Univerzita Hradec Králové
Fakulta informatiky a managementu
Katedra informatiky a kvantitativních metod

Data mining a možnosti nekomerčního softwaru

Bakalářská práce

Autor: David Illner
Studijní obor: Informační Management

Vedoucí práce: prof. RNDr. Hana Skalská, CSc.

Hradec Králové Duben 2020

Prohlášení:

Prohlašuji, že jsem bakalářskou práci zpracoval samostatně s použitím uvedené literatury.

vlastnoruční podpis

V Hradci Králové dne 29.4.2020 David Illner

Poděkování:

Děkuji vedoucí bakalářské práce prof. RNDr. Hana Skalské, CSc za metodické vedení práce a za pomoc a rady při zpracování této práce.

Anotace

Bakalářská práce řeší dolování dat pomocí úloh data mining v nekomerčním softwaru Rattle. Práce vychází z metodiky Knowledge Data Discovery. Zabývá se selekcí, archivací, předzpracováním a transformací dat a klasifikačními úlohami pro vydolování znalostí. Praktická část navazuje na poznatky z teoretické části a na vytvoření vlastního modelu logistické regrese v softwaru Rattle, který je postavený na jazyku R od developera Grahama Williamse. V tomto programu byla předzpracována a transformována k vytvoření efektivního modelu. V programu byl vytvořen model logistické regrese pro předpověď počasí na daném datasetu weather.csv z depozitáře programu. Pomocí tohoto modelu byly nalezeny důležité proměnné pro předpověď, které pozitivně nebo negativně ovlivňují klasifikaci předpovědí pro každý den. Pro zhodnocení efektivnosti a kontrole správnosti předpovědí modelu byly použity metody ROC křivka a klasifikační matice.

Annotation

Title: Data mining with freeware

The bachelor's thesis describes some selected data mining tasks and possibilities of their solutions with the use of non-commercial software Rattle. An approach of the Knowledge Data Discovery methodology is used. This methodology is used in several steps: In selecting, archiving, pre-processing and transformation of data and in classification tasks for mining knowledge. The practical part of this thesis uses knowledge form theoretical part for creating own logistic regression model in Rattle software, which is based on the R language from developer Graham Williams, is used in application. In this program the data were preprocessed and transformed using knowledge from the theoretical part of the work. A logistic regression model was used to forecast the weather on a given dataset. Using this model, important variables were found for the forecast that positively or negatively affect the classification of forecasts for each day. For evaluation process of model, the ROC curve and Error matrix methods were used to get clear understanding of the effectiveness of the model predictions.

Obsah

Seznam obrázků

Seznam tabulek

Seznam grafů

1 Úvod

Tato bakalářská práce se zabývá problematikou oblasti data mining a možností použití nekomerčního softwaru při dolování dat. Oblast data mining se těší poměrné popularitě v dnešní době kvůli velikému rozmachu používání informačních technologií, kde jsme obklopeni neuvěřitelným množstvím dat. Máme proto potřebu tyto data nějak zpracovat a „vydolovat" z nich znalosti, které můžeme použít v našem každodenním životě. Počátky oblasti data mining sahají až do roku 1960, kdy se jednalo zejména o metody ukládání dat v souborech a další zpracování těchto souborů. [1] Dnešní doba nabízí díky pokročilým technologiím, jako jsou datové sklady a databáze mnohem více možností jak sbírat, ukládat a zpracovávat data. Došlo také k rozmachu v komunikačních, archivačních, výpočetních technologiích a tím k vyhledávání možností a realizací pro jejich využití. Rozvoj také nastal v oblasti získání, archivace i zpracování dat nenumerických a nestrukturovaných (obrazových, textových, geografických i dalších typů). Možnosti dat a nutnost jejich předávání nebo kooperace mezi různými systémy si vyžádaly vyvinutí jistých standardů. Řešení typu data mining má jako obecný cíl zpracování různých typů dat včetně nestrukturovaných a jejich převedení pomocí vhodných postupů do formy pro podporu rozhodování a pro vytváření dalších nových znalosti. Nové informace z dat mohou pomoci při rozhodování nebo dokonce předpovídat chování sledovaného objektu do budoucna. Data mining jako oblast prosakuje do spousty jiných odvětví. Znalosti, které můžeme získat z dat, je možné používat v marketingovém, finančním nebo medicínském odvětví.

V teoretické části této práce bude představena metodika vyhledávání znalostí z dat (Knowledge Data Discovery, KDD). Podle této metodiky budeme postupovat a představovat si jednotlivé kroky, které je třeba udělat, abychom zajistili efektivní dolování znalostí z dat. Tato část je hlavně zaměřena na preprocessing dat a na typy úloh data mining.

V praktické části bude představen nekomerční software Rattle, který je postaven na jazyku R. Bude vysvětleno, jak probíhá práce v tomto programu a jak lze přenést teoretické poznatky do praxe. Jaké jsou zde nástroje pro řešení programů a nástroje pro vizualizaci dat, které jsou potřebné k analýze dat. Také bude nazna-

čeno, jak vytvořit model pomocí jednoho z typů úloh oblasti data mining a možnosti validace a kontrole vytvořené modelu pomocí nástrojů, které nám program nabízí. Jako je ROC křivka a klasifikáční matice pro viditelnost efektivity modelu. Tímto poté můžeme objektivně posoudit, zdali vytvořený model funguje a správně dokáže zařadit data z minulých měření, které jsou v testing sample. Podle tohoto výsledku můžeme očekávat, jak bude model v budoucnu zařazovat záznamy.

2 Vznik a vývoj data miningu

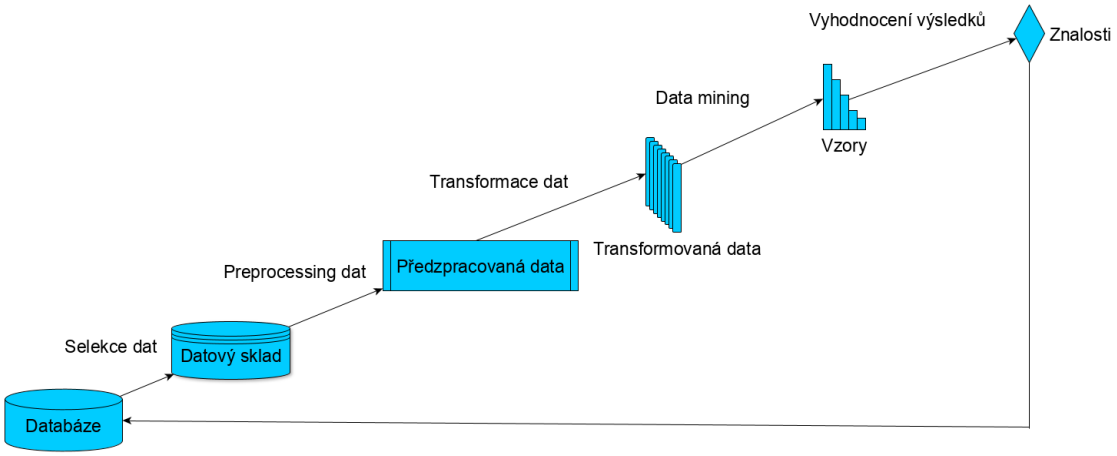
V posledních letech se dostává velké pozornosti data miningu kvůli velkému množství dat a kvůli potřebě dostávat z těchto dat užitečné informace. Pojem Data mining označuje činnost, při které extrahujeme (mining) znalosti z velkého množství dat.[1] Mining by se mohlo přeložit jako „těžení“, „dolování“, které lépe pomáhají pochopit, co se s daty děje při této činnosti. Tento termín data mining prvně použil Tukey v roce 1970. Popisoval zde intenzivní statistické analýzy, jež měly za úkol vyhledávat souvislosti mezi daty. Po nalezení a ověření těchto souvislostí bylo možné vyslovit hypotézy, které se poté ověřovaly. [2]

Data mining se také spojován s pojmem Knowledge Discovery in Databases neboli KDD. V diagramu KDD je vidět, že Data mining jen jedním krokem k získání znalostí z velkého množství dat. KDD je typ metodiky, která vyhledává znalostí z dat a formuluje data do přehledných výsledků pro prezentaci znalostí extrahovaných z databází. Cílem tohoto procesu je nalezení klíčové informace, která pomůže uživateli se rozhodnout efektivně. KDD je označováno jako netriviální proces identifikování validních vzorů v datech, jež dosud nebyly známé. Vzor je podmnožina dat, jež má nějaké společné vlastnosti. Pro efektivní vymezení vyhodnovacích kritérií má KDD několik kroků, které se používají pro účinné extrahování znalostí z databází. [2]

1. **Data Selection** – Selekce dat, která jsou relevantní pro naši potřebu. Vybírá se zde i zdroj těchto dat, jaké databáze se budou používat.
2. **Data Cleaning** – V tomto kroku je odstraněn šum a neúplná data.
3. **Data Integration** – Zde je možno nakombinovat více zdrojů dat.
4. **Data Transformation** – Data jsou změněna do forem, které jsou efektivní pro dolování. Například sumarizace a agregace.
5. **Data Mining** – Proces, který doluje vzory/modely a poté je reprezentuje ve srozumitelné formě.
6. **Pattern Evaluation** – Prezentace nalezených vzorů a hodnocení.
7. **Knowledge presentation** – Uplatněný znalostí, jež byly extrahovány.

[1][2]

Kroky 1 až 4 jsou různé formy Data Preprocessing. Zde jsou data připravována k extrahování. Jak je vidět v seznamu krok 4 neboli data mining je opravdu jediným krokem v KDD. Byl zařazen do KDD v 80. letech 20. století do té doby byl data mining samostatným oborem. [1]



Obrázek 1 Grafické zobrazení KDD
Zdroj: Vlastní zpracování

Analýza dat a jejich skladování se rozvíjely současně. Začátek byl v roce 1960, kdy se jednalo o obyčejné zpracování a archivaci souborů. V současnosti máme nové technologie pro skladování dat. Jako jsou například databáze založené na XML, datové sklady a OLAP nebo objektově orientované databáze. [1][2]

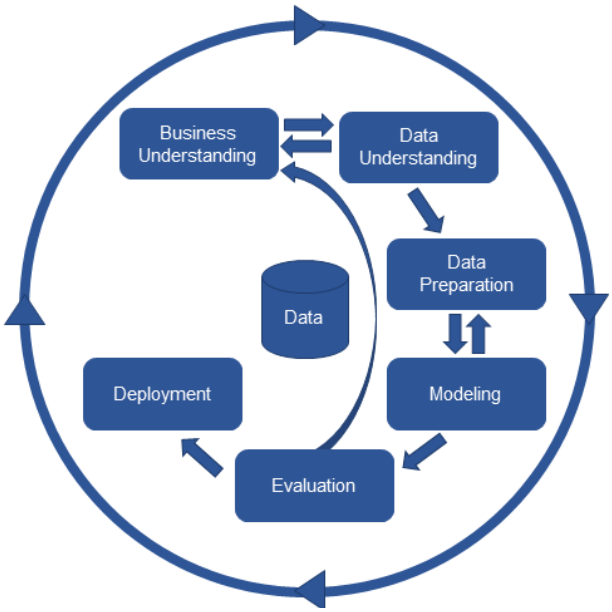
Dnes můžou být data uložena v mnoha různých databázích. Tento neustálý nárůst digitálních dat je jedním z největších důvodů k rozmachu oblasti data mining. Na všechna data se pohlíží jako na potencionální zdroje znalostí. Data mining vychází z empirických dat a detailní analýzy jednotlivých případů a vztahů mezi proměnnými. Po nalezení vztahů mezi vybranými proměnnými jsou vytvářeny takzvané patterns neboli vzory dat. Protože je množství dat opravdu obrovské, existují metodologie pro práci s daty a pro následné procesy extrahování informace z dat, které zajišťují normalizaci napříč odvětvími. Mezi tyto metodologie patří CRISP-DM, SEMMA a KDD. [2]

2.1 CRISP-DM

CRISP-DM neboli cross-industry proces for data mining v překladu „meziodvětvový proces pro dolování dat“. Tato metodologie předkládá strukturu pro naplánování data miningového projektu. Jedná o sekvenci 6 kroků. [3]

1. **Bussines understanding** - porozumění, co chceme jako firma dokázat; porozumění naší přítomné situaci, zavedení cílů pro projekt
2. **Data understanding** - Získání potřebných dat k projektu a následné porozumění, co tato data představují
3. **Data preparation** - Zde nastává práce s daty, které jsme vybrali. Jedná se o čistění dat a jejich integraci.
4. **Modeling** - Vybrání metody pro modelování. Může se jednat o rozhodovací stromy nebo nerurální sítě
5. **Evaluation** - Vyhodnocení podle kritérií, které jsme si nastavili v prvním kroku. Zda je tento model pro nás efektivní.
6. **Deployment** – V tomto kroku vycházíme z předešlého kroku. Hodnotí se zde výběr strategii, jež nám nejlépe pomůže nasadit tento model.

[3]



Obrázek 2 Grafické zobrazení CRISP-DM

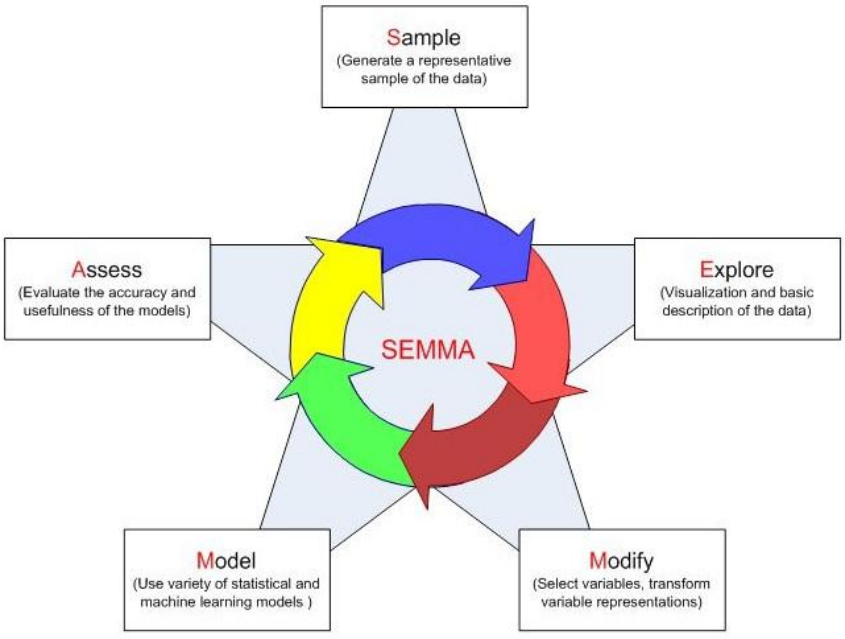
Zdroj: https://www.kdnuggets.com/wp-content/uploads/crisp-dm-4-problems-fig1.png

2.2 SEMMA

Tato metodika nebo spíše organizace funkcí byla navržena firmou SAS pro jejich software SAS Enterprise Miner. SEMMA je navržena speciálně pro tento software. Jsou zde opomenuty kroky jako Business a Data understanding, jelikož se více zaměřuje na software od firmy SAS a vývoj modelu. Metodika SEMMA opět funguje jako cyklus. Jedná se o akronym pro Sample, Explore, Modify, Model a Assess. [4]

1. **Sample** - Selekce, identifikace, sloučení, oddělení vstupních dat do tabulek.
2. **Explore** - Prozkoumávání dat pomocí grafického a statistického znázornění. Dochází zde k identifikaci důležitých proměnných pro grafické znázornění.
3. **Modify** - Příprava dat k analýze. Je zde možno vytvářet a transformovat proměnné. Provádění analýzy clusterů a analýzy dat.
4. **Model** – Použití různých modelů, na již připravená data, jako jsou rozhodovací stromy nebo neuronové sítě.
5. **Assess** – Vyhodnocení přechozích kroků pro znázornění výsledků efektivity daného modelu, který byl vybrán.

[4]



Obrázek 3 Grafické zobrazení SEMMA

Zdroj: https://medium.com/datadriveninvestor/data-science-project-management-methodologies-f6913c6b29eb

2.3 Porovnání CRISP-DM a SEMMA

Obě metodiky jsou velmi podobné, ale jak už bylo řečeno v předchozí kapitole SEMMA je určena přímo pro daný software od firmy SAS a na rozdíl od metodiky CRISP-DM opomíjí kroky Business Understanding a Data Understanding. Soustředí se přímo na primární věci, které se týkají data miningu. Proto použití metodiky SEMMA mimo software firmy SAS Enterprise Miner je nevhodné. Je zde ovšem podobná logika v některých krocích. Například kroky Modify a Data Preparation, Model a Modeling a nakonec Assess a Evaluation. Oba modely pracují jako smyčky nežli jako nějaký přímočarý postup. [5] Metody jsou porovnány v následující tabulce.

Tabulka 1 Porovnání CRISP-DM a SEMMA

<table>
<tr><th></th><th>CRISP-DM</th><th>SEMMA</th></tr>
<tr><td rowspan="3">Výběr dat</td><td rowspan="2">Bussines Understanding</td><td rowspan="2">----------</td></tr>
<tr></tr>
<tr><td rowspan="2">Data Understanding</td><td>Sample</td></tr>
<tr><td>Explore</td></tr>
<tr><td rowspan="2">Úprava dat</td><td>Data Preparation</td><td>Modify</td></tr>
<tr><td>Modeling</td><td>Model</td></tr>
<tr><td rowspan="2">Vyhodnocení dat</td><td>Evaluation</td><td rowspan="2">Assess</td></tr>
<tr><td>Deployment</td></tr>
</table>

Zdroj: https://medium.com/datadriveninvestor/data-science-project-management-methodologies-f6913c6b29eb

3 Vysvětlení pojmů oblasti data mining

3.1 Datové sklady

Po výběru dat, se kterými budeme pracovat, přijde na řadu uložení dat do datových skladů. Tyto sklady mají spíše informační hodnotu a provádějí analýzy na rozdíl od obyčejných databází, které provádějí transakční operace. Datové sklady jsou v dnešní době používány zejména díky nástrojům, které jsou pod názvem Online Analytical Processing zkráceně OLAP. Technologie OLAP se dá definovat pomocí zkratky FASMI (Fast Analysis of Shared Multidimensional Information) neboli rychlá analýza sdílené multidimenzionální informace. [6] Jak bylo dříve řečeno tyto sklady, mají hlavně informační hodnotu, díky analýze, kterou provádějí na datech, jež jsou uložena v datových kostkách. Datové sklady jsou založeny na multidimenzionálním datovém modelu, který pro ukládání dat používá datové kostky. Pokud se bavíme o dimenzích ve vztahu k datovým kostkám, jsou myšleny, jako vybrané entity, které jsou dlouhodobě sledovány. Jedny z nejpoužívanějších operací OLAP jsou operace [1]:

- **Drill-down** – sestupné dopracování k více přesným výsledkům
- **Drill-up** – přecházení vzestupně ve vztahu k dimenzím a následná agregace výsledků

OLAP nenabízí jen jeden typ serverů. Je na výběr o několik více pro potřeby koncového zákazníka.

- **ROLAP** – Relační OLAP. Tyto servery jsou mostem mezi back-end servery a front-end klientskými nástroji. Využívají relační databáze.
- **MOLAP** – Multidimenzionální OLAP. Tento server nabízí zrychlené vyhledávání dat a to díky indexování.
- **HOLAP** – Hybridní OLAP je kombinací ROLAPu a MOLAPu. Využívá plusů obou těchto serverů,

[1][2]

3.2 Příprava dat

Po uložení dat do datových skladů přijde na řadu krok data preprocessing v překladu předzpracování dat. Toto je velmi důležitý krok v KDD. Jelikož jsou databáze v reálném světě, kvůli jejich velikosti, velmi náchylné na šum, chybějící nebo neúplná data. Šum je zde myšlen jako atribut, který obsahuje chybu nebo outlier (hodnota, jež se liší od ostatních mnohonásobně). Neúplná data jsou takové, která nemají hodnotu. Zde platí, fráze GIGO neboli „Garbage in, garbage out" pokud vybíráme data, jež nedávají smysl nebo jsou neúplná potom, dostáváme na výstupu také nesmyslná a neúplná data. Důvodů pro tyto chyby je v reálném světě několik. Například v době záznamu nebyli tyto informace viděni jako potřebné, informace byli smazány, selhání IT nebo modifikace databáze a předchozí záznamy už nejsou kompatibilní. Proto byl zaveden krok v KDD Data preprocessing, který nám pomáhá si data předpřipravit a zbavit se nesmyslných dat, šumu apod. Když jsou tyto rutiny zavedeny před krokem data mining, je zkrácen čas a efektivnost extrahování dat a vzorů ze zdrojů. Bylo zavedeno několik rutinních kroků, které nám pomohou se s těmito problémy vypořádat. [1]

3.3 Čištění dat

Jak jsme si již řekli data v databázích v reálném světě, mohou obsahovat šum, neúplná nebo nekonzistentní data. Proto byla vytvořena rutina Data Cleaning, která doplňuje chybějící hodnoty, odstraňuje šum a napravuje nekonzistentní data. Jde nám o to, aby byla data co nejkvalitnější. Data ovšem musí být platná, a aby nedošlo ke GIGO, musí být také co nejvíce přesná, kompletní a jednotná, aby bylo dosaženo nejlepších výsledků. Ovšem v reálném světě není takto dokonalý systém záznamu nikde. Proto existuje rutina Data Cleaning. K pochopení problematiky Data Cleaningu si nejdříve popíšeme nejvíce využívané rutiny a jak pracuje s daty, pokud se vyskytne nějaká z výše zmíněných chyb. [1]

Můžeme narazit na problém, že nám chybí záznam v nějakém atributu je tu několik možností, které lze provést pro úplnost dat.

1. **Ignorovat chybějící záznam** – Tato metod není velmi efektivní, jelikož vůbec nenapravuje problém, s jakým se potýkáme.
2. **Doplnit chybějící hodnotu ručně** – Tato metoda je velmi časově náročná a někdy může být až nemožná vzhledem k množství dat.
3. **Použití globální konstanty** – Do všech chybějících záznamů můžeme doplnit globální konstantu, ale riskujeme, že při dolování zde najdeme nějaký vzor mezi doplněnými záznamy.
4. **Doplnění průměrné hodnoty** – Pokud chybějící hodnota je číselná, můžeme zde doplnit průměr, vypočtený z ostatních hodnot.
5. **Použití průměru** – Použijeme stejný průměr pro vzorky, které spadají pod stejnou n-tici.
6. **Doplnění nejpravděpodobnější hodnoty** – Tohoto může být dosaženo pomocí regrese nebo rozhodovacích stromy, které jsou popsány v dalších kapitolách.

[1]

Metody 3 až 6 nejsou velmi účinné, jde zde o usměrnění dat. Doplněné hodnoty nemusí být pravdivé a budou mít za následek zkreslení výsledku. Metoda 6 je prozatím nejvíce populární, využívá nejvíce předložených dat k doplnění chybějící hodnot v záznamech. [1] Ovšem všechno směřuje ke špatnému navržení databází.
Pokud se snažíme tyto chybějící záznamy odhalit lze provést pomocí kontrolního součtu, kdy porovnáme nynější databázi se starší verzí. [2] Je třeba dobře navrhnout databázi, aby se v později minimalizovalo riziko těchto chyb.

Dalším problémem je nesprávnost dat, neboli šum považujeme případ, kdy se v záznamu atributu vyskytne chyba. Příčin může být mnoho například duplikátní záznamy, chyba při zápisu do systému. [2] Tyto chyby se snažíme vymazat za pomocí následujících rutin.

- **Binning** – Metoda Binning transformuje proměnné do kategorií. Binning vychází z anglického slova bin neboli koš, kdy rozdělujeme proměnné do jednotlivých košů. Existuje několik možností rozdělení.
- **Regrese** – Pomocí lineární regrese, kde vložíme data do funkce a ta najde nejlepší způsob jak z jednoho atributu, lze předpovědět další.
- **Clustering** – Podobné hodnoty jsou uzavřeny do podmnožin neboli tzv. clusterů. Pokud se vyskytují nějaké hodnoty mimo tuto oblast, může se jednat o outlier.

[1]

3.4 Integrace dat

Integrace dat je proces, při kterém čerpáme data z více zdrojů a kombinujeme do jednoho velkého zdroje, který později používáme. Vyplívá zde otázka, jak můžou být dvě a více odlišných zdrojů dat spolu kompatibilní? Bohužel nejsou, tento problém se nazývá entity identification problem. Další problémem jsou redundantní data. Pokud pracujeme s více zdroji, je zde šance, že se některá data objeví vícekrát. Třetím problémem je the detection and resolution of data value conflicts, je to problém, kdy je stejný atribut reprezentován v různých zdrojích různými způsoby (jméno Adam Novák může být zapsáno několika způsoby jako Novák Adam, A. Novák). Tento problém lze vyřešit pomocí transformace dat. [1]

3.5 Transformace dat

Transformace dat nám napomáhá k efektivnějšímu procesu data miningu. Jelikož jsou data seskupena do celků pro přehlednější práci a rychlejší orientování. Tato rutina má pod sebou několik procesů. [1]

1. **Vyhlazování** – V tomto procesu jde o redukci vlivu náhodných výkyvů hodnot ze zdroje informací. Vhodné v situaci, kdy detekce extrémů není cílem analýzy.
2. **Agregace** – Seskupení dat, která mají nějakou společnou vlastnost (například celkový počet získaných kreditů určitého studenta za akademický rok).
3. **Zobecnění** – Používá princip dědičnost (předek, potomek) skrze hierarchii ve zdroji informací.
4. **Normalizace** – Data jsou škálována, aby jejich hodnoty byly v určitém rozmezí. Například na interval (0, 1) pro lepší klasifikaci, protože se poté pracuje na menším intervalu než původní interval (například příjmy zaměstnanců).
5. **Vytvoření nového atributu** – Vytvoření nových atributů pro zefektivnění data miningového procesu.

[1]

3.6 Data Reduction

Pokud máme množství dat, můžeme na něj aplikovat proces Data Reduction. Tato rutina by měla mít za výsledek redukování objemu, ale přesto zachovat integritu původního množství dat. Můžeme se například zbavit záznamů, při kterých nebyla zaznamenána všechna měření kvůli selhání systému nebo seskupit triviální data do jednoho atributu. Nyní by mělo být extrahování informací pomocí data mining být na tomto zmenšeném množství dat, mělo vyhodnotit stejný výsledek, ale za kratší dobu a mnohem efektivněji. Můžeme využít několik možností pro redukci dat, které jsou například. [1]

1. **Snížení dimenze** – mechanismy, jež nám pomáhají redukovat velikost datasetů.
2. **Výběr podmnožin atributů** – odstranění triviálních záznamů nebo dimenzí
3. **Agregace dat** – Při vytváření datové krychle jsou seskupována některá data do jednoho atributu.
4. **Snížení různorodosti** – některá data jsou nahrazena datovými reprezentacemi nižší úrovně, když pokud má atribut velké množství variant, jejichž rozlišování není pro danou úlohu podstatné (modely). Může zde být použitá metoda shlukování (clustering) nebo výběru (sampling).

[1][7]

3.7 Kategorizace dat

Kategorizace (diskretizace) dat funguje na principu rozdělení veličiny, měřené na spojité stupnici hodnot do jednotlivých intervalů. Daný atribut záznamu ze zdroje je překódovaný kód (label) kategorie, do které přísluší svou hodnotou. Dochází ke zmenšení variant různých hodnot na daném atributu původního data setu. [1]
Diskretizace rozdělujeme podle směru rozkládání top-down nebo bottom-up metoda.

- **Top-Down** – Metoda začíná nalezením tzv. split pointu odkud poté je atribut rozdělen na intervaly. Takto se proces opakuje. Metody jsou buď rozděleny na daný počet intervalů stejné délky, nebo na intervaly se stejnými četnostmi
- **Bottom-up** - Metoda začíná posouzením všech spojitých hodnot jako potenciálních split pointů, sloučí sousední hodnoty a analyzuje, zda se sousední hodnoty se liší statisticky významně. Postupně vyhledá podmnožiny, které mají významnou souvislost s některým z cílových atributů.

[1]

K nejpoužívanějším metodám diskretizace pro číselné hodnoty se používají metody Binning, histogram analysis a cluster analysis. Metoda Binning jak bylo již popsáno dříve, rozdělí atributy na určité kategorie pomocí „bins“ neboli „přihrádek“ nebo „košů“. Tato metoda funguje podobně Top-down metoda. Metoda Histogram Analysis rozdělí atribut na intervaly, buď stejné délky, nebo se stejnou frekvencí výskytu (rozdělení příjmu zaměstnanců po 10 000 Kč). Po použití této metody se snažíme dosáhnout intervalů na stejnou délku nebo frekvenci výskytů. Můžeme tento proces opakovat několikrát pro vytvoření jakési hierarchie (například interval příjmů zaměstnanců 10 000 - 20 000 může být dále rozdělen na interval 10 000 – 15 000 a 15 000 – 20 000). Takto můžeme velmi rychle kategorizovat data pro lepší přehlednost a efektivitu. [1]

4 Typy úloh data mining

V této kapitole jsou představeny typy úloh, které nám pomáhají v dolování znalostí z dat. Je to další krok v KDD neboli samotný krok data mining, kdy už z dat dolujeme znalosti/vzory (patterny). Tyto úlohy zahrnují klasifikaci a diskriminaci, regresi, predikci, segmentaci a asociační pravidla. Nejvíce se budeme věnovat úloze klasifikace, která bude použita i v praktické části.

4.1 Klasifikace a diskriminace

„Klasifikace je proces zařazení nového prvku do některé skupiny pomocí sestaveného modelu a pravidel klasifikace“ [2].

„Diskriminace znamená rozlišení. Sleduje otázku, zda je možnost formalizovat proces zařazování prvků do skupin, které se odlišují některými pozorovanými vlastnostmi. “ [2]

Úloha klasifikace se zabývá daty, která jsou kvalitativní. Tudíž velké množství dat, jež se dělí do skupin podle nějaké společné vlastnosti, společné znaky nebo se shodným vývojem. Tato úloha kategorizuje daná data pomocí atributů (klasifikátorů), jež pomáhají vymezit podobnosti mezi jednotlivými daty. Také vymezuje počet kategorií, do kterých bude možno rozdělit jednotlivá data a vymezí důležité prediktory pro model. [2][8]

Pro vytvoření klasifikačního modelu je třeba množina atributů, které popisují objekt a umožní klasifikaci a pravidla (klasifikační model), pomocí kterých budou další objekty zařazeny do příslušných kategorií. [2]

Kategorie (skupiny objektů, které se mezi sebou liší určitými vlastnostmi), mohou být známé. Potom pro zjištění, do které kategorie náleží nový prvek, se využívá úloha diskriminace. Diskriminační model je sestavený na učícím souboru, ve kterém je známá příslušnost ke kategorii. Pro zařazení nového objektu musí být známé atributy, které používá sestavený diskriminační model a pravidla pro zařazení do kategorie (klasifikaci). Model může být sestavený pomocí lineární kombinace atributů, která umožní největší možnou diskriminaci mezi objekty. [2]

Tudíž pomoci diskriminace sestavíme klasifikátory, podle kterých lze mezi sebou jednotlivě rozeznat kategorie, a poté pomocí klasifikace začneme se zařazením no-

vých objektů, které ještě nebyly zařazeny. Pro vyhledávání těchto klasifikátorů je mnoho způsobů. Pro tuto práci byly vybrány tři z nejvíce používaných způsobů a to jsou: logistická regrese a lineární diskriminační analýza. Metoda K-nearest neighbors neboli KNN umožní naopak nalézt různé kategorie, tedy podmnožiny prvků (pokud nejsou známé, ale existují) takové, které se navzájem liší svými vlastnostmi. [2][8]

4.1.1 Logistická regrese binární

V případě použití logistické regrese v klasifikáční úloze může dojít k dvěma výsledkům mezi jednou a nulou, proto se jedná o logistickou regresi binární. Pro tuto metodu je nutné omezit výsledky na interval (0,1) pro jakoukoliv hodnotu X, která bude dosazena. Jelikož reálná hodnota X nemusí mít lineární průběh na celém intervalu hodnot. Některé hodnoty by mohli překročit hranice, pokud bychom použili lineární regresi, proto je použita logistická regrese, která vymezuje výsledky mezi nulou a jednou pomocí pravděpodobností. [2][8]

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1 + e^{\beta_0+\beta_1 X}}$$

Proměnná X může nabývat jakoukoliv hodnotu a je omezena na intervalu (0,1). Rovnice v tomto tvaru bude ukazovat pro malé hodnoty pravděpodobnost blížící se nule, ale nikdy pod nulou a to stejné pro velké hodnoty, kdy se pravděpodobnost bude blížit jedné, ale nikdy více jak jedna. Koeficienty logistické regrese β_0 a β_1, jsou prozatím neznámé a je třeba je vypočítat pomocí metody maximální věrohodnosti. Hledáme takové odhady β_0 a β_1, které maximalizují funkci p(X). To znamená, že se přibližují nule nebo jedné. Funkce $l(\beta_0,\beta_1)$ je logaritmus věrohodnostní funkce. [2][8]

$$l(\beta_0, \beta_1) = \prod_{i:y_i=1} p(x_i) \prod_{i':y_{i'}=0} (1 - p(x_{i'}))$$

4.1.2 Logistická regrese pro více proměnných

Dosud jsme počítali jen s binárním rozhodnutím, kdy existují pouze dvě možnosti. V této části bude popsána logistická regrese pro více proměnných. Je zde myšleno více prediktorů, které zařazují daný objekt do kategorií. Pro tuto variantu je zde tato generalizace pro logistickou regresi více proměnných. Kde *p* je prediktor. Koeficienty β_0 až β_p se opět vypočítají pomocí metody maximální věrohodnosti. [8]

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}{1 + e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}$$

Pokud používáme logistickou regresi binární, je zde možnost, že zanedbáme prediktory, které nám mohou pomoci zpřesnit výsledek. Je možné, že výsledky se budou lišit, pokud použijeme více prediktorů na rozdíl při použití jednoho prediktoru. Hlavně pokud je mezi těmito prediktory nějaká souvztažnost. Tento jev se poté nazývá confounding. [8]

4.1.3 Lineární diskriminační analýza

Tato klasifikační metoda používá jiný přístup nežli logistická regrese. Metoda lineární diskriminační analýzy (LDA) modeluje distribuci každého prediktoru X zvlášť ke skupině Y ke které byl tento prediktor přiřazen a na konec použijeme Bayesovu větu pro změnu těchto výsledků na odhady pro Pr(Y = k| X = x). Tato metoda je velmi populární díky své stabilitě při výpočtech. [8]

Uvažujme klasifikaci K skupin (kde K >= 2). Nechť π_k představuje celkovou nebo apriorní pravděpodobnost, že náhodně vybrané pozorování je z k-té třídy neboli je z k-té kategorie z proměnné Y. Kde $f_k(x)$ = Pr(X = x| Y = k) značí hustotu funkce pro proměnnou X, která se nachází k-té kategorii. [8]
Tudíž podle Bayesovy věty je tento vzorec:

$$\Pr(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_1 f_1(x)}$$

Ačkoliv je tento způsob klasifikace pomocí Bayesovy věty, kde $p_k(X) = \Pr(Y = k|\, X)$ je velmi rozsáhle má nejmenší míru chyb ze všech možných metod pro klasifikaci. Pravděpodobnost $p_k(X)$ je označována jako posteriorní pravděpodobnost, jelikož udává jakou pravděpodobnost má sledované X = x, že náleží do k-té kategorie, vzhledem k hodnotě prediktoru v určitém pozorování. [8]

4.1.4 Lineární diskriminační analýza pro p = 1

Pokud nastane situace, kdy máme ke klasifikaci jenom jeden prediktor. Musíme zde vypočítat odhad funkce $f_k(x)$ abychom mohli odhadnout $p_k(X)$. Pomocí tohoto odhadu zařadíme objekty do k-té kategorie pro kterou má největší pravděpodobnost. Dojde zde na změnu apriorní (původní) pravděpodobnosti na aposteriorní pravděpodobnost. [2][8]

Budeme předpokládat, že $f_k(x)$ má normální nebo Gaussovské rozdělení, kde μ_k *a* σ_k^2 je střední hodnota a rozptyl v k-té kategorii. Po úpravách ze vzorce pro LDA dojdeme k tomuto vzorci, který nám pomůže vypočítat aposteriorní pravděpodobnost pozorování X = x. Zde je $\delta_k(x)$ diskriminační funkcí. [8]

$$\delta_k(x) = x \cdot \frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \log(\pi_k)$$

Pokud dojde k případu, že K = 2 a $\pi_1 = \pi_2$. Myšleno, že existují 2 kategorie, které se překrývají a není možné určit kam zařadit prvek. Existuje hranice Byesova rozhodnutí, kde bod patří do kategorie 1, pokud $2x(\mu_1 - \mu_2) > \mu_1^2 - \mu_2^2$ a do kategorie 2 pokud se bude jednat o opak. [8]

$$x = \frac{\mu_1 + \mu_2}{2}$$

V LDA opět musíme vypočítat odhady pro klasifikaci pomocí Baysova vzorce a to jsou odhady pro $\pi_k, \mu_k \; a \; \sigma^2$. Kde n je celkový počet pozorovaných objektů a n_k je celkový počet pozorovaných objektů v k-té kategorii. Tyto odhady se poté dosadí do předchozího vzorce pro LDA s 1 prediktorem. [8]

$$\hat{\pi}_k = \frac{n_k}{n}$$

$$\hat{\mu}_k = \frac{1}{n_k} \sum_{i:y_i=k} x_i$$

$$\hat{\sigma}^2 = \frac{1}{n-K} \sum_{k=1}^{K} \sum_{i:y_i=k} (x_i - \hat{\mu}_k)^2$$

4.1.5 Lineární diskriminační analýza pro p > 1

V této kapitole se budeme zabývat LDA, která má více prediktorů pro zařazování sledovaných objektů do kategorií. Tato metoda předpokládá, že známe hustotu rozdělení pravděpodobností. [8]

Nechť X = (X_1,X_2,…,X_p) je z multivariačního Gaussovského rozdělení s vektorem středních hodnot pro specifickou kategorii a také obsahuje společnou kovarianční matici. Pro vyznačení, že proměnná X má Gaussovo rozdělení ji značíme v zápisu takto: $X \sim N(\mu, \Sigma)$, Kde μ je střední hodnota proměnné X a Cov(X) = Σ neboli kovarianční matice, která je symetrická matice o velikosti $n \times n$, kde dva prvky mají mezi sebou lineární závislost, která platí pro všechny kategorie. [2][8]

$$f(x) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} exp\left(-\frac{1}{2}(x - \mu_k)^T \Sigma^{-1}(x - \mu_k)\right)$$

Pro zjištění diskriminační funkce stačí tento vzorec poupravit a vyjde nám:

$$\delta_k(x) = x^T \Sigma^{-1} - \frac{1}{2}\mu_k^T \Sigma^{-1}\mu_k + \log \pi_k$$

Výsledek tohoto vzorce zařazuje sledovaný prvek do správně kategorie a to je taková, která má největší výsledek pro X = x. [8]

4.1.6 Kvadratická diskriminační analýza

V předchozí kapitole jsme brali v potaz Cov(X) = Σ neboli kovarianční matici, která byla pro všechny kategorie stejná. V této kapitole si popíšeme jak klasifikace pomocí Bayse funguje, pokud jsou tyto kovarianční matice rozdílné.

Kvadratická diskriminační analýza (QDA) nám v tomto ohledu pomůže. Metoda QDA stejně jako LDA funguje na principu, že klasifikuje prvky, které mají Gaussovské rozdělení tedy $X \sim N(\mu, \Sigma)$. Je tu ovšem jeden rozdíl a to jen ten, že QDA také počítá s tím, že každá kategorie má svou vlastní kovarianční matici v zápisu je to tedy označeno $X \sim N(\mu_k, \Sigma_k)$, zde je Σ_k tedy kovarianční matice pro danou k-tou kategorii. [8]

$$\delta_k(x) = -\frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1}(x - \mu_k) + \log \pi_k$$

Opět je sledovaný prvek zařazen do kategorie, pro kterou je výsledek největší.
Metoda QDA je velmi flexibilní metoda na rozdíl od LDA, kde při větším množství odhadů může dojít ke zkreslení výsledků. Metoda LDA je doporučena pokud máme množství prvků s menším počtem prediktorů, aby byly možné chyby co minimální. Metoda QDA se naopak uplatní, pokud máme prvky s velkým množstvím prediktorů. [8]

4.1.7 K-Nearest Neighbors

Další metodou pro klasifikaci je metoda K-Nearest Neighbor (K-nejbližší soused) neboli KNN. Při předpovídání rozdělení prvků do kategorií bychom vždy měli používat Bayese, ale u dat která pocházejí z reálného světa, kterých je opravdu velké množství, nemáme tu možnost vždy znát podmíněné rozdělení pro (P (Y|X)). KNN dostalo název podle způsobu výpočtu, kdy odhadneme podmíněné rozdělení pomocí jeho sousedů. Tento výpočet má hodnotu K, která je kladná a udává kolik sousedů, bude zahrnuto do výpočtu a vybraný prvek x_0. Pří výpočtu zjistíme K body, které jsou nejblíže k prvku x_0. Výsledkem je odhadované podmíněné rozdělení pro kategorii j. Tento odhad se následně použije v Bayesu a

prvek x_0 je zařazen do kategorie s největší pravděpodobností. Důležitým prvkem toho výpočtu je hodnota K, podle této hodnoty je vypočítaný klasifikátor méně čí více flexibilní. [8]

$$\Pr(Y = j|X = x_0) = \frac{1}{K}\sum_{i\in N_0} I(y_i = j)$$

Tato metoda prozkoumá počet K sousedů, kteří jsou zde zastoupeni jako N_0. Metoda KNN se může jevit jako velmi jednoduchá metoda, ale její výsledky klasifikátorů mohou být blízko klasifikátorům Bayse. [8]

5 Návrh řešení ve zvoleném softwaru

Pro praktickou část práce byl zvolen nekomerční software Rattle, který je vytvořen v statistickém programovacím jazyce zvaném R. Tento program je od developera a autora Grahama Williamse, který napsal knihu a několik tutoriálů, které nám pomohou v návrhu řešení pro úlohy popsané v předchozí části v oblasti data mining. Bude použita stabilní verze programu Rattle 5.3.0, která byla potvrzena jako funkční na operačním systému Windows k datu 18. 3. 2020. Rattle jakožto nekomerční software je dostupný pro všechny bez poplatků.

Rattle byl vyvinut v roce 1997. V první verzi používal programovací jazyk Python. Později však začal používat R, jelikož tento jazyk je postaven na procedurálním programovacím jazyce C, který umožňuje rychlé výpočty. Také je méně citlivý pokud používáme více programovacích jazyků najednou. [9]

Výběr programu Rattle pro praktickou část je z hlavního důvodu kvůli tomu, že je to jeden z velmi efektivních nekomerčních softwarů a spadá pod licenci GNU General Public Licence pro volné používání. Díky této licenci nejsou zde žádné překážky při používání. Autorská práva k tomuto softwaru drží The R Foundation for Statistical Computing. Rattle je také open source, takže jeho zdrojový kód je k nahlédnutí nebo úpravě. Někdo může toto vidět jako velké mínus, ale Rattle je peer-reviewed software a každá jeho úprava je zkoumána a různé bugy jsou vyřešeny velkou komunitou statistiků, kteří se podílejí na vývoji programu. Dalším důvodem je, že je postaven na jazyku R, který je vytvořen přímo pro statistické účely. Programovací jazyk R je nejvíce rozsáhlý statistický jazyk, který je dostupný volně na trhu. Tento jazyk má tým 19 statistiků, kteří se starají o jeho udržování. Pokud máme jiné nároky nežli data mining existuje přes 4800 balíčků, které jsou nám volně k dispozici. Je také cross-platform, můžeme ho tedy použít na různých operačních systémech. Také nabízí množství knížek k porozumění jazyka R ze série s názvem Use R! series. Kromě jeho výhod je třeba i zmínit nějaké mínusy kvůli efektivnímu používání R. Jazyk R má velmi komplexní syntax pro začátečníky, kteří se statistikou doposud nesetkali. Nainstalovaný balíček nemusí být vždy úplný a mohou se vyskytnout nějaké nedokonalosti ve stažených balíčkách. [10]

5.1 Instalace softwaru

Pro instalaci softwaru Rattle je třeba navštívit stránky od developerů programu https://rattle.togaware.com/. Jako první instalujeme jazyk R ve verzi 3.6.3 pro operační systém, který používáme. Po nainstalování spustíme R pomocí zástupce na ploše a na obrazovce se nám objeví workspace. Pokud chceme pracovat s grafickým prostředím, je třeba nainstalovat balíčky RGtk2 a GIMP Toolkit neboli GTK+, který slouží pro běh programů s grafickým rozhraním. Balíček RGtk2 podporuje GTK v jazyce R, proto je třeba tyto dva balíčky nainstalovat. Nyní je už třeba pracovat v konzoli pro nainstalování těchto dvou balíčků pomocí příkazů install.packages() a spuštění pomocí příkazu library(). Je důležité dodržovat velká a malá písmena, protože jazyk R je case sensitive.

```
> install.packages("RGtk2")
--- Please select a CRAN mirror for use in this session ---
```

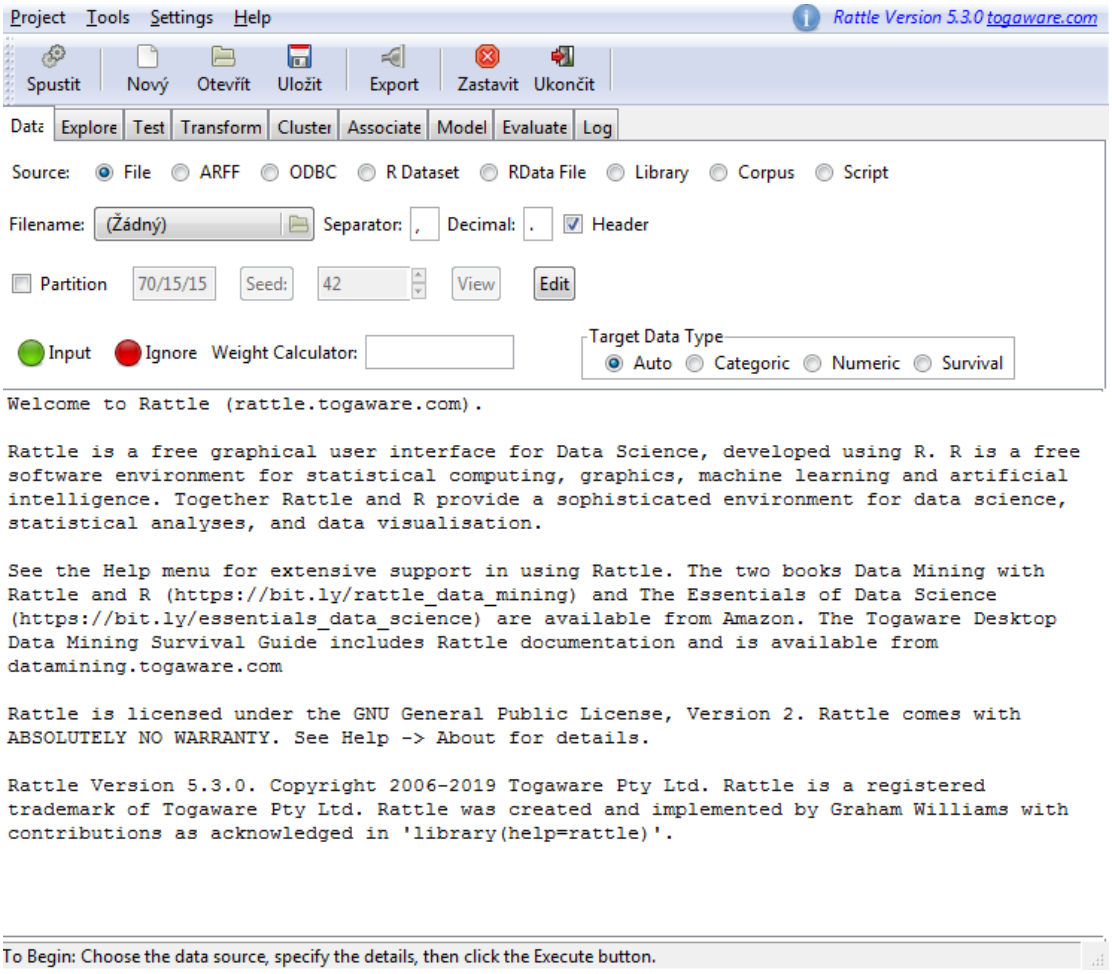
Po vyzvání vybereme příslušný server, ze kterého se stáhne balíček RGtk2. Po instalaci spustíme nainstalovaný balíček RGtk2 v této fázi se nám zobrazí chyba. Chybí nám totiž knihovna, kterou obsahuje jiný balíček. Program nás hned vyzve k doinstalování balíčku GTK+.

```
> library(RGtk2)
Error in inDL(x, as.logical(local), as.logical(now), ...) :
```

Nyní můžeme nainstalovat Rattle stejně jako balíček RGtk2 a spustit ho v konzoli pomocí library. Po výběru knihovny rattle, spustíme program Rattle pomocí příkazu rattle(). Nyní se nám otevře grafické prostředí, kde budeme pracovat.

```
> library(rattle)
Rattle: A free graphical interface for data science with R.
Version 5.3.0 Copyright (c) 2006-2018 Togaware Pty Ltd.
Type 'rattle()' to shake, rattle, and roll your data.
> rattle()
```

Na obrázku 4 vidíme grafické prostředí programu Rattle. Je možné, že se během instalace programu nenainstalují všechny potřebné balíčky. Opět se není čeho obávat Rattle nám hned doporučí, které balíčky je potřeba doinstalovat. Tímto jsme zakončili instalaci softwaru a jsme připraveni s ním pracovat.



Obrázek 4 Grafické okno programu Rattle

Zdroj: Vlastní Zpracování

V této nedotknuté formě má Rattle na výběr 3 datasety a to jsou: weather.csv, dvdtrans.csv a audit.csv. Na jednom z těchto datasetů je možné vyzkoušet základní prvky Rattle a orientovat se v novém prostředí. Tyto datasety můžeme otevřít kliknutím na seznam u položky Filename a zde vybrat příslušný dataset.

5.2 Popis softwaru

Jak je možné vidět na Obrázku 4 Rattle má několik záložek. Pro efektivní práci v tomto programu budou představeny záložky pro práci s daty.

Záložka **Data** jak název napovídá, nám pomáhá k prohlížení dat. Rattle dokáže nahrát data z mnoha různých zdrojů, jako jsou: CSV, TXT, ARFF ODCB (Toto připojení nám zajištuje další zdroje z SQL databází). Při nahrávání může mít každá proměnná jistý prefix, který ji pomůže lépe identifikovat jako například ‚ID_'. [10]

Záložka **Explore** nám dává na výběr ze spousty numerických a grafických možností jak prohledávat data. Nejvíce základní možností prohledávání dat je možnost textového shrnutí neboli Summary. K více přehlednějšímu shrnutí je třeba doinstalovat balíček Hmisc a použít příkaz describe() namísto summary(). Další možností je zde Distributions ta podává data na rozdíl Summary v grafické podobě. Je zde na výběr z řady možných grafů. [9][10]

Záložka **Test** nabízí možnost testů statistických distribucí. Obsahuje testy s parametry (T-test, F-test) a testy s non-parametry (Kolomogorov-Smirnov, Wilconxon Signed Rank). [10]

Záložka **Transform**, jak název napovídá, je k transformování dat. Jak bylo řečeno v předešlých kapitolách je třeba data zpracovat do podoby, která bude srozumitelná. Tato záložka nabízí několik různých metod transformace dat, jako jsou změna měřítka, vložení chybějících záznamů a metoda přemapování. [10]

Záložka **Model** poskytuje rozhraní, které nám pomáhá k vytvoření modelů, jako jsou decision tree, random forest a neural networks. Nabízí jak popisné modely, tak i ty prediktivní modely. Důležité je pro nás možnost vytváření prediktivních modelů kam se řadí klasifikace. [10]

Záložka **Evaluate** umožňuje zhodnocovat vytvořené modely pomocí několika metod, které jsou nám k dispozici. Pro zde vytvořený model vyhovuje metoda křivky ROC (Receiver Operating Characteristics), jelikož se používá při sdělení, zdali námi vytvoření model klasifikoval správně. [10]

Záložka **Log** funguje jako příkazový řádek v programu Rattle. Pokud dojdeme do bodu, kdy grafické rozhraní už nám nemůže pomoci. Poskytuje také komentáře ke každému kroku, který proběhl. [10]

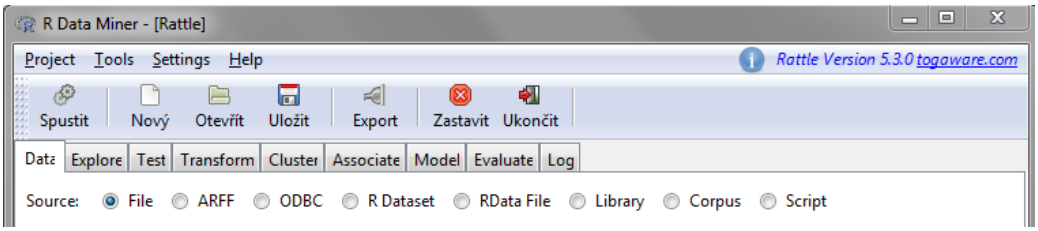
5.3 Výběr a zobrazení dat

V oblasti data mining používáme pro kolekci dat pojem dataset. Tento dataset se skládá z pozorovaných prvků a jeho proměnných. Prvky tvoří řádky a proměnné tvoří sloupce. Konečný počet těchto dvou čísel nám udává dimenzi datasetu. Proměnné jsou dvojího typu input a output. Proměnné typu Input jsou data, která byla dříve nějak změřena nebo přednastavena. V předchozích kapitolách jsme je nazývali prediktory. Output data na druhou stranu jsou data, která jsou ovlivněna input daty. V programu Rattle jsou zaznamenány v kategorii Target, jinak se také nazývají závislé proměnné. Některé proměnné, ale slouží jen k identifikaci. V našem případě se jedná o identifiers neboli ID. Taková proměnná je náhodně generována a je unikátní, aby bylo možné identifikovat správný prvek. Proměnné se také liší podle data typů, některé mají kategorickou hodnotu a některé mohou nabývat číselnou hodnotu. Kategorická proměnná je taková proměnná, která nabývá hodnotu z předem určeného množství možností. Může se jednat o barvu vlasů, věk nebo typickou odpověď Ano/Ne. [10]

Dataset může také mít různé role. V oblasti data mining často rozdělujeme dataset na tři části training dataset, validation dataset a testing dataset. Časté rozložení datasetu je 40/30/30 nebo 70/15/15. Rattle automaticky nastavuje rozložení 70/15/15. Model vytváříme pomocí prvního datasetu training dataset, pro vyhodnocení efektivnosti využíváme druhý dataset validation dataset a odhad budoucí efektivnosti je dělán pomocí posledního datasetu testing dataset. [10]

Když dojde na práci s datasety v reálném světě není vše dokonalé jako v teorii. Je možnost, že některá data budou zapsána špatným způsobem nebo některé části můžou být vynechány, proto je důležité pracovat s daty, která jsou kvalitní. V tomto kroku dojde na řadu preprocessing dat, kdy upravíme daty do podoby, se kterou je možné pracovat. Dalším problémem je zde spojování dat neboli data matching. Což je metoda, která spojuje několik datasetů dohromady, abychom měli úplné informace. [10]

Při nahrávání datasetů do programu Rattle máme na výběr z několika možností, které budou dále popsány. Patří sem CSV, ARFF, ODBC, R Dataset a R Data.



Obrázek 5 Výběr formátů v programu Rattle

Zdroj: Vlastní zpracování

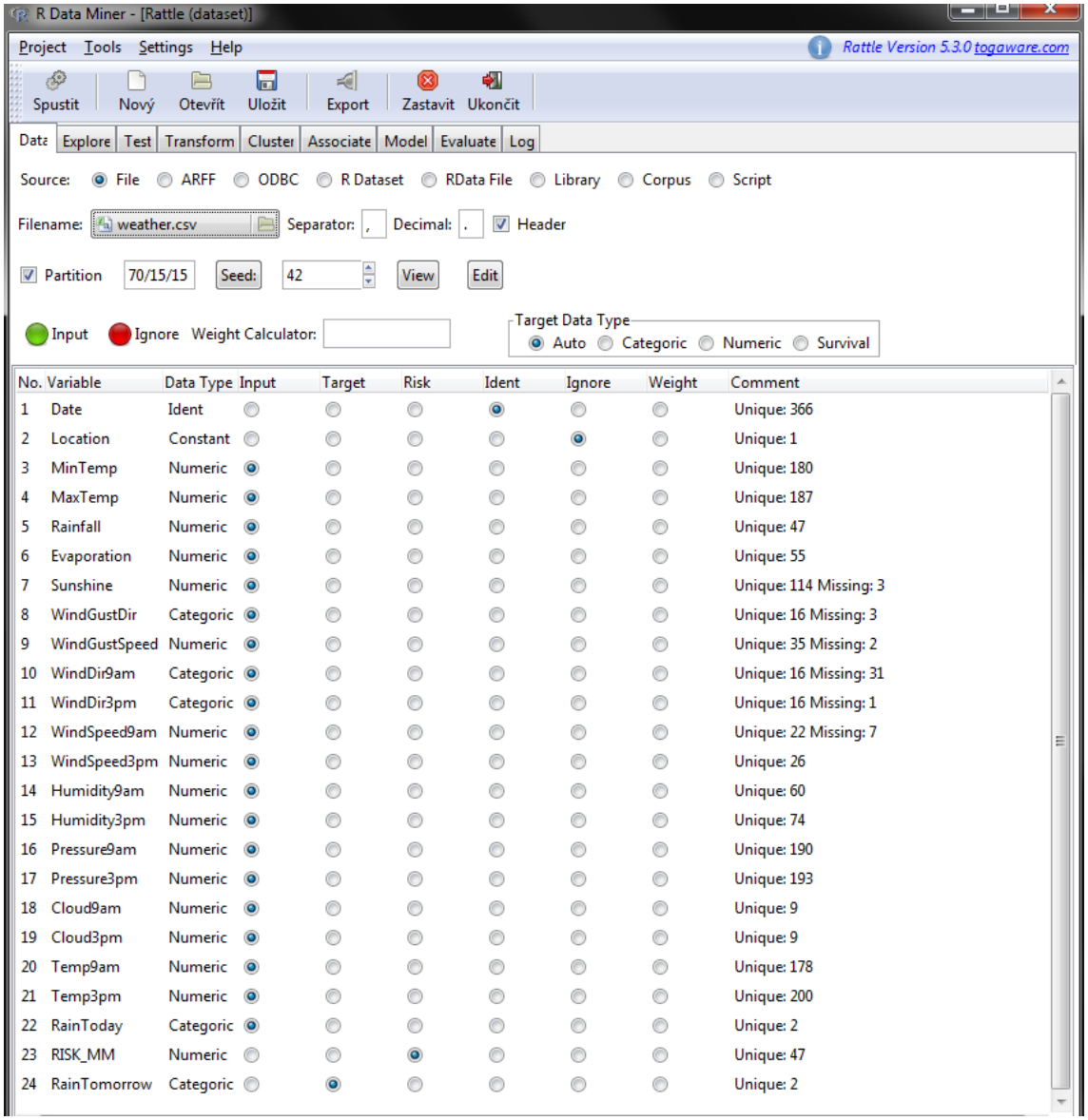
1. CSV – stojí pro comma-separated values. Tento formát je dnes považován za standard. Rozšíření CSV se nachází už v mnoha programech, které denně používáme, jako je Microsoft Excel, SAS/ Enterprise Miner, LibreOffice Calc. Nevýhodou tohoto formátu nulový obsah metadat. Proto Rattle může někdy špatně přiřadit data typ k dané proměnné. Zde najdeme formát CSV pod odrážkou File.
2. ARFF – stojí pro Attribute-Relation File Format. Tento formát je velmi podobný formátu CSV, ale jeho výhodou je, že obsahuje metadata v hlavičce souboru. ARFF formát byl vyvinut pro program Weka, který se také zabývá problematikou oblasti data mining.
3. ODBC – stojí pro Open Database Connectivity. Tato technologie byla vytvořena pro připojení programů k databázím nebo databázovým skladům. Funguje pomocí Structured Query Language neboli SQL dotazů na databáze. K připojení dochází pomocí vyplnění kolonky data source name (DSN) a přihlášení k vybrané databázi či skladu.
4. R Dataset – program Rattle podporuje spoustu možností formátu datasetu. Jsou zde však i jiné možnosti, jak nahrát dataset do programu. Výběr vybraného datasetu upřesníme pomocí vložením názvu do kolonky Data Name. Nahrát můžeme cokoliv, co může být dolováno pro znalosti. Může se jednat o XML soubor nebo data uložená ve schránce.
5. R Data – tento formát jak jméno napovídá je formát datasetu jazyka R. Potřebné datasety lze nahrát z nativní složky R.

[10]

V následujících části je obrázku 6 načten dataset weather.csv pro práci v programu Rattle. Tento soubor, jak je z koncovky vidět patří do formátu CSV. Proto vybereme z možností Source položku File, která zastupuje název CSV. Soubor je uložen v nativní složce R. Vykonaná práce v praktické části na tomto datasetu je podle vlastního uvážení z informací od autora Grahama Williamse.

Po nahrání dat se otevře okno s daty na záložce data. Zde můžeme vidět jak názvy, tak typy proměnných a komentáře k nim. Na obrázku 6 je vidět nahraný dataset a rozepsána data, které obsahuje. Zde vedle předchozího výběru souboru je vidět možnost Separator, Decimal a Header. Separator značí jak jsou položky oddě-leny v CSV souboru. Decimal označuje, jaká značka byla použita pro oddělení dese-tinných čísel a odrážka header označuje metadata. Dříve bylo zmíněno, že CSV formát neobsahuje metadata. To je pravda, zde se jedná o první řádek v tomto sou-boru, který nese názvy sloupců. Pokud by tato položka nebyla zaškrtnuta Rattle by doplnil názvy náhodně. [10]

Dále můžeme pozorovat, že Rattle už rozhodl za nás o rozdělení (Partition) datasetu na 3 různé datasety. Standardně použil rozdělení 70/15/15. Na práci s daty a vytvoření modelu bude použito 70% datasetu, 15% je využito na validaci datasetu a zbylých 15% je na testování správnosti a efektivnosti vytvořeného mo-delu. Seed číslo vybírá různé části datasetu, protože nastavuje generování náhod-ných parametrů, tak abychom zajistili vybrání stejné části datasetu ponecháme číslo, které vložil Rattle. [10]



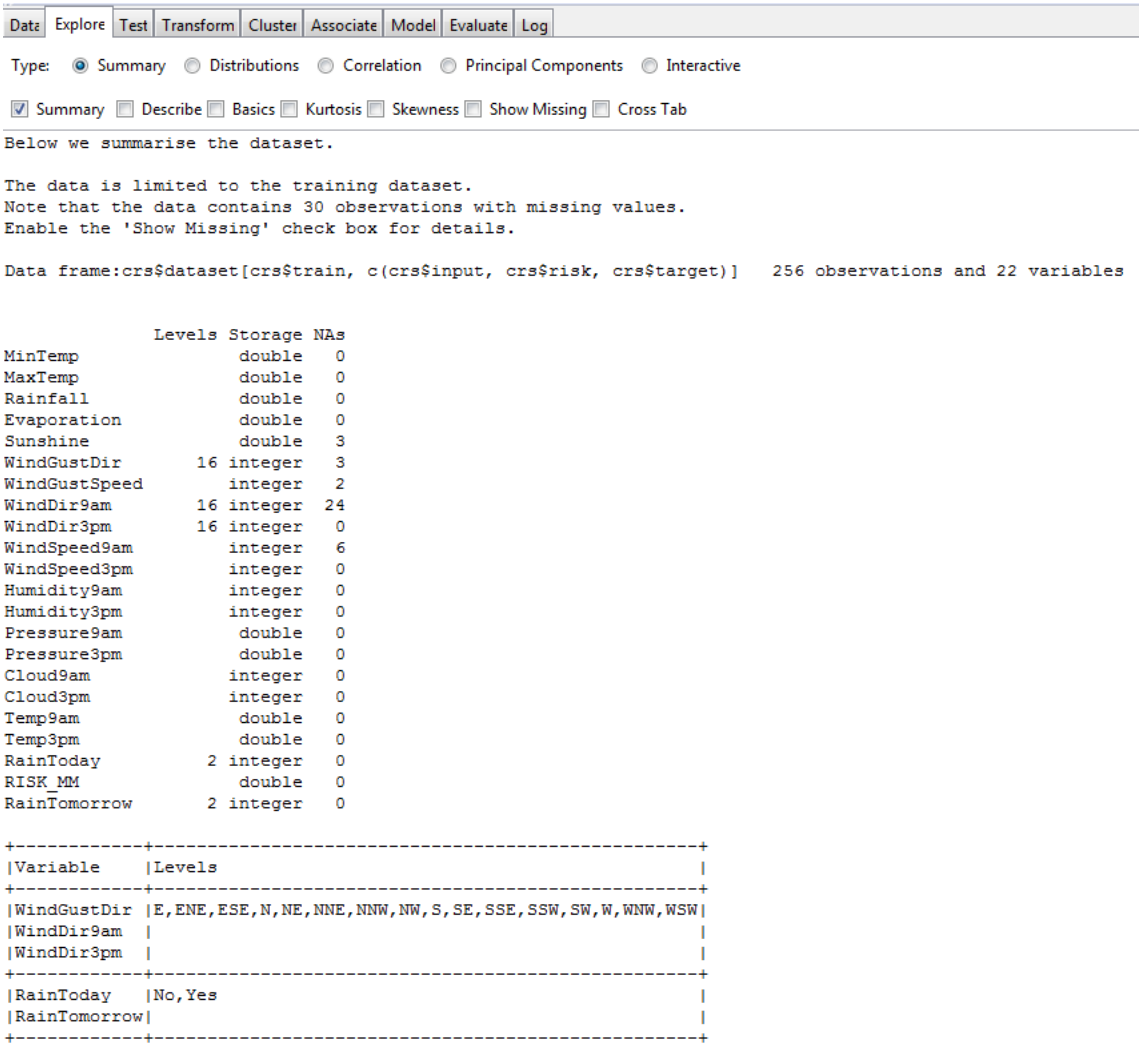
Obrázek 6 Nahraná data v Rattle

Zdroj: Vlastní zpracování

Nyní už jsme u proměnných a jejich výpisu. Zde vidíme názvy proměnných, data typ a roli jako mají v datasetu. Mnoho proměnných zde zastává roli typu input a ovlivňuje nám v tomto datasetu output neboli Target. V souboru weather.csv je Target proměnnou RainTommorrow. Rattle vybírá Target proměnné, tak že vyhledá proměnnou s malým počtem unikátních hodnot a to je maximálně 5 nebo proměnnou která je označena pomocí ‚TARGET_'. [10] Z obrázku 6 vidíme, že jsou tu ještě další proměnné, které mají jinou roli než Input. Další proměnná je Date, kterou Rattle zařadil do role Identifikátoru neboli ID. Rattle to udělal, protože tato

proměnná obsahuje unikátní hodnotu pro každé další měření v datasetu. [10] Další roli dostala i proměnná Location. Ta dostala roli Ignore, jelikož obsahuje jen jednu unikátní proměnnou, která je pro všechny stejná, tudíž nějak nezmění naše modelování a je ji tedy možno ignorovat. Poslední rolí je zde role Risk. Tato proměnná se nepoužívá k modelování, ale upozorňuje nás na závažnost spojenou s daným výsledkem. [10] Pomocí tlačítka View lze zobrazit tabulku s daty. Tlačítko Edit nám dovolí data upravovat, pokud je potřeba. Ovšem se tento přímý zásah do dat nedoporučuje. Z těchto údajů můžeme zjistit dimenze datasetu a ty jsou 366, 24. Tedy 366 řádků a 24 sloupců plných dat.

Můžeme přejít na záložku Explore a spustit Summary pro lepší přehled datasetu.



Obrázek 7 Funkce Summary

Zdroj: Vlastní zpracování

Ze summary vidíme souhrn datasetu weather.csv. Bylo shrnuto 256 řádků a 22 proměnných. Menší počet oproti známému množství 366 a 24 je kvůli tomu, že několik řádku nebylo plně vyplněno a 2 proměnné byly opomenuty, protože Identifikátor se v Summary nevypisuje a Location měla nastavenou roli proměnné na Ignore.

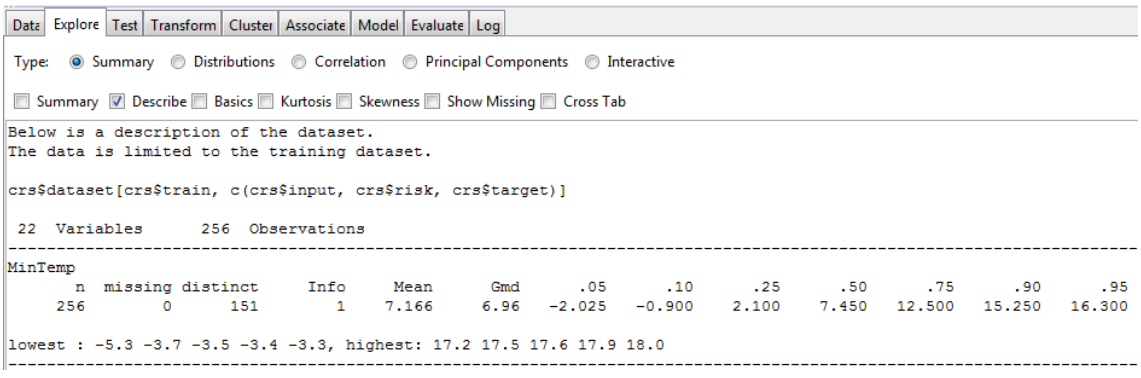
Proměnné jsou zde vypsány ve třech sloupcích Levels, Storage a NAs. Sloupec Levels je pro kategorické proměnné a udává kolik možností je na výběr. Sloupec Storage udává, jak se tato proměnná ukládá a do jakého typu. Sloupec NAs určuje, kolik chybí zápisů s touto proměnnou. Dále tu máme tabulku, kde jsou sepsány možnosti pro kategorické proměnné. Některé proměnné mohou mít na výběr stejné možnosti. Summary také ještě vypíše různé statistiky pro zápisy proměnných, jako jsou kvartály, minimum, maximum a průměr. Pokud se zde průměr (Mean) a medián (Median) velmi liší je zde možnost, že se v datasetu nacházejí odlehlé hodnoty. [10]

```
    MinTemp          MaxTemp          Rainfall       Evaporation
Min.   :-5.300   Min.   : 7.60   Min.   : 0.000   Min.   : 0.200
1st Qu.: 2.100   1st Qu.:14.80   1st Qu.: 0.000   1st Qu.: 2.200
Median : 7.450   Median :19.75   Median : 0.000   Median : 3.800
Mean   : 7.166   Mean   :20.42   Mean   : 1.536   Mean   : 4.418
3rd Qu.:12.500   3rd Qu.:25.55   3rd Qu.: 0.200   3rd Qu.: 6.200
Max.   :18.000   Max.   :35.80   Max.   :39.800   Max.   :12.400
```

Obrázek 8 Funkce Summary 2. část

Zdroj: Vlastní Zpracování

Pokud potřebujeme ještě přesnější souhrn je tu možnost Describe. Ta nám ke každé proměnné napíše souhrn zvlášť.

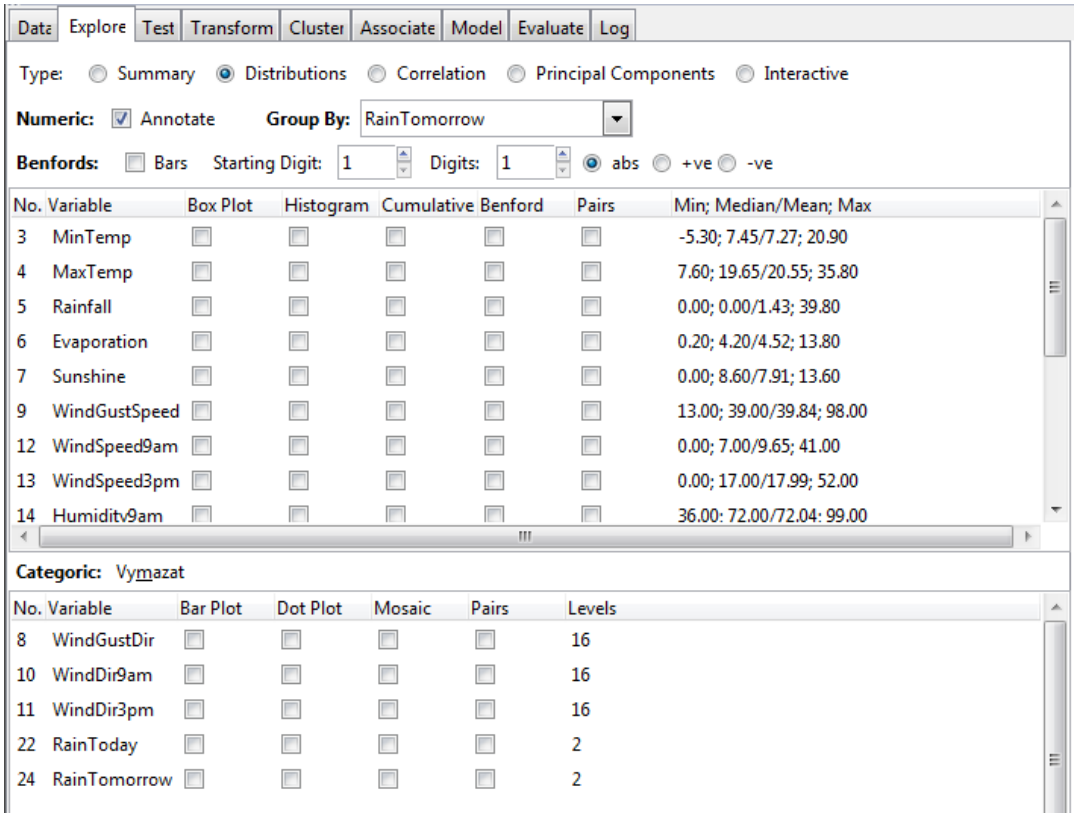


Obrázek 9 Funkce Describe

Zdroj: Vlastní zpracování

Na obrázku 9 vidíme přesný souhrn každé proměnné. Označení *n* jako počet zápisů, počet chybějících zápisů, unikátní hodnoty, průměr a percentily.

Dále je tu i grafické zobrazení dat na grafech pro rychlejší průzkum dat. Tato možnost se nachází také pod záložkou Explore a je názvem Distributions. Zde máme na výběr z několika možných grafů pro sledování vývoje a změn dat nebo jejich distribuci.



Obrázek 10 Distributions funkce

Zdroj: Vlastní zpracování

Jak bylo dříve napsáno, máme zde na výběr z mnoha typů grafů, jako je Box plot, Histogram, Kumulativní a Benfordův graf. Můžeme vybrat, které proměnné chceme graficky zobrazit vzhledem k proměnné, kterou vybereme v GroupBy. Nyní je vybrána kategorické proměnná RainTommorow a data budou zobrazena vzhledem k jejím hodnotám, což jsou Yes/No. [10]

Každý graf má své uplatnění. Pro nás budou nejlepší volbou box plot, histogram a kumulativní graf. Box plot nám ukazuje rozdělení dat v datasetu s různými

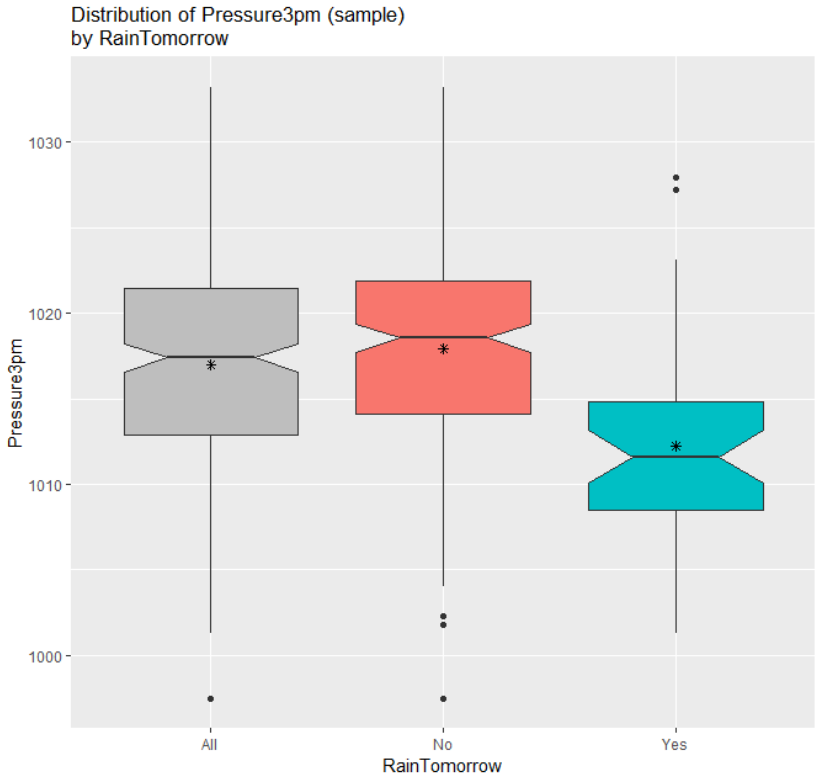
statistikami, které jsme obdrželi ze summary nebo describe a histogram nám ukáže rozsah dat v datasetu. Kumulativní graf ukazuje poměr dat, která mají hodnotu menší nebo rovná zobrazenou na ose x. [10]

Je možné vidět další možnost grafů a to jsou Benfordův a Pairs graf. Benfordův graf se hlavně používá při zjišťování nesrovnalostí v datech (například při odhalování podvodů v účetních datech). Funguje tak, že vypisuje v grafu frekvenci první číslice ve sbírce dat. Například výskyt teploty v rozmezí (10, 19), kde je první číslicí 1 má výskyt v 45 % případů. Tento graf se nejlépe využije, pokud je v datasetu velké měřítko (jako platy zaměstnanců). Graf Pairs používá dvě a více proměnné. Porovnává tyto proměnné a je možné, že najde korelaci mezi vybranými proměnnými pomocí scatter plot. Rattle podporuje maximálně 6 možných proměnných v tomto grafu. [10]

Pro kategorické proměnné jsou zde jiné grafy. Důležité pro nás jsou Bar plot a Dot plot. Bar plot používá vertikální sloupce pro zobrazení počtu výskytu dané kategorie. To stejné dělá Dot plot jen jiným způsobem. Dot plot je zobrazuje data v horizontálním směru a vrchol sloupce je označen tečkou. Dalším grafem je Mosaic Plot neboli mozaikový graf. Tento graf nám pomáhá k prezentování, jak jedna proměnná je rozdělena nad hodnotami druhé proměnné. [10]

V programu Rattle můžeme nastavit, jestli chceme mít grafy popsané zaškrtnutím možnosti Annotate. Dále zde máme měřítko pro grafy, kdy si můžeme nastavit Starting Digits (První číslo na ose x) a Digits (měřítko). Je zde i možnost grafického zobrazení jen kladných nebo záporných veličin zaškrtnutím možností abs, +ve nebo –ve. Možnost abs zobrazí absolutní hodnoty. Jak název napovídá možnost +ve zobrazí jen kladné hodnoty a možnost –ve zobrazí jen záporné hodnoty v grafu.

Během vytváření těchto grafů je možné že nalezneme na nějaké souvislosti nebo vztahy mezi nějakými proměnnými. Jako je v následujícím grafu.



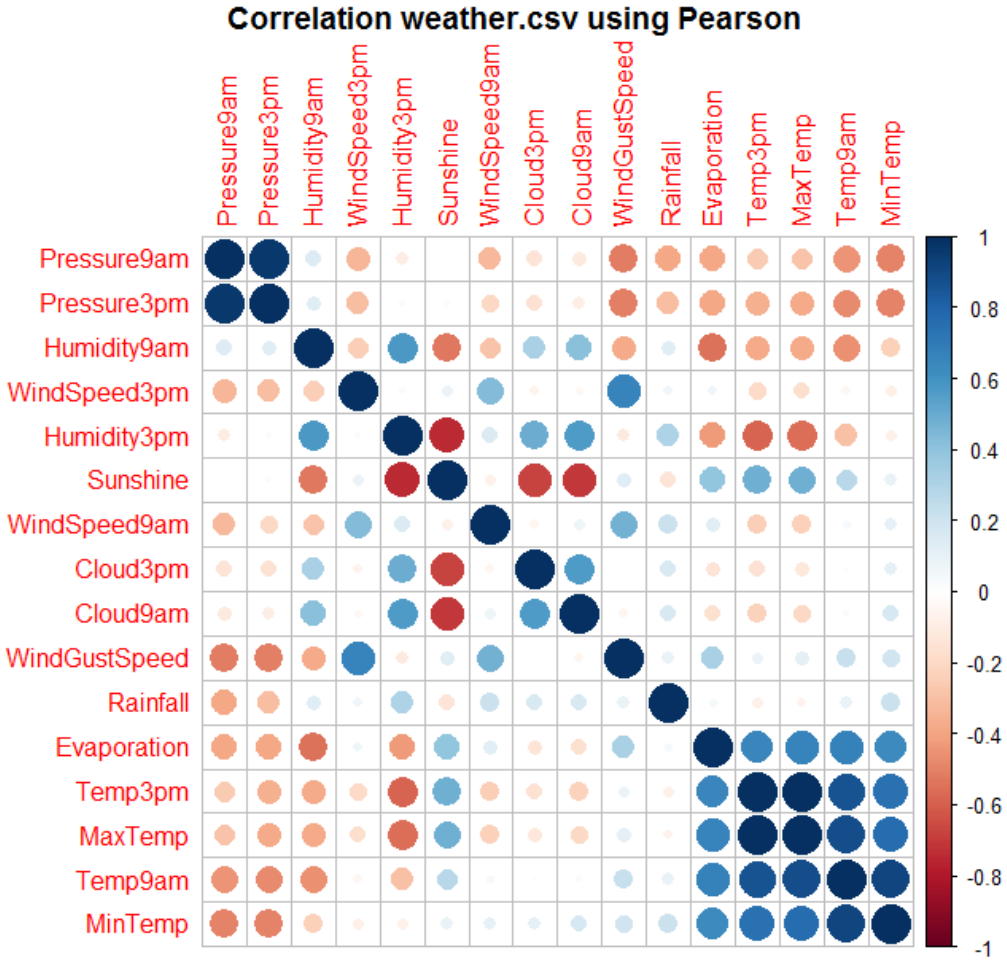
Graf 1 Boxplot Pressure3pm k RainTomorrow

Zdroj: Vlastní zpracování

Na grafu 1 lze vidět rozdělení veličin pro Pressure3pm k RainTomorrow. Rozdělení veličin pro All (všechny) a poté pro No a Yes. Vidíme zde také vyznačený medián pro každý box plot, který je vyznačen čárou se zobáčky. Zde vidíme i průměr, který je vyznačen pomocí *. Tyto zobáčky, které se nacházejí kolem mediánu, ukazují na 95% úroveň spolehlivosti na rozdíl mezi mediány a průměrem pokud by se jednalo o nezávislé sledování. [10] Dále vidíme i graficky zobrazené mezikvartilní rozpěti i horní a dolní hranici box plotu, která je vyznačena na grafu. Jsou zde vyznačeny i odlehlé hodnoty neboli outliers. Můžeme mezi proměnnými vypozorovat vztah, že menší tlak neboli Pressure3pm vyústí v déšť. Toto si lze ověřit ještě podle další možnosti, kterou Rattle nabízí a to je korelační analýza.

Pokud dojdeme k tomu, že používáme grafické zobrazení s více proměnnými. Můžeme mezi daty vypozorovat nějaké vztahy. V tomto případě můžeme využít korelační analýzu. Tato analýza nám odhalí různé stupně vztahů mezi těmito proměnnými.

Korelační analýza používá korelační koeficient, který nabývá hodnot od -1 do 1 a značí, jak moc jsou spolu dvě proměnné ve vztahu. Pokud se koeficient přibližuje k hodnotě -1, mají mezi sebou negativní vztah (nárůst jedné proměnné způsobí pokles druhé) a pokud se blíží k hodnotě 1, mají pozitivní vztah (nárůst jedné proměnné způsobí nárůst druhé proměnné). Pokud se objeví případ, kdy je korelační koeficient 1 nebo -1 značí to, že tyto proměnné jsou identické. [10]



Graf 2 Korelační graf

Zdroj: Vlastní zpracování

V tomto grafu lze vidět, že korelace mezi dvěma stejnými proměnnými nabývá hodnotu korelační koeficient hodnotu 1 na hlavní diagonále. Tedy se dá říci, že tento graf je symetrický podle diagonály. [10]

Můžeme zde vypozorovat i mnoho dalších vztahů. Je vidět z grafu, že nějaké proměnné mezi sebou vztah nemají například Temp9am a WindSpeed9am jejich korelační koeficient tedy nabývá hodnotu 0. Naopak můžeme pozorovat, že Pressure9am a Pressure3pm mají mezi sebou velmi velkou korelaci. Z grafu se může zdát, že jde o úplnou korelaci a tedy jejich korelační koeficient nabývá hodnotu 1 a jedná se o identické hodnoty. Ovšem při takové situaci můžeme nahlédnout do programu Rattle. Ten nám dá i textový výstup kde jsou vypsané hodnoty jednotlivých koeficientů v přehledné tabulce.

	Pressure9am	Pressure3pm
Pressure9am	1.000000000	0.96446164
Pressure3pm	0.964461635	1.00000000

Obrázek 11 Korelační koeficienty
Zdroj: Vlastní zpracování

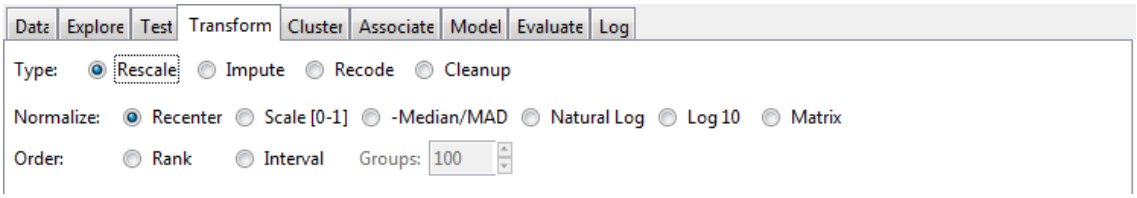
Zde můžeme vidět, že jich korelační koeficienty nabývají hodnoty velmi blízko k 1, ale nejsou 1. Toto nám naznačuje velmi silný vztah mezi těmito dvěma proměnnými, jak se ostatně v přírodě tento stav vyskytuje. Díky této analýze máme přehled o tom, jak zvýšení čí snížení jedné proměnné ovlivní ostatní proměnné.

5.4 Transformace dat

Jak bylo řečeno v teoretické části v kapitole data preprocessing. Před tím než začneme data modelovat je třeba je zpracovat a transformovat. Tento krok existuje, protože obdržená data nejsou vždy v perfektním stavu, jaký bychom chtěli. Někde mohou chybět zápisy nebo může obsahovat odlehlé hodnoty. Je třeba se o tyto problémy postarat než začneme s vytvářením modelů. Tyto nesrovnalosti by mohly mít nepříznivé účinky na modely. Musíme proto data upravit do jisté kvality, než začneme modelovat a dolovat z nich znalosti. [1]

Je třeba také datům porozumět, než s nimi začneme pracovat. Je třeba vědět, proč a jak tyto data byly nasbírány. Velkou část času zabere samotné čistění dat, kdy nasbírána data musíme upravit do určité kvality. V datech se můžou vyskytnout jakékoliv chyby. Mohl v tu chvíli selhat systém nebo se vyskytl error a záznam nebyl uložen nebo nebyl záznam ani zapsán, jelikož nebylo třeba. [10]

V tomto datasetu se může jednat třeba o WindGustDirection. Pokud v tu dobu nefoukal žádný vítr, tak nebyl zaznamenán ani jeho směr. To samé pro WindGustSpeed. Kdy v určenou dobu nefoukalo, tak nebyl záznam uložen. Pro tyto případy máme v programu Rattle záložku Transform, kde máme několik typu funkcí, které nám s touto problematikou pomohou.



Obrázek 12 Záložka Transform

Zdroj: Vlastní zpracování

První na záložce Transform jsou funkce typu Rescale. Různá data mohou mít různá měřítka, proto existují funkce tohoto typu, které nám pomohou změnit měřítka dat. Pomocí několika funkcí.

- **Recenter** – Používá takzvané Z-score. Změní naše data, tak aby průměr byl 0 a standardní odchylka byla 1.
- **Scale [0-1]** – Změní měřítka dat, aby spadaly do intervalu (0,1). Stejně jako Recenter, ale používá se pro proměnné, které nabývají kladných hodnot na celém intervalu.
- **Median/MAD** – Median Absolute Deviation tato funkce se týká hlavně odlehlých hodnot. Používá stejný výpočet jako funkce Recenter.
- **Natural log, log 10** – Obě funkce dělají to samé. Vezmou velké rozpětí kladných hodnot a sníží je na menší spektrum pro lepší přehlednost. Rozdíl je v tom, že funkce natural log využívá logaritmus jako svou bázi číslo *e* neboli Eulerovo číslo. Log 10 naopak používá v bázi číslo 10.
- **Rank a Interval** – Pokud nechceme určité číslo, ale zajímá nás pouze pozice čísla v distribuci, můžeme transformovat data pomocí funkce Rank. Funkce Interval funguje na stejném principu, ale můžeme zvolit velikost intervalu na, který budou data transformována doplněním čísla do kolonky Groups.

[10]

Dříve jsme poukázali na fakt, že nám chybějí záznamy v datasetu s tím nám pomohou funkce typu Impute. Tyto funkce byly popsány v teoretické části v podkapitole čištění dat.

- **Zero/Missing** - pokud záznam chybí, můžeme doplnit do tohoto záznamu 0, pokud jsme si jisti, že tam logicky patří (pokud zaznamenána rychlost větru v WindGustSpeed můžeme zde doplnit 0, jelikož logicky lze odvodit, že v tu chvíli nefoukal žádný vítr). Pro kategorické proměnné se doporučuje doplnit Missing.

- **Mean** – Doplní do chybějících hodnot průměr. Tato metoda je doporučena, pokud je koeficient šikmosti nulový.
- **Median –** Doplní do chybějících hodnot medián. Tato metoda je doporučena, pokud hodnoty nemají nulový koeficient šikmosti.
- **Mode** – Doplní do chybějící hodnoty modus. Tato metoda je doporučena pro kategorické proměnné.
- **Constant** – Tato funkce nám dovolí doplnit námi zvolenou konstantu nebo znak pro kategorické proměnné.

[10]

Další typem funkcí jsou funkce Recode. Tyto funkce nám dovolí změnit datový typ proměnných podle potřeby. Zde se používá již dříve zmíněna funkce Binning. Lze numerické proměnné rozdělit do několika kategorií pro lepší přehlednost, ale tímto ztratíme přesnější data ze záznamu. Je tu i možnost rozdělit kategorické proměnné na numerické proměnné, které uchovávají hodnotu kolikrát je tato kategorická hodnota zaznamenána v datasetu. [10]

Pro smazání záznamů nebo proměnných slouží funkce typu Cleanup. Zde je na výběr jaké možnost máme:

- **Delete Ignored** – Smazání ignorovaných proměnných.
- **Delete Selected** – Smazání vybraných proměnných
- **Delete Missing** – Smazaní proměnných, které mají chybějící záznamy
- **Delete Observations with any Missing Value** – Smazání řádků, které mají chybějící záznam v jedné z proměnných

[10]

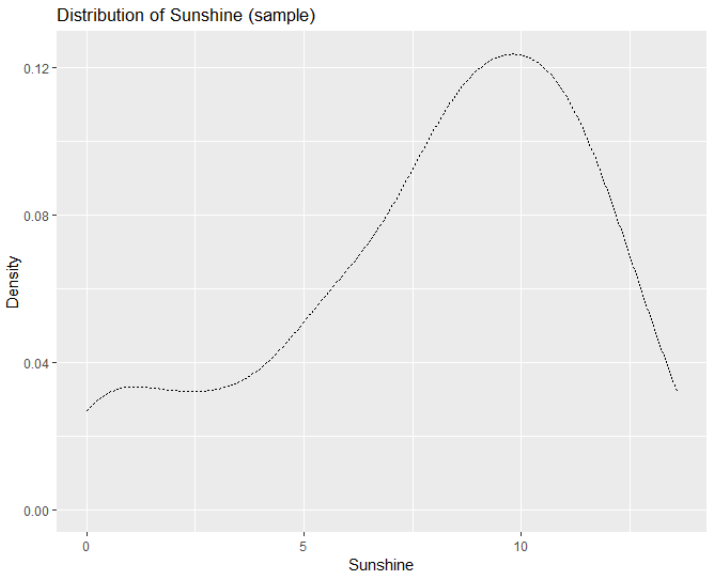
Nyní aplikujeme tyto funkce na náš dataset weather.csv. Nejdříve se zaměříme na chybějící hodnoty. První na řadě je proměnná *Sunshine*. Tato proměnná má tři chybějící záznamy. Ze struktury záznamů vidíme, že záznamy byly prováděny v 9 hodin ráno a 3 odpoledne. Touto dobou svití slunce ve slunné lokaci jako je Canberra. Z dat lze vidět, že i když slunce nesvítilo, byla zaznamenána hodnota 0 (záznam 263). Tudíž se může jednat o systémovou chybu.

	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine
263	2008-07-20	Canberra	-1.1	11.0	0.2	1.8	0.0
301	2008-08-27	Canberra	-3.3	15.1	0.0	3.0	NA

Obrázek 13 Záznam Sunshine

Zdroj: Vlastní zpracování

Z prohledávání dat také můžeme zjistit, že hodnoty Sunshine jsou kladné. Zde bychom mohli použít funkci typu Impute a to Median neboli doplnění mediánu. Tato metoda je doporučena pokud záznamy v datasetu mají neasymetrické rozdělení, neboli mají koeficient šikmosti nenulový. [10] To si můžeme potvrdit pomocí Explore záložky a grafického zobrazení v Distributions.



Graf 3 Rozložení proměnné Sunshine

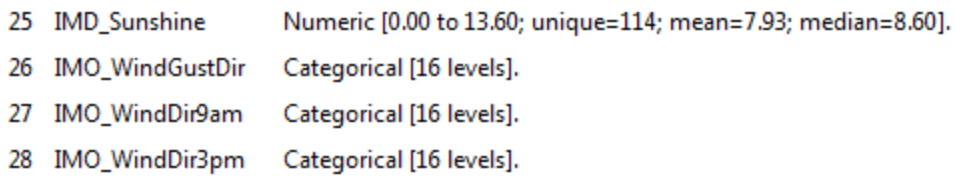
Zdroj: Vlastní zpracování

Nyní stačí v programu Rattle označit proměnnou Sunshine, zvolit typ funkce Impute, zaškrtnout funkci Median a kliknout na tlačítko Spustit. Tímto nám vznikne nová proměnná IMD_Sunshine zatímco Sunshine zůstane zachována a její role je automaticky změněna na Ignored. Už nebude nadále v naší práci figurovat.

Tento postup opakujeme pro další proměnné a podle potřeby vybereme funkci, která je nejvíce vhodná pro transformování dat z dostupných informací, jako je analýza pomocí grafů nebo funkce summary.

Další je proměnná WindGustDirection pro ni vybereme funkci Mode. Funkce Mode je obzvláště výhodná pro kategorické hodnoty, kdy vybere tu nejčastěji používanou kategorii a doplníme ji do záznamů. [10] Podle grafu 4 se jedná o NW. Tento postup aplikujeme i pro WindDir9am, WinDir3pm. Toto můžeme provést díky analýze dat, kde zjistíme pomocí grafu 5, který nám ukáže rozložení kategorií pro tyto kategorické proměnné. Nyní doplníme záznamy pro proměnné WindDir9am a WinDir3pm pomocí funkce Mode typu Impute.

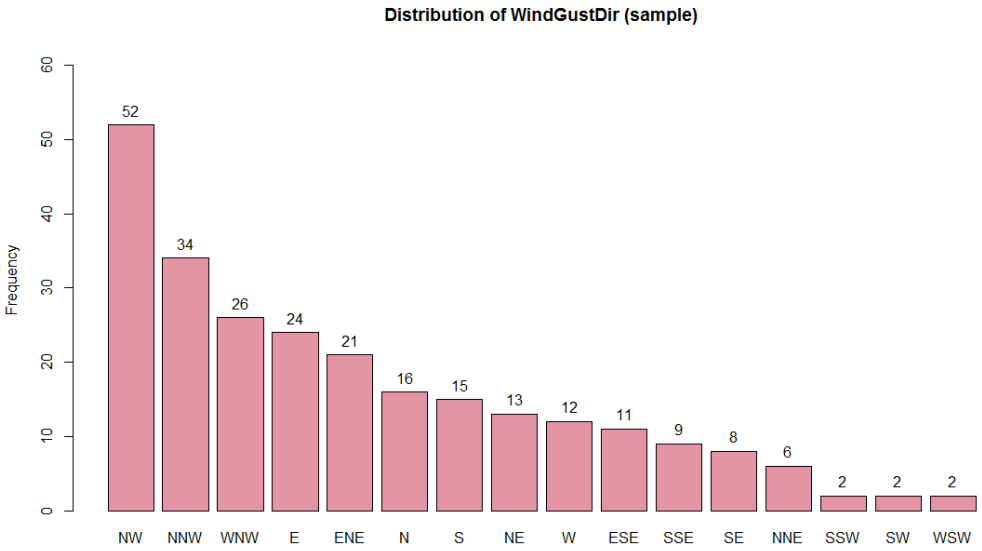
Nyní na proměnnou WindGustSpeed, kde se nacházejí dva chybějící záznamy. Podle analýzy dat můžeme zpozorovat, že pokud vítr nefoukal, byla doplněna hodnota 0, zde však bylo zaznamenáno NA, což může opět značit nějakou chybu v uložení nebo zápisu záznamu. Toto by mohlo později způsobit nejasnosti v modelu. Proto vymažeme řádky, kde chybí záznamy. Máme doplněné WindGustDirection, ale může nastat situace, kde nebyla vyplněna ani jedna proměnná, proto je lepší tyto záznamy smazat. Ovšem je vymažeme, až doplníme záznamy pro ostatní proměnné.



Obrázek 14 Nově transformované proměnné

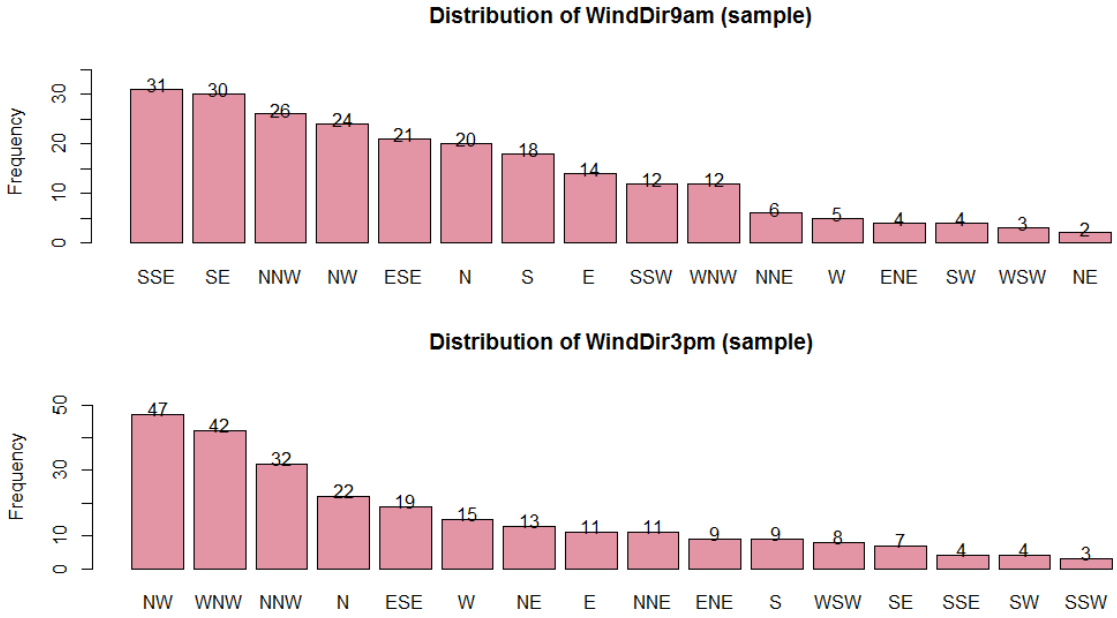
Zdroj: Vlastní zpracování

Zde na grafech můžeme vidět rozdělení kategorických proměnných a pochopit proč byla vybrána funkce Mode typu Impute.



Graf 4 Rozložení WindGustDir

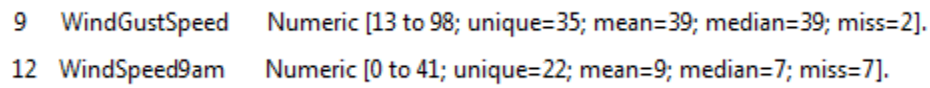
Zdroj: Vlastní zpracování



Graf 5 Rozložení WindDir9am a WindDir3pm

Zdroj: Vlastní zpracování

Nyní se zaměříme na chybějící hodnoty v proměnných WindSpeed9am a WindGustSpeed. Z výpisů dat ze záložky Transform, můžeme vidět výpis dat.

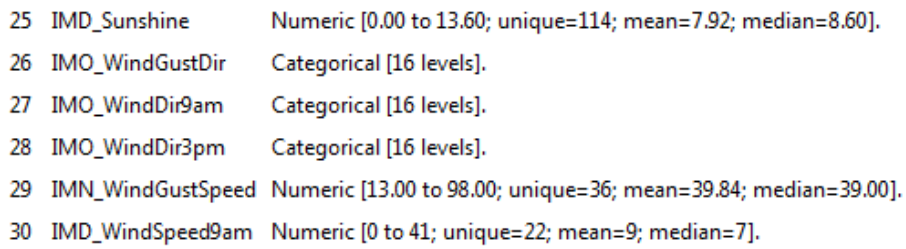


Obrázek 15 Výpis ze záložky Transform

Zdroj: Vlastní zpracování

Zde vidíme, že WindGustSpeed má symetrické rozdělení, jelikož je medián a průměr stejný. Použijeme tedy funkce typu Impute a doplníme hodnotu. Můžeme vybrat průměr nebo medián v tomto případě na výběru nezáleží. Ovšem v případě WindSpeed9am je to jiný případ. Koeficient šikmosti je kladný a jedná se o neasymetrické rozdělení. V těchto případech je funkce Median lepší volbou. [10]

Po tomto zpracování dat bude zobrazeno na konci několik nových proměnných. Každá nová proměnná má prefix, který označuje jaká funkce, byla použita na tuto proměnnou. Například IMD_ pro funkci typu Impute Median, IMN_ funkce typu Impute Mean, IMO_ značí funkci Impute Mode. Tato označení mají i ostatní funkce pro transformování dat, aby bylo lépe rozeznat, jak byla která proměnná byla zpracována.



Obrázek 16 Transformované proměnné

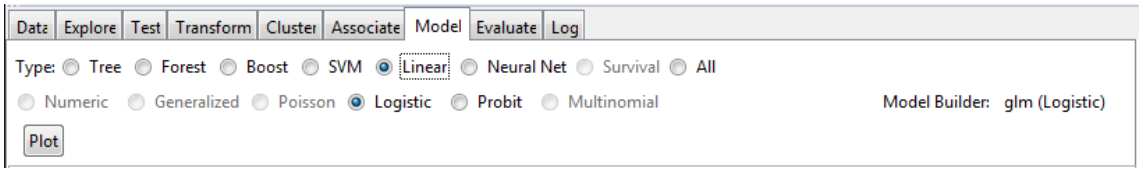
Zdroj: Vlastní zpracování

Nyní můžeme vymazat nepotřebné záznamy, které zůstaly bez záznamu proměnných i po úpravě. Pomocí funkce Delete Observations With Missing Value typu Cleanup. Pokud jsme vše udělali správně, mělo by být smazáno 9 záznamů (můžeme smazat i Ignored proměnné ty se ovšem i tak nemodelují) a tím bychom měli změnit dimenze datasetu na 357, 24. Pokud tedy nepočítáme Ignored proměnné jinak jsou dimenze 357, 28.

5.5 Použití úloh data mining

V této podkapitole bude vysvětleno jak namodelovat jednu z úloh oblasti data mining pomocí programu Rattle a bude také vysvětleno, co nám její výsledek podává za znalosti.

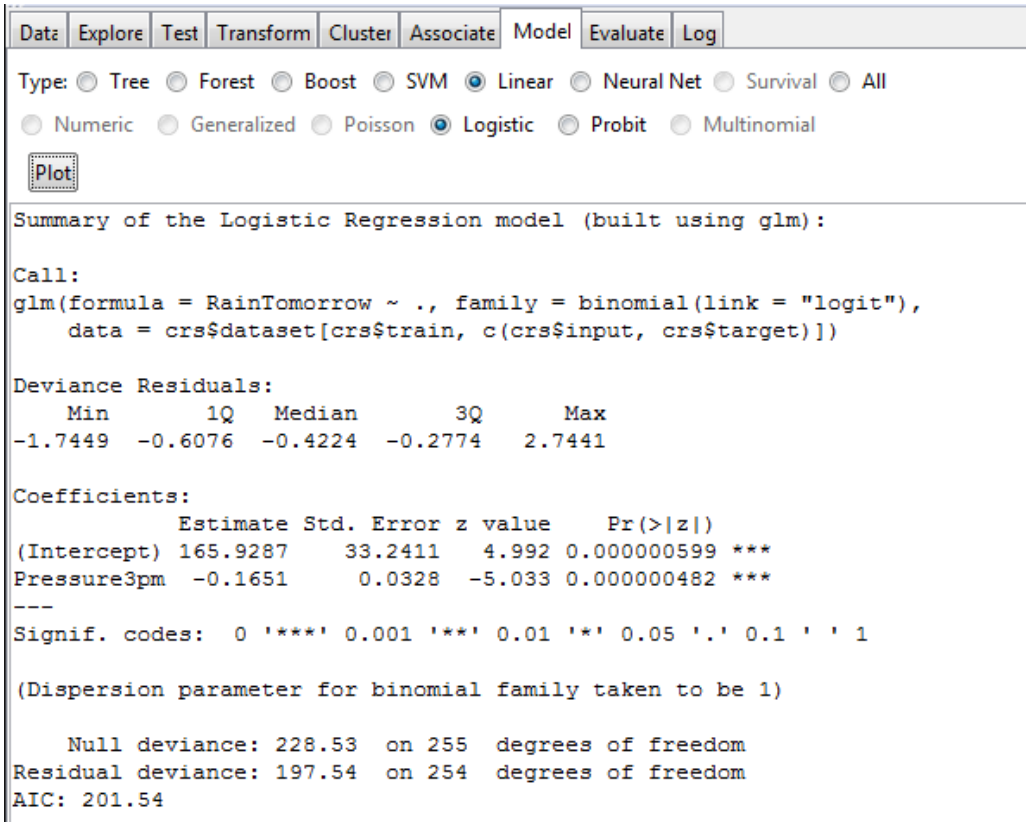
Nyní přejdeme na záložku Model. Na této záložce budeme vytvářet model pro dolování znalostí z upraveného datasetu weather.csv. Vybereme typ funkce Linear a vybereme funkci Logistic. Tudíž budeme modelovat Logistickou regresi. Jak bylo řečeno v předešlých kapitolách logistická regrese, vrací výsledek ve formě hodnot 0 nebo 1. [8] Rattle vypočítá koeficienty pro prediktory podle toho, jak jsou důležité pro zařazování. Poté stačí dopočítat podle vzorce a vrátí se nám výsledek ve formě 0 nebo 1. [8] Také tento výsledek můžeme nazvat jako kategorický neboli Ano nebo Ne. Toto je pro nás velmi prospěšné, jelikož naší Target proměnnou je kategorická proměnná RainTomorrow. Nejprve ukážeme logistickou regresi s jednou proměnnou a poté s více proměnnými.



Obrázek 17 Výběr modelu

Zdroj: Vlastní zpracování

Pokud chceme pracovat jen s určitými proměnnými je třeba v záložce Data, zatrhnout Ignore u proměnných, které si nepřejeme mít v modelu. Proměnná Presurre3pm bude jediná Input numerická proměnná, která bude vystupovat v našem modelu. Teď jen zmáčkneme tlačítko Spustit a Rattle vše vypočítá za nás. Je možné, že budeme opět upozorněni na chybu, protože není stažen balíček glm, který se k těmto výpočtům používá. Opět se není čeho obávat. Rattle za nás provede instalací daného balíčku.



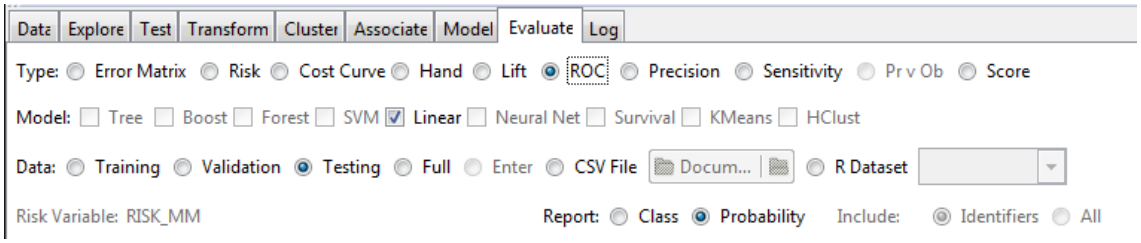
Obrázek 18 Logistická regrese s p = 1

Zdroj: Vlastní zpracování

Zde jako první můžeme vidět příkaz, který bychom využili konzoli v R. Kdybychom pracovali jen s příkazovou řádkou.

To co nás zajímá je Coefficients. Pro zjištění kvality regrese používáme std. error neboli Standard Error. Tento error nám ukazuje, jak moc budeme odchýleni od skutečných koeficientů β_0, β_1. [8] Hlavní dva koeficienty, které nás zajímají, jsou estimate a p-value (v Programu Rattle označena jako Pr(>|z|)). Záporný estimate nám značí, že pokud byl nárůst Pressure3pm je menší šance, že zítra dojde k dešti, neboli stoupne pravděpodobnost pro RainTomorrow = No (logaritmus šance je zmenšen o 0.1651). [8] Ovšem to nám nenaznačuje, že existuje nějaký vztah mezi těmito dvěma proměnnými. Proto zde existuje p-value. Pokud je p-value velmi malá ukazuje nám tím, že je zde nějaká asociace mezi proměnnými. [8] Rattle používá systém označení pomocí hvězdiček. Ty přiřazuje k jednotlivým koeficientům p-value podle toho, jak je významný. Na obrázku vidíme, že Pressure3pm má tři hvězdičky a je tady velmi významnou v předpovědi počasí na příští den.

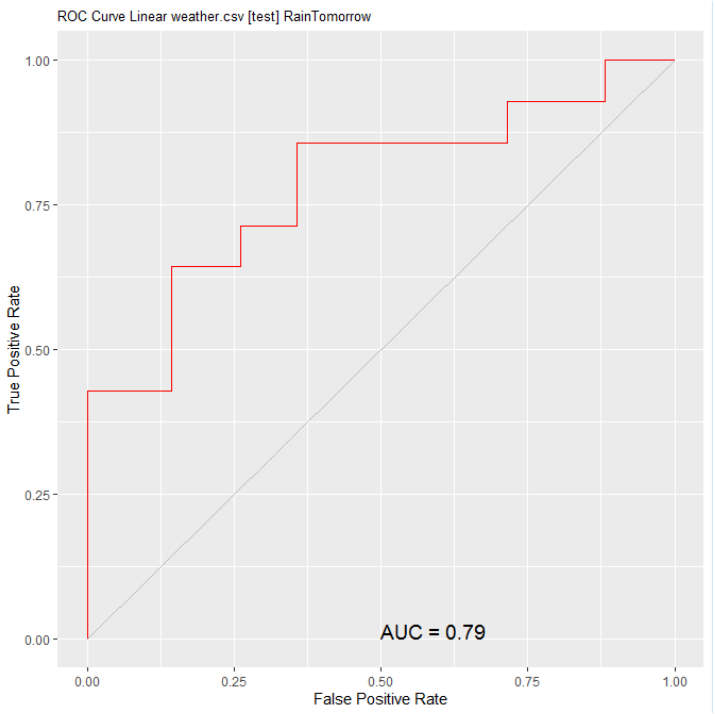
Pokud potřebujeme zhodnotit námi vytvořený model, přejdeme na záložku Evaluate. Vybereme, jaký typ zhodnocení chceme a zjistíme tím kvalitu modelu.



Obrázek 19 Záložka Evaluate

Zdroj: Vlastní zpracování

Zde vybereme typ hodnocení pomocí ROC křivky (Reciever Operating Characteristics). Tento grafický nástroj je velmi populární pro zachycení efektivnosti klasifikátoru. Toto měření se provádí pomocí výpočtu Area Under the Curve neboli AUC. Dobrý klasifikátor je jakýkoliv klasifikátor, který má AUC nad 0.5. [8]

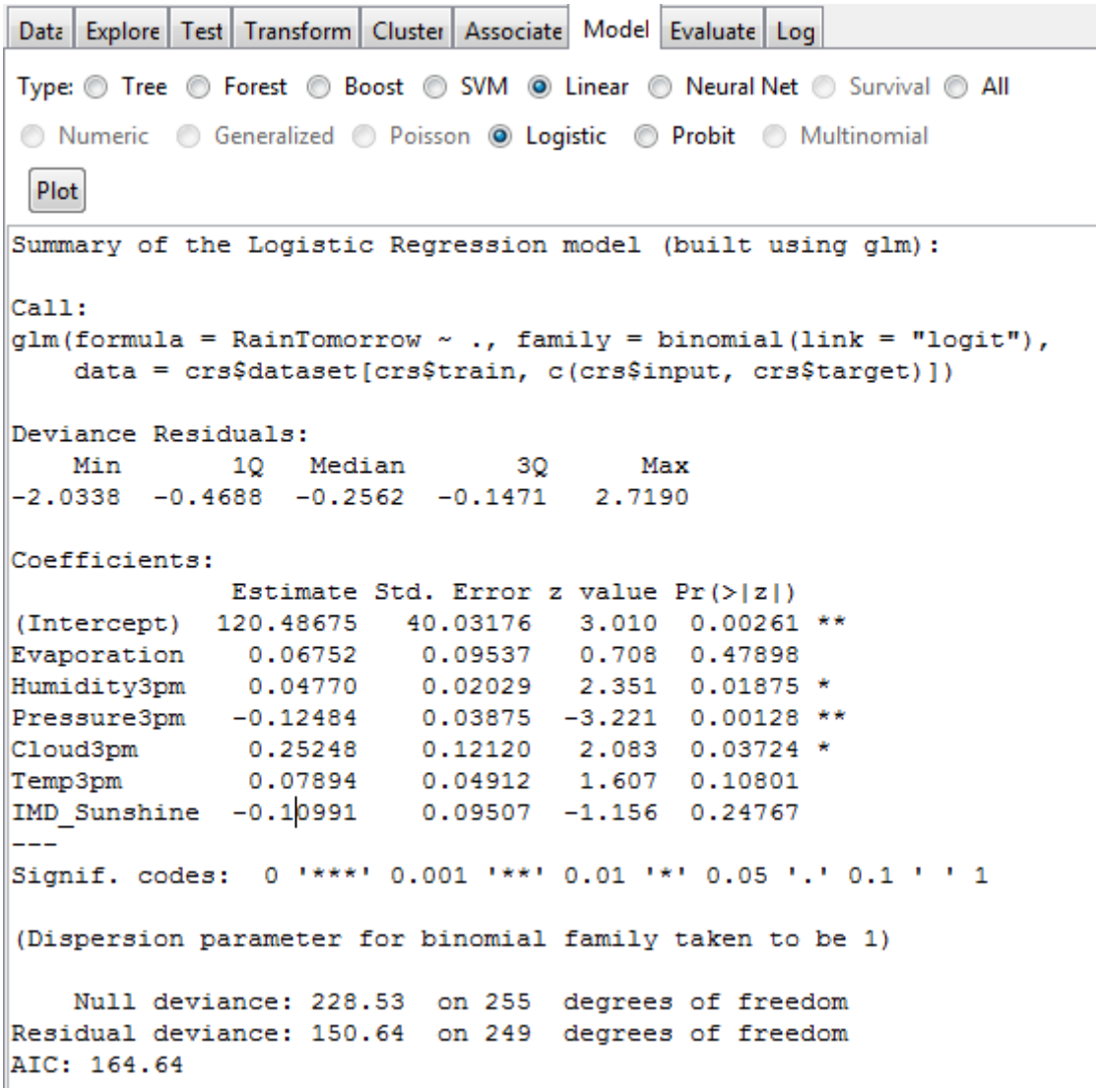


Graf 6 ROC křivka

Zdroj: Vlastní zpracování

Na tomto grafu můžeme vidět, že AUC je 0.79 na testing sample, který nebyl použit při vytváření modelu. Tudíž je to dobrý klasifikátor. Pozorovatelnou malé zakřivení této křivky může přisoudit 15% pozorování z Testing sample.

Nyní ukážeme logistickou regresi pro více proměnných. Nejdříve ovšem musíme přejít na záložku Data a zaškrtnou, které proměnné chceme jako Input proměnné.

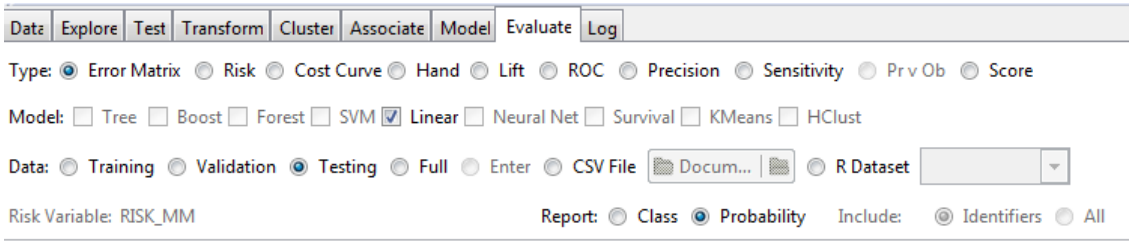


Obrázek 20 Logistická regrese pro p > 1

Zdroj: Vlastní zpracování

Z výpisu vidíme, že opět Pressure3pm je velmi významná. Další významné proměnné jsou Humidity3pm a Cloud3pm, které na rozdíl od Pressure3pm má kladný Estimate. Koeficient s kladným estimate značí, že zvýšením této proměnné se zvětší šance na RainTomorrow=Yes (logaritmus šance je zvýšen o 0.04770 při zvýšení humidity3pm a zvýšen o 0.25248 pří zvýšení Cloud3pm). [8] Například p-value pro Evaporation je 0.47898 a IMD_Sunshine je 0.24767 jsou velmi vysoké a program Rattle ji neoznačuje jako významné proměnné pro předpověď.

Při zhodnocování tohoto modelu použijeme jinou metodu a to je metoda klasifikační matice. Na záložce Evaluation zvolíme typ Error Matrix (klasifikáční matice) a zaškrtneme Testing.




```
Error matrix for the Linear model on weather.csv [test] (counts):

      Predicted
Actual No Yes Error
   No  40   2   4.8
   Yes  7   7  50.0

Error matrix for the Linear model on weather.csv [test] (proportions):

      Predicted
Actual   No  Yes Error
   No  71.4  3.6   4.8
   Yes 12.5 12.5  50.0

Overall error: 16.1%, Averaged class error: 27.4%
```

Obrázek 21 Klasifikační matice

Zdroj: Vlastní zpracování

Testing sample jsme si určili na úplném začátku pomocí partition na 15%. Těchto 15 % je z původních 366, proto máme v testing sample jen 54.8 pozorování.

Na obrázku 22 vidíme výstup z Programu Rattle ve formě klasifikační matice na Test Sample podle modelu, který byl vytvořen. V matici na diagonále můžeme vidět správně zařazené prvky. [8] Rattle podle vytvořeného modelu předpověděl 50 pozorování a 4.8 pozorování předpověděl špatně. Celková chybnost pro tento model je 16.1%.

6 Shrnutí výsledků a závěr

V teoretické části byl popsán postup metody KDD k znalostem. Data mining zde byl obsažen jako jeden krok. Bylo však třeba ukázat teorii, která nám přispívá k efektivnosti data mining ještě než k tomuto kroku dojdeme. Prvním krokem je zde výběr vhodných dat pro dolování informací. Poté přišly na řadu datové sklady pro archivaci dat. Po uložení dat je třeba data předzpracovat pro další krok (k tomuto kroku bylo vysvětleno mnoho metod, jako jsou čistění a kategorizace dat). Nyní už přišel na řadu samostatný krok data mining. Zde byly ukázány a vysvětleny úlohy typu klasifikace a diskriminace.

V praktické části byla předvedena práce s daty a vytváření modelu v programu Rattle. Jak bylo řečeno Rattle je postaven na programovacím jazyku R. Práce na datasetu weather.csv byla provedena podle poznatků z knih od Grahama Williamse. Všechno, co bylo provedeno v grafickém prostředí, bylo možné zapsat v konzoli v jazyce R. Tento kód je k nahlédnutí na záložce Log. Je ovšem poněkud složitý na syntaxi. Grafické prostředí Rattle nám velmi ulehčilo práci a také jsme měli rozsáhlý přehled o datech. Důležité byly dvě záložky Data a Explore pro analýzu dat. Mohli jsme rychle zjistit role a typy proměnných a rychle vytvořit grafy, což nám zefektivnilo práci na datasetu. Byla také demonstrována základní transformace dat a data preprocessing. Důležitou roli zde hrála práce s chybějícími hodnotami a jejich doplněním.

Grafické zobrazení dat velmi pomohlo při prozkoumání datasetu a rozhodnutí o použití správné funkce nebo i dokonce typu funkce pro zajištění správnosti později vytvořeného modelu. Ovšem ne všechny funkce nebyly použity. To z několika důvodů. Nebyly použity funkce typu Rescale, které změní naše proměnné, když jsou mezi měřítky proměnných veliké rozdíly. Na druhou stranu jsou zde typy funkcí Recode. Byla možnost použít funkce tohoto typu, ale došlo by ke ztrátě přesnějších dat, které jsme dále používali a zmenšili bychom tím i dataset a to nebylo potřeba pro tento dataset.

Dále byl vytvořen daný model logistické regrese. Kde byla popsána logistická regrese s jednou a s více proměnnými. Rattle vypočítal koeficienty a ty které byli důležité pro klasifikaci označil hvězdičkami. Také bylo možné vidět, jestli tato

proměnná pozitivně nebo negativně ovlivňuje šanci na výsledek pomocí výpočtu koeficientů. Poté jsme tento model ohodnocovali na záložce Evaluate. Zde byly předvedeny dva typy zhodnocení modelu pomocí ROC křivky a klasifikační matice. Oba tyto typy zhodnocení testovali model na testing sample, který činil 15% z původního datasetu, jež jsme rozdělili na úplném začátku při zapnutí programu Rattle. Model podle očekávání fungoval správně a dokázal předpovídat na základě proměnných, jestli bude pršet nebo ne. Model s jednou proměnnou měl senzitivitu 0.79 neboli 79% a pro více proměnných byla celková úspěšnost 83.9%. S těmito výsledky se dá říci, že model by dokázal předpovídat, zdali bude pršet druhý den velmi přesně.

7 Seznam použité literatury

[1] HAN, Jiawei; KAMBER Micheline. Data mining: Concepts nad Techniques. USA: Elsevier, 2006. 745. ISBN 13:978-1-55860-901-3.

[2] SKALSKÁ, Hana. Data Mining a klasifikační modely. Hradec Králové: GAUDEAMUS, 2010. 154. ISBN 978-80-7435-088-7

[3] What is the CRISP-DM methodology?. Home - Smart Vision - Europe [online]. [cit. 24.03.2020]. Dostupné z: https://www.sv-europe.com/crisp-dm-methodology/

[4] Data Mining and the Case For Sampling, A SAS Institute Best Practice Paper. [online]. [cit. 24.03.2020]. Dostupné z: https://sceweb.uhcl.edu/boetticher/ML_DataMining/SAS-SEMMA.pdf

[5] HAMPTON Jessica. SEMMA and CRISP-DM: Data Mining Methodologies. [online]. [cit. 24.03.2018] Dostupné z: http://jesshampton.com/2011/02/16/semma-and-crisp-dm-data-mining-methodologies/

[6] NAGABHUSHANA S.. Data Warehousing: OLAP and Data Mining. New Age International, 2006. 323. ISBN 978-81-224-2705-9

[7] TUTA Andrei. Data Mining 101 Dimensionality and Data reduction. [online]. [cit. 24.03.2018]. Dostupné z: https://towardsdatascience.com/data-mining-101-dimensionality-and-data-reduction-2a8fa427b092

[8] GARETH James, WITTEN Daniela, HASTIE Trevor, TIBSHIRANI Robert. An Introduction to Statistical Learning. USA. Springer, 2013. ISBN 978-1-4614-7138-7

[9] WILLIAMS Graham. Data mining with Rattle and R: The art of excavating data for Knowledge Discovery. USA. Springer, 2011. ISBN 978-1-4419-9889-7

[10] The R Journal Volume 1/2. Rattle: A data Mining GUI for R. USA: Graham Williams, 2009. ISSN 2073-4859

Příloha č. 1

UNIVERZITA HRADEC KRÁLOVÉ
Fakulta informatiky a managementu
Akademický rok: 2018/2019

Studijní program: Systémové inženýrství a informatika
Forma studia: Prezenční
Obor/kombinace: Informační management (im3-p)

Podklad pro zadání BAKALÁŘSKÉ práce studenta

Jméno a příjmení: **David Illner**
Osobní číslo: **I1600689**
Adresa: Palackého 887, Úpice, 54232 Úpice, Česká republika

Téma práce: Data mining a možnosti nekomerčního softwaru
Téma práce anglicky: Data mining with freeware

Vedoucí práce: prof. RNDr. Hana Skalská, CSc.
Katedra informatiky a kvantitativních metod

Zásady pro vypracování:

Cíl: Úloha data mining v procesu vyhledávání znalostí z dat (Knowledge Data Discovery) a možnosti nekomerčního softwaru (například Rattle, Orange nebo Weka) pro vybrané typy úloh.
Osnova:

1. Úvod
2. Rešerše literatury, Vývoj a popis data mining, vysvětlení termínů této oblasti.
3. Typy úloh data mining
4. Pro zvolený typ úlohy popsat metodu a nástroje řešení.
5. Vyhledávání dat pro analýzu, návrh řešení a jeho realizace ve zvoleném softwaru, popis řešení a výsledků.
6. Shrnutí výsledků a závěr.

Seznam doporučené literatury:

HAN, Jiawei; KAMBER Micheline. Data mining: Concepts nad Techniques. USA: Elsevier, 2006. 745. ISBN 10: 1-55860-901-6. ISBN 13:978-1-55860-901-3
SKALSKÁ, Hana. Data Mining a klasifikační modely. Hradec Králové: GAUDEAMUS, 2010. 154. 978-80-7435-088-7
GARETH James, WITTEN Daniela, HASTIE Trevor, TIBSHIRANI Robert. An Introduction to Statistical Learning. USA. Springer, 2013. 978-1-4614-7138-7

Podpis studenta: Datum:

Podpis vedoucího práce: Datum: