátě 10% variability respektive informace.

Korelující proměnné je třeba transformovat tak, aby osy pomyslné elipsy (nerovnoběžné s osami proměnných) tvořily osy nové souřadné soustavy. Tyto osy se nazývají hlavní komponenty transformovaných, dekorelovaných dat. Kromě dekorelace a redukce proměnných, je PCA užitečná pro základní průzkum vlastností dat.

5.2.1 Normalizace, sestavení kovarianční matice

PCA je náchylná na různé škály proměnných. Například může být proměnná rpm (otáčky za minutu) milně označena jako proměnná s největší variabilitou, protože dosahuje mnohem vyšších hodnot než na příklad proměnná FrontAcc (akcelerace). Ta nabývá pouze hodnot přibližně od –2 do 2 a podle PCA by oproti otáčkám nesla jen nepatrný objem variability. Proto je třeba nejprve data normalizovat. Nejprve jsou data převedena do formátu *numpy.array,* který je nejčastěji užíván analytickými nástroji a umožňuje snazší a rychlejší manipulaci. K účelu normalizace je volána funkce *StandardScaler()* z knihovny *sklearn.preprocessing*. Cíl je průměr 0 a směrodatná odchylka 1 u všech naškálovaných proměnných. Hodnota každého vzorku x je přepočtena podle tohoto vzorce:

$$z = \frac{x - u}{s}$$

Kde *u* je průměr všech vzorků proměnné, *s* je směrodatná odchylka a *z* je normalizovaný vzorek. Standardní škálování není důležité jen pro PCA ale je také základem mnoha algoritmů strojového učení včetně shlukovacích. Zvláště pak je důležitá u shlukovacích algoritmů založených na eukleidovské vzdálenosti mezi body (např. K-means). Nyní lze začít s procedurou PCA.

Prvním krokem dekompozice je sestrojení kovarianční matice. Kovarinace je statistická míra linearní závislosti dvou proměnných a je definována takto:

$$cov(X,Y) = E[(X - E[X])(Y - E[Y])]$$

Kde cov(X, Y) je kovariance proměnných X a Y, E[X] je střední hodnota měření. Podobně jako korelace, i kovariance nabývá hodnot v intervalu od –1 do 1, kde –1 značí nepřímou úměrnost, 1 značí přímou úměru a 0 linearní nezávislost. Kovarianční matice je matice rozměrů k×k, kde k je počet proměnných. Pro každý prvek s indexy i a j v matici platí, že je roven hodnotě kovariance i-té a j-té proměnné. U kovariance nezáleží na pořadí, a proto je matice diagonálně symetrická. U normalizovaných proměnných jsou všechny hodnoty na diagonále rovny 1 a kovarianční matice je ekvivalentní korelační matici. V jazyce Python lze kovarianční matici určit mimo jiné pomocí funkce *cov(X)* z knihovny *numpy*.

	rpm	consumption	sideG	outputPower	frontAcc
rpm	1.	-0.11	-0.046	0.6	0.23
consumption	-0.11	1.	-0.072	0.231	0.407
sideG	-0.046	-0.072	1.	0.008	-0.018
outputPower	0.6	0.231	0.008	1.	0.575
frontAcc	0.23	0.407	-0.018	0.575	1.

Tabulka 8: Kovarianční matice proměnných vybrané pro analýzu

Významná lineární závislost je mezi otáčkami a výkonem, akcelerací a výkonem, akcelerací a spotřebou.

5.2.2 Vlastní vektory a hodnoty

Druhým krokem PCA je nalezení vlastních vektorů a vlastních hodnot. Vlastní vektor (tj. Eigenvector) je takový nenulový vektor, který se po aplikaci lineární transformace změní pouze o skalár (tedy délku) a nemění svůj směr s výjimkou obráceného směru při násobením zápornou hodnotou. Koeficient, kterým se vlastní vektor při transformaci násobí se nazývá vlastní hodnota (eigenvalue). Reálné vlastní vektory mají jen některé čtvercové matice. Vlastní vektory jsou u symetrických matic na sebe kolmé.

V tomto případě je potřeba získat vlastní vektory kovarianční matice Σ. Matici vlastních vektorů (ve sloupcích) nazveme E_x. Vlastní vektory a vlastní hodnoty lze získat pomocí funkce *linalg.eig()* z knihovny *numpy*. Vlastní vektory definují směry, podél kterých mají data největší rozptyl.

Vlastní vektory				
0.44671	-0.52834	0.38369	0.59496	0.14184
0.3088	-0.12403	0.60712	-0.71899	-0.06096
-0.0449	-0.06187	0.09271	0.15284	-0.98092
0.6257	0.75592	0.0469	0.18163	-0.04359
0.55819	-0.36088	-0.68804	-0.26967	-0.10984
Vlastní hodnoty				
2.05516	0.26752	0.47623	1.20063	1.00053

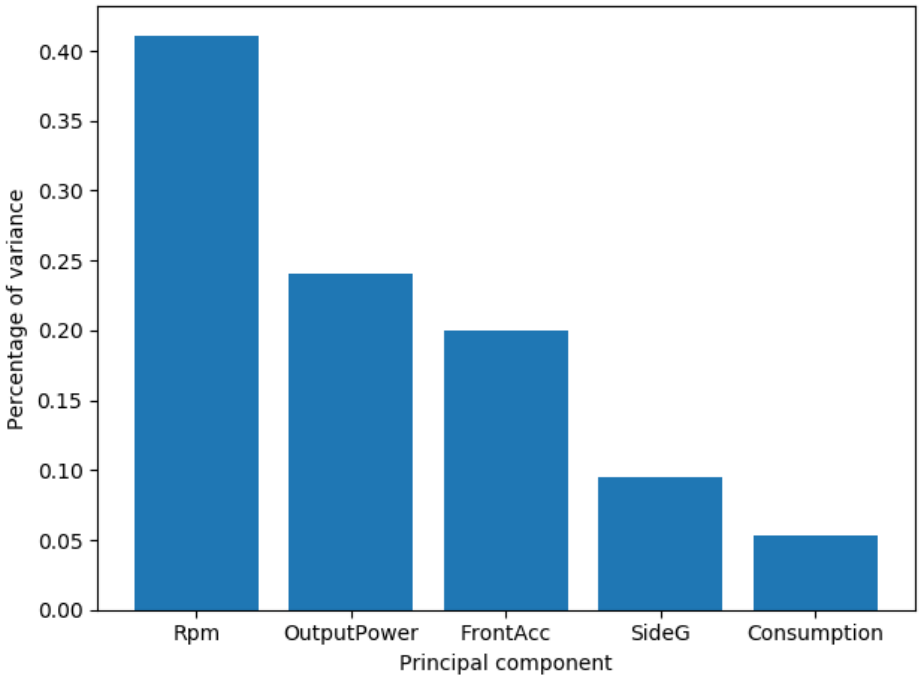
Tabulka 9: Vlastní vektory a hodnoty kovarianční matice

Matici dekorelovaných dat D získáme podle následujícího vztahu:

$$D = E_x^T P^T$$

Kde E_X^T je transponovaná matice vlastních vektorů a P^T je transponovaná matice normalizovaných dat.

Pro vyhodnocení a výběr hlavních komponent pro analýzu se používá procento rozptylu, jež každá z komponent obsahuje. Komponenty jsou seřazeny podle procenta obsažené variability. Přestože se jedná o hlavní komponenty analýzy, záměrně jsem použil názvy původních proměnných pro lepší přehled. Nelze však tvrdit, že proměnné obsahují daná procenta rozptylu, neboť jsou to hlavní komponenty z nich odvozené.



Graf 3: Histogram hlavních komponent a jejich podílu na celkové variabilitě

Otáčky a výkon jsou dvě nejdůležitější hlavní komponenty a společně tvoří přes 60 % veškeré variability. Při odstraňování komponent je ideální zachovat alespoň 90-95 % původního rozptylu. Proměnná frontAcc, která reprezentuje akceleraci, obsahuje 20 % procent rozptylu. Boční přetížení obsahuje 9,5 %. Z kovarianční matice je patrné, že spotřeba významně souvisí se akcelerací a výkonem. Není tedy překvapením, že proměnná pozbyla v dekorelovaných datech svého významu a její odstranění nezpůsobí velkou ztrátu informace. Přesto je třeba brát v úvahu, které proměnné mají význam při řešení konkrétního zadání. Spotřeba může vypovídat o chování řidičů a je důležité jí ponechat. Pro shlukování jsem vybral všechny proměnné.

5.3 Klasifikace vzorků pomocí K-means

Jedním z nejpoužívanějších shlukovacích algoritmů je k-means (nebo také Lloydův algoritmus). K-means vytváří třídy se stejným rozptylem. Algoritmus vyžaduje parametr počet tříd *k*. Předpokládá kulovité rozložení vzorků ve třídě a nepodává dobré výsledky u podlouhlých clusterů. Nejprve je náhodně vybráno *k* bodů nazývané centroidy. Obvykle to nebývají přímo vzorky, ale body, které s nimi sdílí prostor. Každý vzorek se přiřadí k tomu centroidu, jehož eukleidovská vzdálenost je nejmenší. Poloha centroidu se poté přepočítá tak, aby tvořil těžiště těchto bodů. Tyto operace se provádějí iterativně, dokud se centroidy neustálí. Algoritmus cílí na minimalizaci součtu druhých mocnin eukleidovských vzdáleností bodů uvnitř clusteru.

Časová složitost algoritmu je pouze O(nkd) kde:

n je počet vzorků pro klasifikaci

k je počet clusterů

d je počet dimenzí (proměnných)

Při klasifikaci bez učitele nezbývá než postupovat heuristicky. Jelikož neexistuje žádný způsob ověření správnosti, jsou výsledky hodnoceny podle metrik, které jsou implementované v knihovně sci-kit learn a vlastního úsudku. Díky složitosti k-means lze použít všechny hlavní komponenty bez výrazně delších časů na provedení. Rovněž lze klasifikovat velkou skupinu jízd a porovnat, jak se centroidy mění.

K-means má kromě počtu clusterů několik dalších parametrů. První *init* formátu np.array jsou souřadnice počátečních centroidů (velikosti počet clusterů × počet proměnných). Pokud nejsou poskytnuty souřadnice, vybere je algoritmu tak, aby dosáhl ustálení (konvergence) v co nejmenším počtu iterací. Druhý *n_init* značí, kolikrát bude algoritmus proveden s různými souřadnicemi počátečních centroidů. Výchozí hodnota je 10 a algoritmus vždy vybere tu skupinu centroidů, která v rámci metrik podá nejlepší výkon. Třetí *max_iter* je počet iterací (výchozí 300). Centroidy se ve většině případů ustálí dřív, než je tato hodnota překročena. Desetinné číslo *tol* udává relativní toleranci pro hlavní metriku k vyhodnocení ustálenosti centroidů. Výchozí hodnota je $1*10^{-4}$. Pátý parametr *precompute_distances* předpočítá vzdálenosti bodů, pokud je jeho hodnota "True". Výchozí hodnota je "auto", která předpočítá vzdálenosti pouze pokud n × d < 12 milionů. Šestý parametr *random_state* ovlivňuje determinismus při náhodném výběru inicializačních centroidů. Výchozí hodnota je None. Sedmý parametr *n_jobs* je typu integer a specifikuje počet procesů v případě paralelního zpracování. Výchozí hodnota je None a odpovídá jednomu procesu. Posledním prametrem je

algorithm, který udává typ algoritmu. Výchozí hodnotou je *auto*, která zvolí algoritmus *full* pro řídká data a *elkan* pro hustá data. *Elkan* využívá trojúhelníkové nerovnosti ke snížení počtu nezbytných výpočtů a tím přispívá ke zkrácení času na provedení. Knihovna ale zatím nepodporuje jeho využití na řídkých datech.

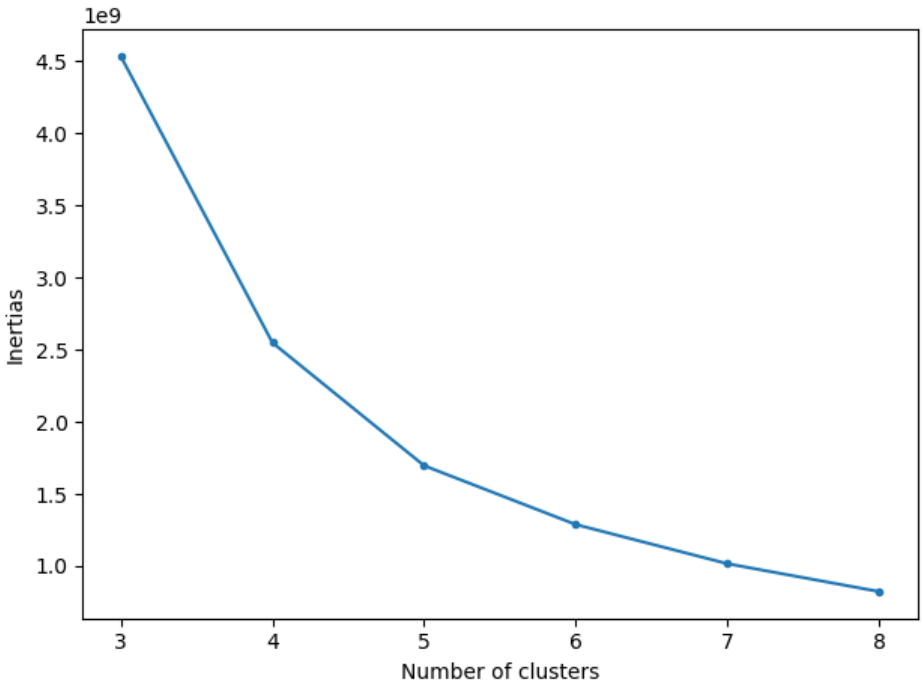
5.3.1 Výběr vhodného počtu clusterů

První série pokusů bude provedena s hlavní testovací skupinou a za použití všech proměnných. Skript bude postupně dosazovat za *k* čísla od 3 do 9. Jednotlivé součty druhých mocnin vzdáleností bodů od clusterů se budou ukládat do pole. Dvojice můžeme poté vizualizovat. Python kód vypadá následovně.

```
inertias = []
clust_num = [3,4,5,6,7,8,9]
for one in clust_num:
     kmeans = KMeans(n_clusters=one, init="k-means++",
     n_init=10, max_iter=300, precompute_distances="auto",
     random_state=None, algorithm="auto").fit(X)
         inertias.append(kmeans.inertia_)
```

Vizualizace se vždy provádí pomocí knihovny matplotlib.

Počty clusterů a součty druhých mocnin vzdáleností vzorků od nejbližšího centroidu (inertia) tvoří pomyslnou exponenciální funkci. Nejlepší volbou je bod v místě prohybu exponenciály, neboť s každým dalším přidaným clusterem už nedojde výraznému zlepšení. V různých anglicky psaných pracích se nazývá „elbow“ point. Provedl jsem pokus na datech před normalizací a PCA.



Graf 4: Počty klastrů a sumy druhých mocnin vzdáleností vzorků od centroidů

Kvůli vlastnostem dat, se exponenciála se neprohýbá příliš významně. Pětice clusterů bude v tomto případě ideální. Existují i další metriky pro vyhodnocování kvality separace clusterů. Například *sklearn.metrics.silhouette_score* je koeficient vypočtený z průměru vzdáleností uvnitř clusteru od kterého je odečten průměr vzdáleností vzorků k nejbližšímu cizímu clusteru. Jelikož počítá vzdálenosti všech možných dvojic, je metoda paměťově velice náročná a může být provedena pouze s několika tisíci vzorky (jedna desetina vzorků ve skupině). Při opakovaných pokusech se výsledky velmi mění a metodu nelze spolehlivě využít.

5.3.2 Popis clusterů

Porovnání výsledků pro k=5 spočívá především v porovnávání pozic centroidů clusterů, směrodatných odchylek a počty obsažených vzorků. K definování reprezentace clusterů je potřeba označit vzorky v původní formě před dekorelací a škálováním. K tomuto účelu poslouží výstupní atributy *labels_* a *cluster_centers_* objektu kmeans, který je definován v příkladu. Oba atributy jsou předány jako výstupní parametry funkce, která zastřešuje všechny operace spojené s algoritmem k-means. Parametr *labels_* je *numpy.array* délky *n* (počet vzorků) obsahující značky jednotlivých clusterů. Clustery jsou číslované od nuly. Pomocí funkce *numpy.vstack()* lze snadno rozšířit původní datovou matici o sloupec se značkami clusterů. Parametr *cluster_centers_* je stejného typu ale rozměrů $k \times d$ (počet clusterů a počet dimenzí).

Obsahuje souřadnice všech centroidů avšak v jiné souřadné soustavě. Místo složitého převádění, jsem znovu transponoval datovou matici, a rozdělil jí do *k* menších matic. Dále jsem použil funkci *numpy.mean()* pro každou z nich, která spočte aritmetický průměr pro jednotlivé sloupce. Výsledkem je matice $k \times d$ aritmetických průměrů. Řádky matice jsou souřadnice centroidů v původní souřadné soustavě, neboť jedna souřadnice centroidu odpovídá aritmetickému průměru vzorků v clusteru (z názvu k-průměrů).

Aritmetický průměr								
	speed	rpm	consumption	efficiency	sideG	outputPower	frontAcc	Počet vzorků
A	54.436	1773.966	5.176	65.457	-0.02157	4.953	-0.107	8850
B	65.584	1773.727	7.275	75.260	-0.00131	12.610	0.194	23758
C	5.334	1010.628	40.675	74.514	-0.00671	3.365	0.172	2890
D	99.534	2502.652	12.814	54.247	-0.00578	38.601	0.583	5809
E	42.860	1312.344	3.020	82.451	-0.00123	1.409	-0.376	26122

Tabulka 10: Aritmetické průměry proměnných v jednotlivých clusterech, set 100 jízd

Některé centroidy se v určitých dimenzích významně liší od ostatních. Označil jsem konkrétní hodnoty podtržením. Samotné průměry ale mohou být zavádějící a je vhodnější je doplnit o směrodatné odchylky pomocí funkce *numpy.std()*.

Směrodatná odchylka								
	speed	rpm	consumption	efficiency	sideG	outputPower	frontAcc	Počet vzorků
A	31.567	542.858	5.099	28.406	0.0112	5.493	0.363	8850
B	28.810	362.879	4.110	23.824	0.0076	6.191	0.263	23758
C	6.034	292.425	13.886	22.315	0.01376	5.299	0.543	2890
D	40.077	577.435	6.634	28.367	0.00761	15.178	0.471	5809
E	28.125	424.697	3.701	18.513	0.00805	2.067	0.436	26122

Tabulka 11: Směrodatné odchylky proměnných v jednotlivých clusterech, set 100 jízd

Cluster A se vyznačuje pouze vyšším bočním zrychlením. Všechny ostatní hodnoty jsou nevýrazné.

Cluster B je druhým největším co do obsahu vzorků. V průměru mírně zrychluje což potvrzuje i vyšší výkon a spotřeba. Akcelerace má nízkou směrodatnou odchylku, a tudíž nedosahuje extrémů. Cluster reprezentuje většinu vzorků naměřených během plynulé jízdy nebo mírného rozjezdu.

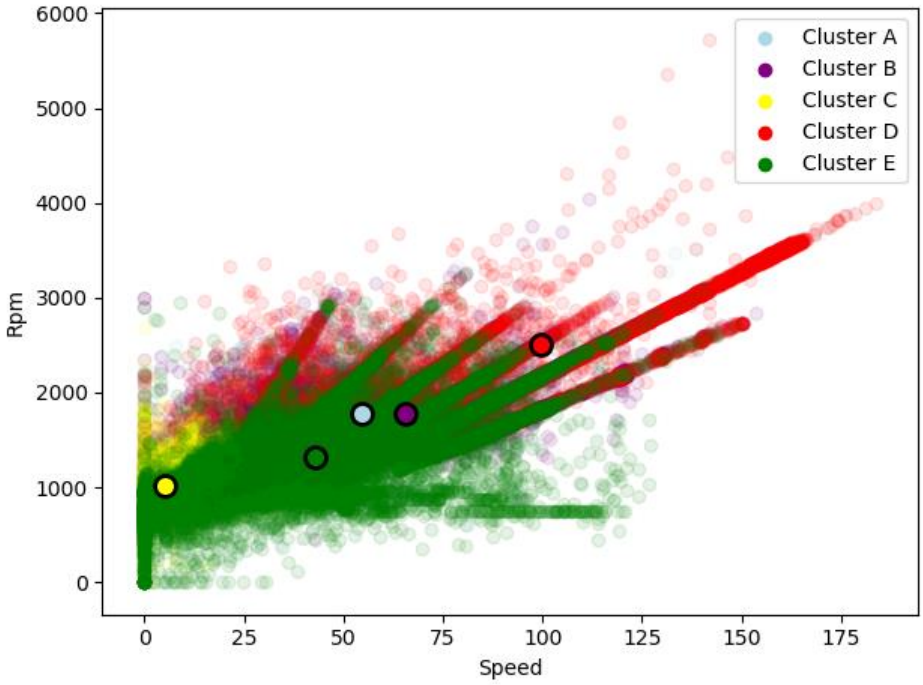
Cluster C je nejodlehlejší a nabývá extrémních hodnot v několika atributech. Rychlost je velice nízká a s nízkou odchylkou podobně jako otáčky za minutu. Naopak spotřeba je extrémně vysoká a s velkou odchylkou. Cluster obsahuje pouze necelé tři tisíce vzorků.

Pravděpodobně jde o produkt kubické interpolace v úsecích s dlouhým intervalem vzorkování. V těchto případech může jediná extrémní hodnota způsobit vysokou amplitudu a doplnění několika dalších extrémů v rámci nadvzorkování.

Cluster D je nejhodnotnější třídou v analýze. Obsahuje vzorky s vysokými otáčkami, spotřebou, akcelerací i výkonem. Odchylky u rychlosti a výkonu jsou znatelně vyšší než u ostatních clusterů. Cluster popisuje prudké zrychlování v různých rychlostech. Vzorky tohoto clusteru budou mít důležitou vypovídající hodnotu pro klasifikaci jízd.

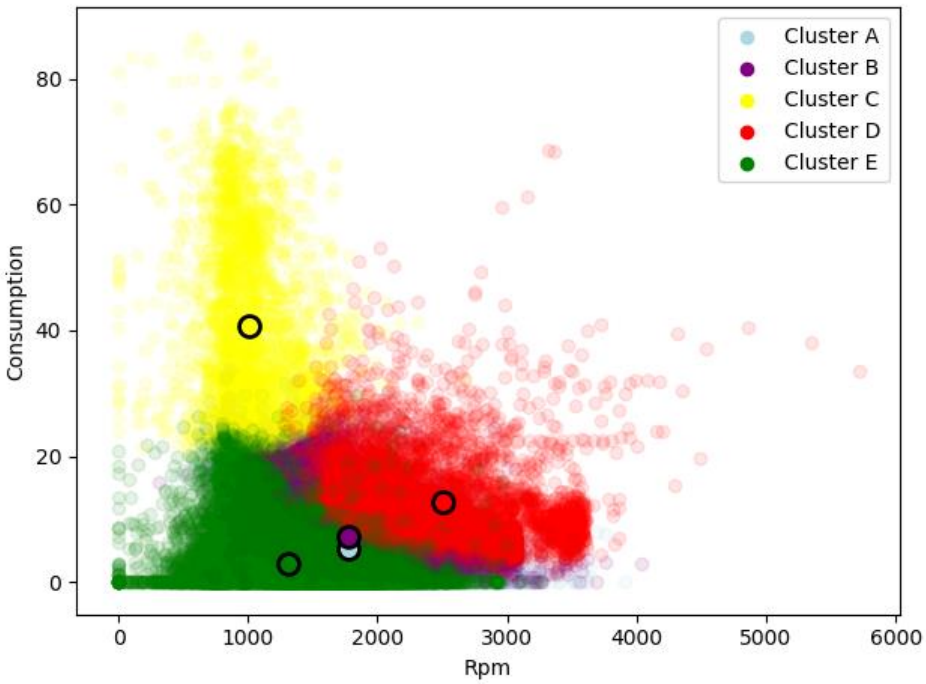
Cluster E je největším clusterem a obsahuje téměř všechny vzorky popisující zpomalování nebo brždění. Napovídá tomu nízký výkon a záporná hodnota akcelerace. Rovněž spotřeba je velice nízká.

Pro lepší představu jsem provedl několik vizualizací.



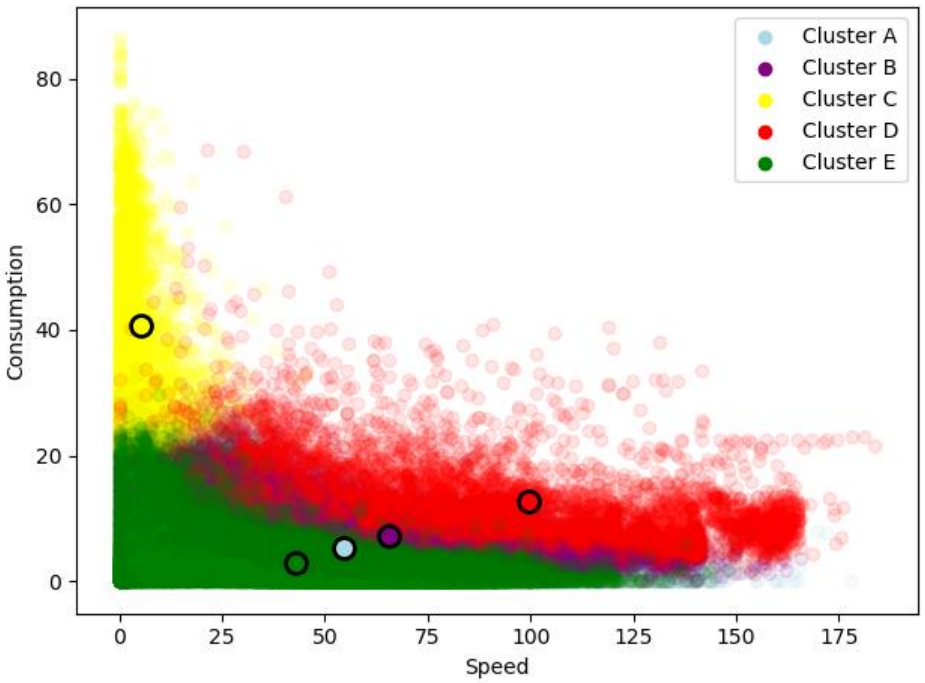
Graf 5: Množina vzorků vizualizována na osách „rpm“ a „speed“, obarvená podle clusterů, set 100 jízd

První graf demonstruje rozložení clusterů v dimenzích otáčky a rychlost. Černě obtažené body jsou centroidy. Vzorky se překrývají v pořadí, v jakém byly do grafu zakresleny. Například vzorky clusteru B téměř nelze pozorovat.



Graf 6: Množina vzorků vizualizována na osách „consumption" a „rpm", obarvená podle clusterů, set 100 jízd

Na druhém grafu je zvlášť patrná vzdálenost C od všech ostatních na ose spotřeby.



Graf 7: Množina vzorků vizualizována na osách „consumption" a „speed", obarvená podle clusterů, set 100 jízd

Na grafu 5 lze pozorovat, že se cluster E drží v nízké spotřebě paliva ve vyšších rychlostech, tak jak ukazují tabulky. Rozptyl v rychlosti je ohraničen rychlostí 120 km/h.

Proměnná sideG, tedy boční přetížení, nehraje v klasifikaci příliš velkou roli. Může to být způsobeno faktem, že i přes standardizaci má proměnná v porovnání s ostatními velice malý rozptyl. Rovněž její použití nemuselo být zvoleno zrovna šťastně, neboť rozptyl od kladných do záporných hodnot reprezentuje vektor bočního zrychlení. Bočního zrychlení má tedy vliv na eukleidovskou vzdálenost vzorků od centroidů. Vhodnější by bylo použít absolutní hodnotu a sledovat, zda se vzorky s vyššími hodnotami seskupí dohromady.

Tuto metodu považuji za částečně úspěšnou. S její pomocí došlo k nalezení a definici základní struktury v datech. Tyto třídy lze použít jako základ k dalším pokusům.

5.4 Testování procedury na větší skupině jízd

V předchozích pokusech došlo ke klasifikaci vzorků ze stovky jízd. Cílem práce je ale klasifikace všech (nebo většiny) jízd. Nejjednodušším způsobem je použití centroidů z posledního pokusu a přiřadit všechny zbývající vzorky k té třídě, jejíž centroid je vzorku nejblíž. Současně by ale mělo dojít k přepočtení souřadnic centroidů, neboť se jejich poloha bude se vzrůstajícím počtem vzorků mírně měnit. Tato procedura je ale neudržitelná, neboť se přepočtení pozice s přibývajícími vzorky stane časově i paměťově náročnou operací. Rozhodl jsem se podrobit pokusu novou skupinu jízd, která bude o řád větší než v předchozím pokusu. Výsledky napoví, zda byla původní množina dostatečně velká.

Aritmetický průměr								
	speed	rpm	consumption	efficiency	sideG	outputPower	frontAcc	Počet vzorků
A	50.393	1742.661	6.289	67.403	-0.02557	6.299	-0.048	74638
B	65.147	1813.709	6.314	72.268	-0.00224	10.160	0.102	231453
C	5.418	996.325	40.508	75.632	-0.00915	3.189	0.176	28941
D	89.560	2315.661	12.494	59.730	-0.00611	33.634	0.572	74669
E	38.410	1256.071	3.051	81.657	-0.0043	1.095	-0.430	206860
Směrodatná odchylka								
	speed	rpm	consumption	efficiency	sideG	outputPower	frontAcc	Počet vzorků
A	29.039	500.909	5.396	28.304	0.01308	6.263	0.372	74638
B	28.594	391.855	3.993	26.330	0.00797	6.010	0.274	231453
C	5.804	283.289	14.947	22.271	0.01265	4.902	0.567	28941
D	37.41142	515.87509	6.453	27.019	0.00859	14.260	0.456	74669
E	28.398	414.02561	4.012	18.485	0.0075	1.811	0.464	206860

Tabulka 12: Aritmetické průměry a směrodatné odchylky proměnných v jednotlivých clusterech, set 1000 jízd

Nové centoridy se až na výjimky od původních příliš neliší. Souřadnice centroidů (průměry) jsou téměř identické a mírné rozdíly v průměrech i rozptylech můžeme tolerovat. Dokonce

i počty vzorků v jednotlivých clusterech takřka jen vzrostly o řád a zachovaly mezi clustery stejný poměr. Původní množina jedné stovky jízd je tedy dostatečně velká na clusterování.

Funkce *KMeans* obsahuje kromě metody *fit()* i další metody, mezi které patří i metoda *predict()*. Ta na základě původního modelu predikuje zařazení nové množiny vzorků do původních clusterů. Každý vzorek je přiřazen k té třídě, jejíž centroid má nejmenší eukleidovskou vzdálenost od vzorku. Přestože se původní centroidy od nových příliš neliší, použiji pro vytvoření modelu množinu jednoho tisíce jízd. Centroidy tak budou stabilnější a spolehlivější, neboť jde o jednu čtvrtinu jízd ve skupině.

Nejprve vyberu pseudonáhodně jednu tisícovku jízd a vytvořím k-means model. Identifikátory uložím do pole, abych při predikci zbylých jízd nepoužil jízdy dvakrát. Vzorky ze všech ostatních jízd uložím do datové struktury *numpy.array* a provedu analýzu hlavních komponent. Nepřekvapivě jsou její výsledky srovnatelné s předchozími pokusy. Je však nezbytné opět všechny vzorky standardizovat a použít v dekorelované souřadné soustavě. Pro predikci jsem vytvořil novou metodu *k-means_prediction()*. Oproti původní vyžaduje kromě matice hlavních komponent i původní model. Z původního modelu je volána zmíněná metoda *fit()* s maticí jako parametrem. Výstupem metody je pouze jednorozměrné pole se značkami clusterů pro každý vzorek.

V této fázi už nepřidám značky do původní datové matice ale přímo do databáze. Použiji zmíněné pole identifikátorů pro dotazy MongoDB. Algoritmus vrací značky ve stejném pořadí, jako byly vzorky vloženy. Není tedy třeba pole značek dělit ale pouze postupovat iterativně po jednotlivých jízdách. Každá jízda je upravena pomocí příkazu *update_one()*. S uloženými značkami vzorků mohu začít s návaznými experimenty, které se budou soustředit na třídění jízd samotných.

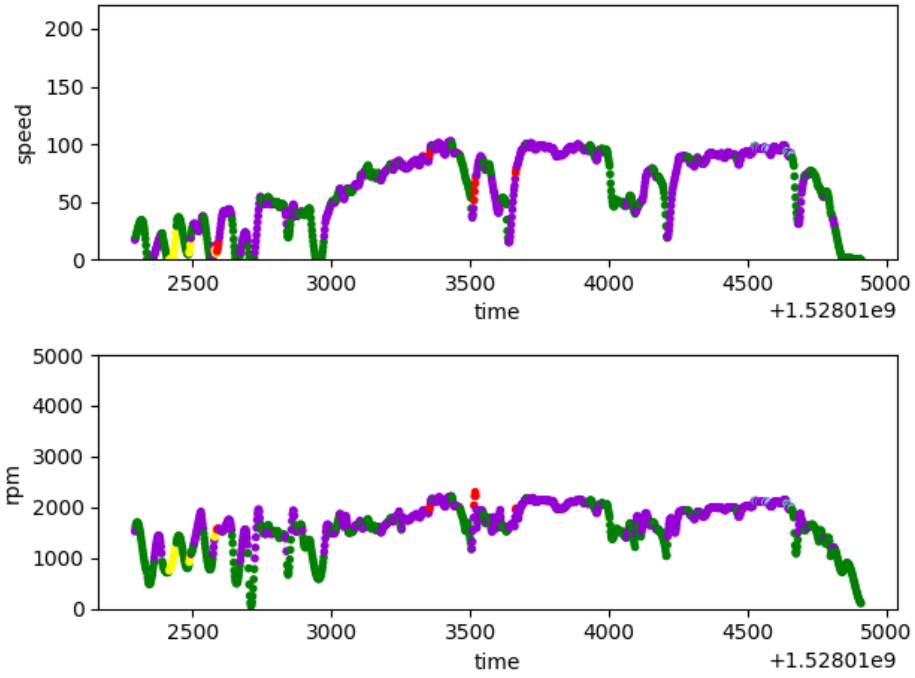
5.5 Řazení jízd podle vytvořených tříd vzorků

Nejjednodušším způsobem, jak klasifikovat jízdy na základě obsahu vzorků je použít jejich procentuální zastoupení. Každá jízda je tvořena vzorky a každý z nich náleží do jedné z pěti tříd. Počet vzorků jedné třídy dělený počtem vzorků v jízdě udává procentuální zastoupení třídy v jízdě. Tímto postupem lze z jízd vytvořit matici rozměrů počet jízd × počet tříd, a seřadit jízdy na základě zastoupení některé z tříd. Hlavní nevýhodou tohoto postupu je ztráta informace o přesné posloupnosti vzorků. Průměry a rozptyly procentuálních zastoupení opět poskytnou základní přehled.

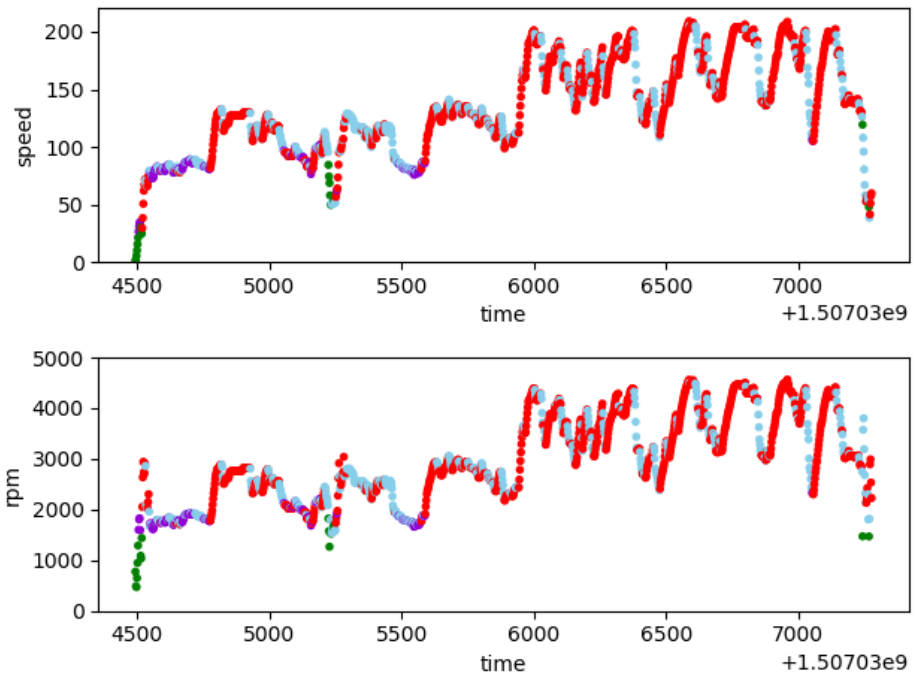
Cluster	A	B	C	D	E
Průměrné procentuální zastoupení	0.191857	0.3186936	0.0499171	0.0949739	0.344557
Směrodatná odchylka procentuálního zastoupení	0.194513	0.1448269	0.0507001	0.1007545	0.146950

Cluster A je tvořen vzorky s vyšším bočním zrychlením. Jak ale bylo zmíněno, použití hodnot z bočních akcelerometrů nebylo správné a hodnoty nejsou věrohodné. Cluster B je vhodnější, neboť je dostatečně velký a reprezentuje plynulou jízdu nebo mírné zrychlování. Odchylky v rámci B jsou přijatelné a podíl vzorků na jízdách je třetinový. Cluster C pravděpodobně nebude mít na klasifikaci žádný vliv, jelikož je tvořen převážně chybami. V průměru má pouze tříprocentní zastoupení v každé jízdě a můžeme ho tedy zanedbat. Hlavní rozlišení ponesou vzorky clusteru D, které dobře ilustrují neekonomické chování nebo agresivní styl jízdy. Cluster E obsahuje většinu vzorků zaznamenávající zpomalování a jeho velikost i zastoupení jsou významné. Nerozlišuje však mezi mírným a prudkým brzděním což je pro zadání zásadní. Využiji tedy clusterů B a D k třídění jízd podle zastoupení vzorků z těchto clusterů.

Jak ale bylo zjištěno během přípravy, nelze spolehlivě analyzovat jízdy v nízké rychlosti kvůli nízké vzorkovací frekvenci. Přestože se nelze těmto vzorkům úplně vyhnout, postačí se soustředit na delší jízdy ve vyšších rychlostech. Vyberu tedy pouze jízdy s průměrnou rychlostí vyšší než 60 km/h. Takových jízd je v rámci zkoumané motorizace 1034. Seřadím jízdy nejprve podle zastoupení třídy B a vyberu několik jízd s nejvyšším zastoupením. Stejnou množinu jízd seřadím podle zastoupení třídy D a vyberu několik jízd. Jízdy z obou řazení porovnám.



Graf 8: Rychlost a otáčky v čase u vybrané jízdy s vysokým podílem vzorků z clusteru B



Graf 9: Rychlost a otáčky v čase u vybrané jízdy s vysokým podílem vzorků z clusteru D

Graf 6 ilustruje převážně plynulý styl jízdy. Otáčky za minulu se pohybují v rozmezí 1500 a 2000 za minutu a oscilují jen minimálně. Rychlost se až na výjimky mění plynule nebo zůstává konstantní. Na počátku grafu 7 je patrný podobný úsek konstantní jízdy. V druhé polovině však dochází k významnému kmitání rychlosti i otáček. Otáčky nezřídka překračují 4000

za minutu, rychlost dosahuje i 200 km/h. Přestože typ komunikace, po které se vozidla pohybují, je pravděpodobně jiný, druhou jízdu lze označit při nejmenším za neekonomickou.

Závěr

Příprava dat v tomto měřítku a kvalitě je velice náročná a vyžaduje mnoho dodatečných operací. Jejich využití v komerčním měřítku je značně omezené. V mnoha úsecích měření nelze data ošetřit do použitelné podoby, protože interval vzorkování je příliš dlouhý na zachování relevantní informace. Současně jsou data v mnoha ohledech redundantní. Ukládání dat pomocí MongoDB hodnotím převážně pozitivně. Analýza hlavních komponent naznačuje, že sběr informací o spotřebě paliva je nadbytečný, pokud současně probíhá sběr otáček, výkonu a akcelerace. Rovněž zpochybňuji využitelnost bočního zrychlení v těchto datech. Pomocí algoritmu k-means lze dobře nalézt základní struktury a vlastnosti dat. Jeho využití má však své limity; vzorky na okrajích clusterů jsou často špatně zařazeny, ideální počet clusterů není jednoznačný, velké množství odlehlých hodnot (prudké brždění) zůstalo ve stejném clusteru jako hodnoty průměrné. Cluster D je jednoznačně třídou vzorků definující neekonomickou jízdu. Přestože obsahuje spíše vzorky s vyšší rychlostí, jde o hodnotný výsledek. Podle této třídy je možné detekovat neekonomický styl řízení. Za největší nevýhodu klasifikátoru považuji zanedbání posloupnosti hodnot.

Jelikož Škoda Auto ukončila provoz aplikace OneApp, nedošlo k otestování modelu na zkušební jízdě.

5.6 Návrhy na vylepšení v rámci analýzy dat

Výsledky algoritmu k-means jsou dále využitelné. Pokud dojde k ručním úpravám a ošetřením, mohou clustery B a D sloužit jako trénovací množina pro metody strojového učení s učitelem. Prudké brždění je možné oddělit od mírného. Například označit všechny vzorky, jejich vzdálenost od centroidu je dvojnásobná nebo větší než průměr vzdáleností všech vzorků od centroidu.

Pro analýzu posloupností bych vybral n-gram. Jde o tradiční pravděpodobnostní model z oblasti počítačové lingvistiky. Model je třeba nejprve natrénovat na celých jízdách nebo jejich částech. N-gram ukládá informaci, s jakou pravděpodobností se jeden typ vzorku vyskytuje jako předchůdce druhého. Při dostatečném počtu předchůdců je možné natrénovat model, který bude obsahovat zvýšenou pravděpodobnost např. pro prudké zrychlení a prudké brždění v krátkém sledu po sobě. Pro každý jízdní styl je možné natrénovat takový model. Periodické děje pak bude možné detekovat přesněji.

5.7 Návrhy na změny v rámci systému a architektury OneApp

Systém jako celek vyžaduje optimalizaci na více úrovních abstrakce. Na úrovni MIB modulu jde především o zpracování signálů ze senzorů automobilu. Na této úrovni je nezbytné definovat

vzorkovací intervaly tak, aby nedocházelo k aliasingu a ztrátě podstatné části informace. Intervaly pro jednotlivé signály se budou pravděpodobně lišit. Ve vybraných intervalech se poté bude OneApp dotazovat na data. Druhým problémem je velký objem dat při krátkých intervalech vzorkování, který by zatěžoval paměť telefonu (v případě odesílání dat přes Wi-Fi později) nebo telekomunikační síť. Najít rovnováhu mezi těmito dvěma problémy není snadný úkol. Jeden z prvních kroků, je vybrání jen těch nejnutnějších proměnných ke sběru a analýze. Jde o komplexní úlohu pro několik expertních týmů.

Použitá literatura

Anton, H., 1987. Elementary linear algebra. Wiley, New York.

Arumugam, S., Bhargavi, R., 2019. A survey on driving behavior analysis in usage based insurance using big data. J. Big Data 6, 86. https://doi.org/10.1186/s40537-019-0249-5

Bergasa, L.M., Almeria, D., Almazan, J., Yebes, J.J., Arroyo, R., 2014. DriveSafe: An app for alerting inattentive drivers and scoring driving behaviors, in: 2014 IEEE Intelligent Vehicles Symposium Proceedings. Presented at the 2014 IEEE Intelligent Vehicles Symposium (IV), IEEE, MI, USA, pp. 240–245. https://doi.org/10.1109/IVS.2014.6856461

BSON Types — MongoDB Manual [WWW Document], n.d. . Httpsgithubcommongodbdocsblobv42sourcereferencebson-Typestxt. URL https://docs.mongodb.com/manual/reference/bson-types (accessed 5.23.20).

Cai, C., n.d. Unsupervised Detection of Drivers' Behavior Patterns 10.

Cam, L.M.L., Neyman, J., 1967. Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability: Biology and problems of health. University of California Press.

Drivewise from Allstate - Good Driver Discount [WWW Document], n.d. . Allstate. URL https://www.allstate.com/drive-wise.aspx (accessed 5.22.20).

Firican, G., n.d. The 10 Vs of Big Data [WWW Document]. Transform. Data Intell. URL https://tdwi.org/articles/2017/02/08/10-vs-of-big-data.aspx (accessed 5.22.20).

Free Educational Licenses - Community Support [WWW Document], n.d. . JetBrains. URL https://www.jetbrains.com/community/education/ (accessed 5.26.20).

Hu, J., Xu, L., He, X., Meng, W., 2017. Abnormal Driving Detection Based on Normalized Driving Behavior. IEEE Trans. Veh. Technol. 66, 6645–6652. https://doi.org/10.1109/TVT.2017.2660497

IBM Analytics - Insurance [WWW Document], 2015. URL http://www.ibm.com/analytics/vn/en/industry/insurance/ (accessed 5.22.20).

Lloyd, S., 1982. Least squares quantization in PCM. IEEE Trans. Inf. Theory 28, 129–137. https://doi.org/10.1109/TIT.1982.1056489

MongoDB Documentation [WWW Document], n.d. URL https://docs.mongodb.com/ (accessed 5.26.20).

NumPy Documentation [WWW Document], n.d. URL https://numpy.org/doc/ (accessed 5.26.20).

Overview — Matplotlib 3.2.1 documentation [WWW Document], n.d. URL https://matplotlib.org/3.2.1/contents.html (accessed 5.26.20).

Raschka, S., 2015. Principal Component Analysis [WWW Document]. URL https://sebastianraschka.com/Articles/2015_pca_in_3_steps.html

scikit-learn: machine learning in Python — scikit-learn 0.23.1 documentation [WWW Document], n.d. URL https://scikit-learn.org/stable/index.html (accessed 5.26.20).

SciPy — SciPy v1.4.1 Reference Guide [WWW Document], n.d. URL https://docs.scipy.org/doc/scipy/reference/ (accessed 5.26.20).

Shi, B., Xu, L., Hu, J., Tang, Y., Jiang, H., Meng, W., Liu, H., 2015. Evaluating Driving Styles by Normalizing Driving Behavior Based on Personalized Driver Modeling. IEEE Trans. Syst. Man Cybern. Syst. 45, 1502–1508. https://doi.org/10.1109/TSMC.2015.2417837

sklearn.cluster.KMeans — scikit-learn 0.22.2 documentation [WWW Document], n.d. URL https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html#sklearn.cluster.KMeans (accessed 4.7.20).

sklearn.decomposition.PCA — scikit-learn 0.23.1 documentation [WWW Document], n.d. URL https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html?highlight=pca#sklearn.decomposition.PCA (accessed 5.26.20).

TD MyAdvantage - Safe Driving Discount | TD Insurance [WWW Document], n.d. URL https://www.tdinsurance.com/products-services/auto-car-insurance/my-advantage (accessed 5.22.20).

What is the relation between k-means clustering and PCA? [WWW Document], n.d. . Cross Validated. URL https://stats.stackexchange.com/questions/183236/what-is-the-relation-between-k-means-clustering-and-pca (accessed 4.7.20).

Wollschläger, D., 2014. The connected car preconditions, requirements and prospects. ATZelektronik Worldw. 9, 4–9. https://doi.org/10.1365/s38314-014-0258-2

Yu, J., Chen, Z., Zhu, Y., Chen, Y., Kong, L., Li, M., 2017. Fine-Grained Abnormal Driving Behaviors Detection and Identification with Smartphones. IEEE Trans. Mob. Comput. 16, 2198–2212. https://doi.org/10.1109/TMC.2016.2618873