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

Analýza dat s využitím jazyků R a JAVA

Bakalářská práce

Autor: Pavel Vojtas
Studijní obor: Aplikovaná informatika

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

Hradec Králové 04.2015

Prohlášení:

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

V Hradci Králové dne: 22.4.2015 Pavel Vojtas

Poděkování:

Děkuji vedoucí bakalářské práce Prof. RNDr., Haně Skalské, CSc. za metodické vedení práce, cenné rady a uživatelský pohled na vyvíjený software.

Anotace

Tato bakalářská práce se zabývá softwarovým řešením analýzy datových souborů. Věnuje se možnostem využití existujícího softwaru a v něm použitých analytických metod. Popisuje sjednocení různých technologií, konkrétně R a Java. Na jejichž základě je vystavěn vlastní analytický software JRA (Java-R Analysis) pro analýzu datových souborů, jehož výhoda spočívá v jednoduchosti a intuitivní dostupnosti analytických metod. V závěru práce je software JRA porovnán s existujícími řešeními.

Annotation

Title: Data analysis using the R language and Java

This bachelor thesis follows up possibilities for data sets analysis. It describes existing software and explains the principles of selected analytical methods. Similarities and linkage between two different technologies are describe here, specifically between R language and Java. On this base of Java and R, own analytical software JRA (Java-R Analysis) was built for analyzing data sets, whose advantages are the simplicity and intuitive availability of analytical methods. Possibilities of the JRA and its comparison with existing software solutions are also presented here.

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

V oblasti analýzy datových souborů existují různé produkty zabývající se touto problematikou. Nejrozšířenější aplikací je MS Excel. Ten ale obsahuje pouze omezené analytické metody. Pro složitější metody je potřeba vzít Pro složitější metody je potřeba vzít specializovaný statistický software (zpravidla komerční) nebo speciální programovací jazyky.

Profesionální analytický software bývá úzce specializován na určité okruhy aplikačních oblastí a je vyvíjen většinou komerčně. Analytické metody jsou v nich plně integrované, uživatelsky a intuitivně dostupné. Výsledky analýz jsou profesionálně strukturované. Pro použití analytických metod jiné aplikační oblasti bývá nutné použít software jiný.

Speciální programovací jazyky jsou typy softwarů s nejširší možností specializace. Tento software je většinou modulární a funkce jsou seskupeny ve formě externích knihoven. Tímto se dá software přizpůsobit pro široké odvětví. S tím ale vzniká potřeba přizpůsobení uživatele/analytika, nebo potřeba více expertů na různé oblasti. Kvůli velkému množství knihoven není pro každé funkce dostupné intuitivní a uživatelsky přívětivé rozhraní.

2 Cíl práce a metodika zpracování

Cílem této práce bude problematika analýzy dat a jejich zpracování pomocí různých typů softwaru. Zaměří se na metody jazyka R a na vytvoření vlastního softwaru. Dalším cílem práce bude možnost sjednocení technologií R a Java. Na základě těchto poznatků bude sestaven vlastní software. Ten bude psán v jazyce Java a bude implementovat metody jazyka R. Dále se bude porovnávat vlastní implementace s existujícími řešeními.

Literární rešerše

Oblast analýzy dat je velice rozsáhlá. Nejedná se pouze o statistické metody, ale i o metody pro získávání znalostí (Data Mining). Mezi tyto metody například patří Klasifikační stromy, Neuronové sítě, aj. Metody data miningu a jejich použití vychází z literatury[5],[6],[8]. Jsou zde popsány metody statistické i klasifikační a rozhodovací. Dále jsou zde doplněny možnosti využití výsledků těchto metod.

Zdrojem dat k analýze mohou být výsledky z dotazníkových šetření, výzkumu, statistik prodeje či návštěvnosti. Tyto zdrojová data se pomocí speciálního softwaru, například IBM SPSS, R aj., zpracují. Zpracování probíhá aplikováním patřičných analytických metod na zdrojová data. Tento princip a jednotlivé kroky zpracování jsou dobře popsány v literatuře [7], která se zabývá primárně získáním zdrojových dat z dotazníkového šetření a jejich zpracování v softwaru IBM SPSS.

Výsledky zvolených analytických metod se mohou využívat například při rozhodovacích procesech, marketingu nebo výzkumu.

Metodika

První část práce vychází z rešerše odpovídající literatury a bude se zabývat z větší části teoretickým úvodem do problémové oblasti a přehledem používaného softwaru, ze kterého budou vytvořeny modely pro analýzu dat. Z těchto modelů se identifikují a popíší analytické metody a jejich implementace. Na základě těchto poznatků budou další části práce směřovat k návrhu vlastního softwaru.

Druhá část práce vychází ze sestaveného modelu pro analýzu dat a bude se zabývat vývojem vlastního softwaru JRA. Nejprve se bude zabývat grafickou a uživatelskou částí s možnostmi ukázek použití. Následovat bude uvedení do vnitřní implementace softwaru. Na závěr bude zhodnocení implementace JRA.

3 Úvod do problematiky analýzy dat a popis vybraného softwaru

Tato část se zabývá problematikou analýzy dat obecně a možnostmi softwarových řešení. Vysvětlí obecné postupy analýzy pro běžného uživatele, přinese přehled statistického softwaru a popis jejich typických vlastností a funkcí.

Analýza dat

Analýzou dat se v této práci rozumí zpracování datového souboru. Datový soubor obsahuje proměnné a jejich hodnoty. Může se jednat o soubor hodnot z výzkumu, dotazníkového šetření, aj.

Zpracování datového souboru je možné v několika variantách. Používají se přitom různé statistické a pravděpodobnostní metody. Pro zjednodušení a zefektivnění práce a výpočtů existuje mnoho druhů softwaru implementujících tyto metody.

Analytický software existuje ve free variantě i v placené variantě. Ve free variantě se jedná např. o tabulkový editor, R. Mezi software nabízející obě varianty patří např. RExcel. Také existuje profesionální software, který je pouze v placené variantě - např. IBM SPSS.

Software SIMSTAT

Jedná se o offline aplikaci s nutností instalace. Po nainstalování a spuštění aplikace se otevře prostředí obsahující 4 perspektivy (okna). Jedná se o perspektivy *notebook*, *data*, *chart*, *script*.

Po importu datového souboru se otevře perspektiva *data*. V této perspektivě lze upravovat jednotlivé proměnné datového souboru. Dále se po stisknutí pravého tlačítka myši zobrazí nabídka „rychlých akcí“. Ta obsahuje metodu pro úpravy proměnných a metodu pro výpočet statistik (POZOR! Výpočet statistik pomocí „rychlé akce“ funguje pouze pro numerickou proměnnou).

Pro plnohodnotnou práci s datovým souborem slouží nabídka menu. Kategorie v nabídce jsou hlavně pro úpravu proměnných a výpočet statistik.

Pro výpočet statistik jsou možnosti deskriptivní statistiky, tvorby tabulek, srovnávacích metod, regresí, atd. Nejprve je ale potřeba vybrat jaké proměnné analyzovat. K tomu slouží nabídka *Choose X-Y*.

Po provedení určité analytické metody se otevře *notebook*, kde budou k dispozici výsledky metody. Tyto výsledky se poté dají exportovat ve formě prostého textu.

Po výběru perspektivy *charts* je možno prohlížet grafy, které jsou k dispozici. Pro úpravu grafů existuje v menu možnost C*harts*.

Např: Zjištění četnosti dvou kategoriálních proměnných (kontingenční tabulka). Nejprve nutno specifikovat závislou a nezávislou proměnnou, **Statictics -> Choose X-Y**. Poté vybrat možnost vytvoření kontingenční tabulky, **Statistics -> Table -> Crosstable**. Po nastavení možností analýzy se metoda provede. Výsledek je následně dostupný v perspektivě *notebook*.

Tento popis vzniknul z testování trial verze, více na webu [1].

Software SPSS

Jedná se o offline aplikaci. Popis vychází z uživatelských zkušeností a manuálu [2]. Po spuštění aplikace se otevře tabulkový editor, perspektiva *data*. Dále je k dispozici perspektiva *variable*, kde lze specifikovat podrobnosti jednotlivých proměnných, např. typ proměnné, chybějící hodnoty, zarovnání textu, specifikaci proměnné (ordinal, nominal, …). Rozhraní dále obsahuje menu pro rozšířenou práci s datovým souborem. Pro rychlé volby obsahuje grafický panel s vybranými funkcemi.

Při otevření souboru s daty se otevře průvodce importem. V něm lze specifikovat typ otevíraného souboru a způsob importu. Dále zadání názvů a typů proměnných.

Pro analýzu načteného datového souboru slouží z menu nabídka *Analyse*. Ta obsahuje metody pro analýzu deskriptivní statistiky, tabulek, srovnání průměrů, lineární

modely, regrese, aj.

Analýza deskriptivních charakteristik pracuje pouze s numerickými proměnnými. Vypočítá charakteristiky jako rozsah souboru, minimum, maximum, průměr, směrodatná odchylka.

Dále je možnost vytvoření tabulky četností a kontingenční tabulky. Při tvorbě tabulky četností nutno specifikovat proměnné pro analýzu. Kategorie pro numerické proměnné jsou ve výchozím stavu jednotlivé hodnoty, ne intervaly. Dále lze specifikovat typ grafu pro reprezentaci tabulky četností. Obdobný princip platí při vytváření kontingenční tabulky. Zde je ale nutno specifikovat proměnnou X a proměnnou Y.

Výsledky jsou zaznamenávány do speciálního okna, perspektivy *view*. Zde jsou k dispozici všechny reporty. Reporty lze poté exportovat do souboru ve formě DOC, PDF, či jako prostý text.

Good Data

Jedná se o online, cloud, aplikaci. Detailnější popis se nachází v referenční příručce [3]. Prvním krokem k používání je vytvoření a zaregistrování uživatele. Přístup do prostředí probíhá skrz portál GoodData.

Po prvním přihlášení se spustí průvodce, který uživatele seznámí s možnostmi a funkcemi prostředí. Samotné prostředí se skládá z tzv. *Dashboards*, dále z kategorie *Report* a kategorie *Search*.

Dashboard jsou pracovní listy obsahující reporty a výsledky k jednotlivým analýzám v rámci jednoho projektu. Tyto listy se dají individuálně přizpůsobovat.

Report je kategorie pro vytváření jednotlivých reportů. Průvodce vyzve uživatele k zadání parametrů pro analýzu, tj. výběr proměnné a výběr analytické metody. Jednotlivé analytické metody jsou na výběr z přednastavených, nebo je zde možnost napsat si vlastní metody s využitím jazyka MAQL. K tomu je ale potřeba seznámení

s tímto jazykem. Po provedení analýzy je zde možnost výběru reprezentace výsledků. Reprezentace pomocí tabulky, či pomocí grafů. Výsledný report se použije v patřičném dashboardu.

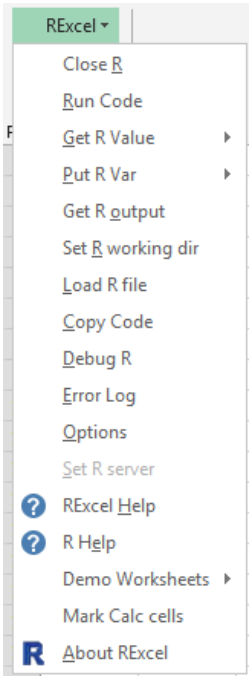
Prostředí dále umožňuje import dat ze souboru CSV pomocí nástroje CSV Uploader. Po načtení souboru vyzve uživatele k nastavení typů proměnných. Úprava datového souboru je dále možná ze sekce *Manage*.

RExcel

MS Excel je rozšířený tabulkový editor. Za vývojem stojí společnost Microsoft a je součástí balíčku Microsoft Office. Tento tabulkový editor obsahuje základní funkce pro statistické, matematické a další výpočty. Dále obsahuje základní možnosti vizualizace, většinou ve formě grafů. Pro složitější, profesionálnější statistické funkce však nenabízí uspokojivé řešení.

Jedna z možností řešení je použití pluginu RExcel, viz [4]. Tento plugin importuje možnosti využití R jazyka uvnitř MS Excel. Tím MS Excel získá navíc možnosti R jazyka, ale oproti čistému R jazyku nabízí přívětivé uživatelské prostředí. Plugin je bezplatně dostupný pro nekomerční použití pod licencí Home and Student.

K instalaci pluginu je potřeba nainstalovat nejprve DCOM server. (více informací na http://rcom.univie.ac.at/). Poté lze spustit instalaci samotného RExcelu. Při instalaci se plugin zároveň aktivuje v nainstalované verzi MS Excel.



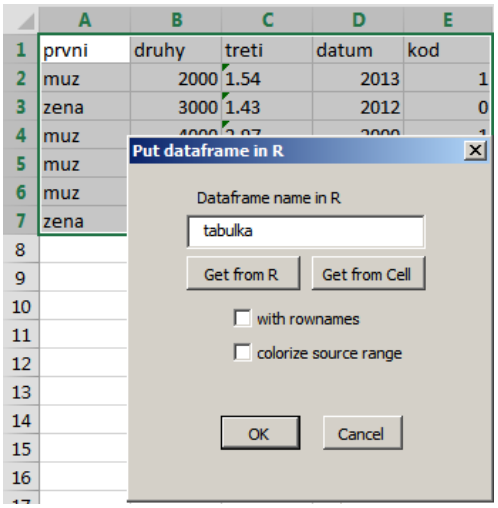
Obrázek 1: Možnosti pluginu RExcel

Po spuštění MS Excel je plugin dostupný v sekci doplňky. Nejprve je potřeba spustit přes **Doplňky -> RExcel -> Start R**. Po spuštění se aktivují další možnosti. Možnost **Run Code**, umožní spuštění textu v označené buňce jako příkaz jazyka R. Možnost **Get R Value**, umožní získat obsah proměnné uložené v R. **Put R Value,** umožní uložení označených buněk do proměnné v R. **Get R Output,** vypíše výsledek naposledy použitých příkazů,

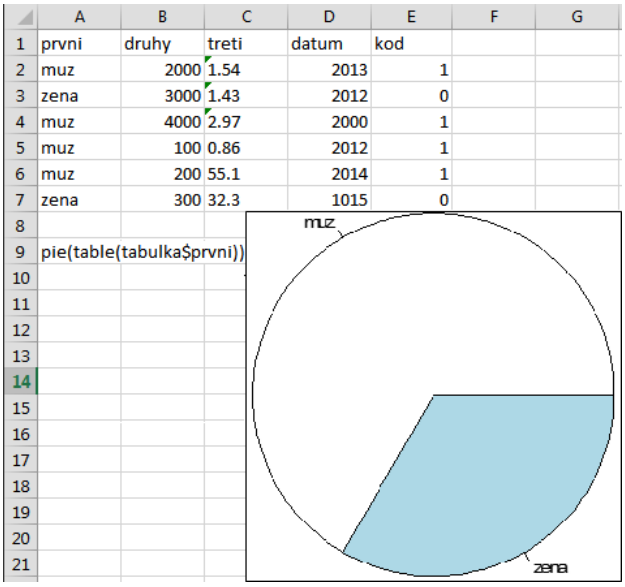
popř. zpráv warning, či error. **Load R file** umožní načíst externí R skripty. Zastavení pluginu provádí možnost **Close R**.

Práci s pluginem popisuje následující příklad:

> Po otevření datového souboru se označí oblast pro požadovanou analýzu. Oblast se uloží do R proměnné typu data frame, kterou si uživatel zvolí zadáním názvu, viz Obrázek 2. Napsáním příkazu **table()** a **pie()**, se vypočítají četnosti proměnné a vykreslí se kruhový graf, viz Obrázek 3.



Obrázek 2: RExcel výběr proměnné



Obrázek 3: RExcel příkazy

Závěry

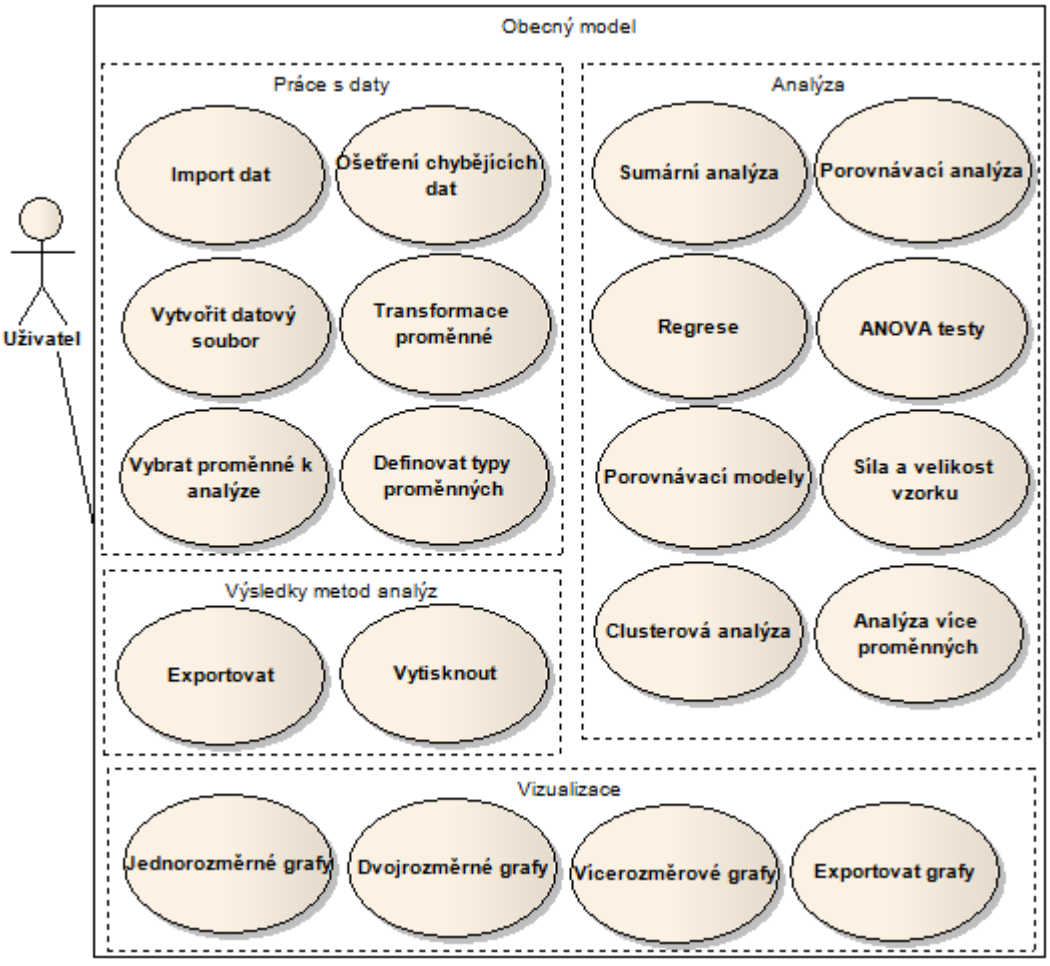
Z výše popsaných softwarů se pro další část práce vyberou typické základní funkcionality. Z těchto funkcionalit se v následující části bude sestavovat obecný model analýzy dat. Tento model podchytí základní funkce a graficky je zobrazí.

4 Modely analýzy dat

Model pro analýzu dat má sloužit pro zobrazení funkcionalit existujících softwarů. Další význam modelu je zobrazení aktérů pracujících se softwarem a zobrazení vazeb na funkcionality, které mohou obsluhovat. Z tohoto modelu se vychází při sestavování scénářů jednotlivých funkcionalit, které je rozšiřují o sekvenční kroky vedoucí k jejich splnění.

4.1 Obecný model

Obecný model bude vytvořen na základě vybraných analytických softwarů a literatury [7],[8]. Z tohoto modelu bude vycházet návrh, podle kterého bude vyvíjena aplikace JRA.

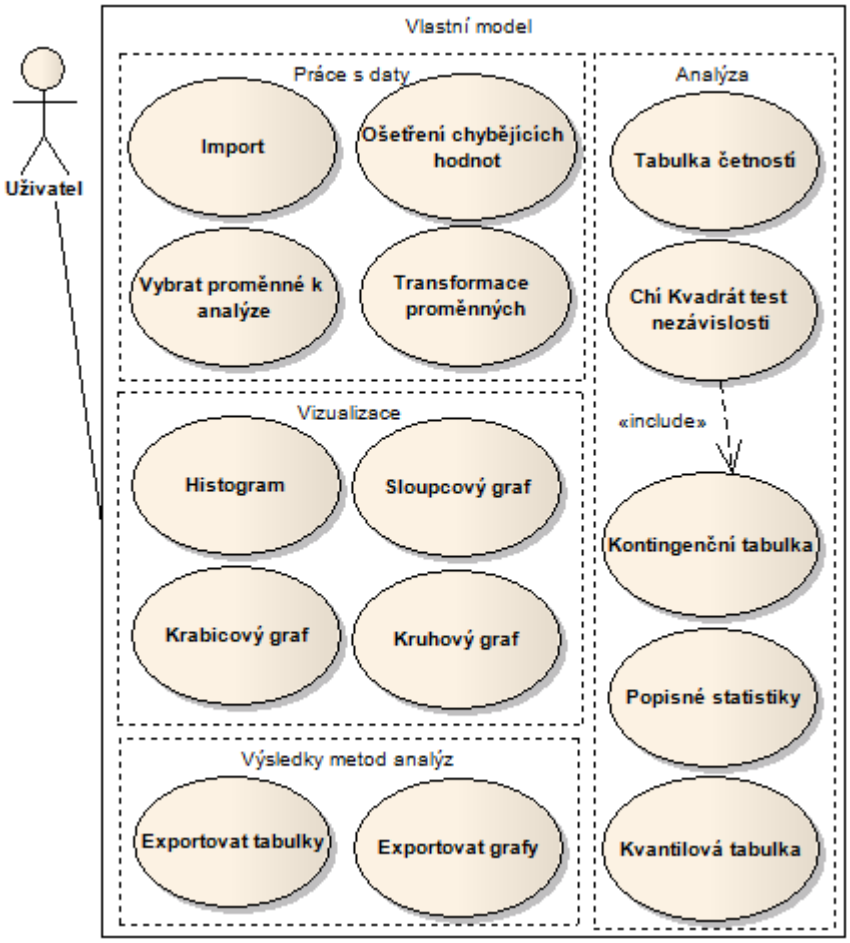


Obrázek 4: Obecný model analýzy dat

Diagram modelu užití popisuje stavbu aplikace z hlediska funkčnosti. Pro přehlednost jsou jednotlivé funkce rozděleny do kategorií. Jedná se o systém, který má pouze jednoho aktéra. Ten využívá každou funkci systému. Pro přehlednost je vazba USE vedena pouze na systém jako celek. Základní funkce jsou popsány na obrázku výše.

4.2 Vlastní model

Tento model je zjednodušením obecného modelu a obsahuje pouze vybrané funkce, které jsou použity pro vývoj vlastního softwaru JRA.



Obrázek 5: Vlastní model analýzy dat

Diagram výše popisuje vybrané funkcionality, které se budou implementovat do vlastního uživatelského rozhraní. Pro přehlednost jsou funkce rozdělené do kategorií **Práce s daty, Analýza, Vizualizace, Výsledky metod analýz**. Návrh systému počítá, jako v minulém případě, s jedním aktérem, který bude využívat všechny funkce systému. Z hlediska přehlednosti je vazba USE vedena na systém jako celek.

4.3 Scénáře vlastního modelu

Každá funkcionalita vlastního modelu se skládá ze scénáře, tedy číslované sekvence kroků, která vede k vyřešení dané funkcionality. Z těchto scénářů a jejich kroků se identifikují konkrétní statistické a pravděpodobnostní metody pro analýzu.

Import

Typová úloha se zabývá načtením datové tabulky ze souboru a vizualizací ve formě tabulky v uživatelském rozhraní. Tento soubor musí být ve formátu CSV, text oddělovaný středníkem.

Scénář:

1. Aktér zavolá import.
2. Systém zobrazí dialog pro výběr CSV souboru.
3. Aktér vybere CSV soubor.
4. Systém načte CSV soubor a vytvoří tabulku.
5. Systém zobrazí tabulku.

Ošetření chybějících hodnot

Typová úloha se zabývá výběrem a nastavením chybějících hodnot pro vybranou proměnnou.

Scénář:

1. Aktér zavolá ošetření chybějících hodnot.
2. Systém načte vybranou proměnnou.
3. Systém zobrazí dialog pro výběr hodnoty.
4. Aktér vybere hodnotu.
5. Systém nastaví hodnotu na chybějící a zobrazí upravenou tabulku.

Transformace proměnných

Typová úloha se zabývá výběrem proměnné a změnou typu proměnné na kategorickou či numerickou.

Scénář:

1. Aktér zavolá transformaci proměnných.
2. Systém načte vybranou proměnnou.

3. Systém zobrazí dialog pro změnu typu.

4. Aktér vybere typ.

5. Systém změní typ proměnné.

5.1. Pokud změna neproběhla v pořádku.

5.1.1 Systém zobrazí chybu

5.1.2. Systém vrátí zpět změny.

5.2. Pokud změna proběhla v pořádku.

5.2.1 Systém uloží změny.

Vybrat proměnné k analýze

Typová úloha se zabývá výběrem jedné či více proměnných. Z těchto proměnných je pak sestavena podmnožinová tabulka. Proměnné v ní obsažené jsou následně analyzovány.

Scénář:

1. Aktér zavolá výběr proměnné k analýze.

2. Systém zobrazí dialog pro výběr proměnných.

3. Aktér vybere jednu či více proměnných.

4. Systém vytvoří z vybraných proměnných tabulku.

5. Systém zobrazí nově vytvořenou tabulku.

Tabulka četností

Typová úloha se zabývá výpočtem tabulky četností z jednotlivých proměnných z podmnožinové tabulky. Tabulka četností vypočítá absolutní, relativní, kumulativní četnosti.

Scénář:

1. Aktér zavolá tabulku četností.

2. Systém načte proměnnou z podmnožinové tabulky.

2.1. Pokud neexistuje další proměnná, systém zobrazí tabulku četností.

3. Systém vypočte tabulku četností pro vybranou proměnnou.

4. Systém pokračuje bodem 2.

Popisné statistiky

Typová úloha se zabývá výpočtem popisných charakteristik. Vypočtené charakteristiky poté zobrazí ve formě tabulky.

Scénář:

1. Aktér zavolá výpočet popisné statistiky.
2. Systém načte proměnnou z podmnožinové tabulky.

2.1. Pokud neexistuje další proměnná, systém zobrazí tabulku charakteristik.

3. Systém vypočte tabulku charakteristik pro vybranou proměnnou.
4. Systém pokračuje bodem 2.

Kvantilová tabulka

Typová úloha se zabývá výpočtem kvantilové tabulky. Kvantilová tabulka bude obsahovat 0%, 25%, 50%, 75%, 100% kvantil.

Scénář:

1. Aktér zavolá výpočet kvantilové tabulky.
2. Systém načte numerickou proměnnou z podmnožinové tabulky.

2.1. Pokud neexistuje další proměnná, systém zobrazí tabulku kvantilů.

3. Systém vypočte tabulku kvantilů pro vybranou proměnnou.
4. Systém pokračuje bodem 2.

Kontingenční tabulka

Typová úloha se zabývá výpočtem kontingenční tabulky pro dvě proměnné.

Scénář:

1. Aktér zavolá výpočet kontingenční tabulky.
2. Systém zobrazí formulář pro zadání dvou názvů proměnných.
3. Aktér zadá dvě proměnné.
4. Systém vypočte kontingenční tabulku.
5. Systém zobrazí kontingenční tabulku.

Chí kvadrát test nezávislosti

Typová úloha se zabývá výpočtem Chí kvadrát testu nezávislosti. Test určuje, zda jsou proměnné obsažené v kontingenční tabulce na sobě vzájemně závislé či nikoli.

Z toho plyne, že tato typová úloha je závislá na výsledku typové úlohy Vypočítat kontingenční tabulku

Scénář:

1. Aktér zavolá výpočet Chí kvadrát testu nezávislosti.
2. include Vypočítat kontingenční tabulku.
3. Systém vypočte chí kvadrát test.
4. Systém zobrazí výsledky testu.

Histogram

Typová úloha se zabývá vykreslením histogramu pro numerickou proměnnou. Dále umožňuje výběr barev pro vykreslení.

Scénář:

1. Aktér zavolá vykreslení histogramu.
2. Systém zobrazí formulář pro výběr numerické proměnné.
3. Aktér vybere proměnnou.
4. Systém zobrazí dialog pro výběr barev grafu.
5. Aktér zvolí bravy grafu.
6. Systém vykreslí histogram.

Sloupcový graf

Typová úloha se zabývá vykreslením sloupcového grafu pro tabulku četností. Dále umožňuje výběr barev pro vykreslení.

Scénář:

1. Aktér zavolá vykreslení sloupcového grafu.
2. Systém zobrazí formulář pro výběr tabulky četností a typu četnosti.
3. Aktér vybere tabulku četností a zvolí typ četnosti.
4. Systém zobrazí dialog pro výběr barev grafu.
5. Aktér zvolí bravy grafu.
6. Systém vykreslí sloupcový graf

Kruhový graf

Typová úloha se zabývá vykreslením kruhového grafu pro tabulku četností. Dále umožňuje výběr barev pro vykreslení.

Scénář:

1. Aktér zavolá vykreslení kruhového grafu.
2. Systém zobrazí formulář pro výběr tabulky četností a typu četnosti.
3. Aktér vybere tabulku četností a zvolí typ četnosti.
4. Systém zobrazí dialog pro výběr barev grafu.
5. Aktér zvolí bravy grafu.
6. Systém vykreslí kruhový graf

Krabicový graf

Typová úloha se zabývá vykreslením krabicového grafu pro tabulku kvantilů. Dále umožňuje výběr barev pro vykreslení.

Scénář:

1. Aktér zavolá vykreslení krabicového grafu.
2. Systém zobrazí formulář pro výběr tabulky kvantilů.
3. Aktér vybere tabulku kvantilů.
4. Systém zobrazí dialog pro výběr barev grafu.
5. Aktér zvolí barvy grafu.
6. Systém vykreslí krabicový graf

Exportovat tabulky

Typová úloha se zabývá exportem tabulek do externího souboru.

Scénář:

1. Aktér zavolá exportování tabulky.
2. Systém zobrazí formulář pro výběr typu tabulky.
3. Aktér vybere typ tabulky.
4. Systém zobrazí dialog pro výběr souboru.
5. Aktér vybere soubor pro uložení.
6. Systém uloží tabulku.

Exportovat grafy

Typová úloha se zabývá exportem aktuálně zobrazeného grafu do externího grafického souboru.

Scénář:

1. Aktér zavolá exportování grafu.
2. Systém zjistí aktuálně otevřený graf.
3. Systém zobrazí dialog pro výběr souboru.
4. Aktér vybere soubor pro uložení.
5. Systém uloží graf.

5 Jazyk R a vybrané knihovny

Znalost jazyka R je důležitá k pochopení implementace analytických metod, identifikovaných z vlastního modelu a použitých ve vývoji JRA. K popisu jazyka R jsou využity znalosti získané z literatury [9].

5.1 Úvod do jazyka R, datové typy a struktury

Jedná se o programovací jazyk vyšší úrovně. Vychází ze dvou jazyků: jazyka S a Scheme. Jedná se o objektový jazyk. Používá se pro statistickou analýzu a zobrazování výsledků. Na rozdíl od jazyka S se jedná o freeware implementaci.

Datové typy

R jazyk pracuje se pěti základními datovými typy neboli *mody*. Jsou to: numerický (numeric), komplexní (complex), znakový (character), logický (logic) a neurčitý (raw). Všechny datové typy jsou objekty. Dále se zde vyskytuje speciální „hodnota" typu NA, označující chybějící údaje. Typ a velikost datového typu lze zjistit příkazy mode(x) a length(x).

Datové struktury

V R jazyce se dále vyskytují datové struktury. Jedná se o struktury typu: vektor (vector), pole (array), matice (matrix), seznam (list) a datová tabulka (data frame).

Vektor

Vektor označuje obecně posloupnost hodnot. Hodnoty vektoru musí být vždy stejného datového typu.

Vytvoření:

Vektor x se skládá ze čtyř numerických hodnot 1, 2, 3, 4

```
> x <- c(1,2,3,4)
```

Vektor x se skládá ze sekvence numerických hodnot od 1 do 5.

```
> x <- 1:5
```

Vektor y se skládá z dvou po sobě jdoucích vektorů x ukončených 0.

```
> y <- c(x,x,0)
```

Tedy zadání vektoru může proběhnout výčtem prvků nebo zadáním prvního a koncového čísla. Operátoru přiřazení (<-) odpovídá i funkce `assign()`. Dále se vektor může vytvořit pomocí jiného vektoru.

Sekvence:
Při vytvoření vektoru lze vytvořit sekvenci hodnot, viz výše. Tato sekvence se dá také vytvořit pomocí funkce `seq()`. Při použití funkce se může upřesnit kromě počáteční a koncové hodnoty také přírůstek.

Např.: sekvence od 1 do 10 s přírůstkem 2,5.

```
> x <- seq(1, 10, by=2.5)
> x
[1] 1.0 3.5 6.0 8.5
```

Aritmetika:
Při výpočtu se používají elementární operátory +, -, *, /. Mezi další funkce při výpočtu patří aritmetické metody (`log()`, `exp()`, `cos()`, aj.) a statistické metody (`max()`, `min()`, `sum()`, aj.). Operace probíhají nad celým vektorem, tj. nad všemi jeho hodnotami.

Např.: odmocnina z vektoru x obsahující čísla 1, 2, 3, 4

```
> x <- c(1,2,3,4)
> sqrt(x)
[1] 1.000000 1.414214 1.732051 2.000000
```

Pole
Pole se skládá s pod-kolekce datových záznamů. Pole musí obsahovat atribut označující dimenzi *dim*. Obecně může být pole k-rozměrné. Dvourozměrné pole je matice.

Vytvoření:

```
> p <- array(datovy_vektor, dimenze)
```

Matice

Jedná se o speciální typ pole, konkrétně dvourozměrné pole. Existuje mnoho způsobu vytvoření.

Vytvoření matice pomocí funkce matrix(prvky, podmínka):

Matice obsahující sekvenci od 1 do 21 s pevným počtem řádků

```
> matrix(1:21,nrow=3)
     [,1][,2][,3][,4][,5][,6][,7]
[1,]  1   4   7   10  13  16  19
[2,]  2   5   8   11  14  17  20
[3,]  3   6   9   12  15  18  21
```

Matice obsahující sekvenci od 1 do 21 s pevným počtem sloupců

```
>  matrix(1:21,ncol=3)
     [,1][,2][,3]
[1,]  1   8   15
[2,]  2   9   16
[3,]  3   10  17
[4,]  4   11  19
[5,]  5   12  19
[6,]  6   13  20
[7,]  7   14  21
```

Matice 3x3 naplněná sekvenci od 1 do 21.

```
>  matrix(1:21,nrow=3,ncol=3)
     [,1][,2][,3]
[1,]  1   8   15
[2,]  2   9   16
[3,]  3   10  17
```

Vytvoření matice po sloupcích a po řádcích pomocí funkcí rbind() a cbind()

Matice obsahující hodnoty 0,0,0 a 1,1,1 vytvořená po řádcích

```
> rbind(c(0,0,0),c(1,1,1))
```

```
     [,1][,2][,3]
[1,]  0   0   0
[2,]  1   1   1
```

Matice obsahující hodnoty 0,0,0 a 1,1,1 vytvořená po sloupcích

```
> cbind(c(0,0,0),c(1,1,1))
     [,1][,2]
[1,]  0   1
[2,]  0   1
[3,]  0   1
```

Pojmenování sloupců a řádků:

Pomocí funkcí rownames() a colnames(). Lze pojmenovat řádky a sloupce matice.

```
> rownames(m) <- c("prvni","druhy","treti")
> m
     [,1][,2][,3]
prvni  1   0   0
druhy  0   1   0
treti  0   0   1

> colnames(m) <- c("jedna","dva","tri")
> m
      jedna dva tri
prvni   1    0   0
druhy   0    1   0
treti   0    0   1
```

Seznam

Seznam je jedna z nejdůležitějších struktur v R jazyce. Skládá se ze skupiny komponentů (objektů), kde na rozdíl od pole mohou být objekty různého datového typu a délky.

Vytvoření:

Pomocí funkce list(vektor, vektor, ..., vektor) se vytvoří seznam.

```
> seznam <- list(c(1:3),c("Adam","Eva"))
> seznam
[[1]]
[1] 1 2 3

[[2]]
[1] "Adam" "Eva"
```

Datová tabulka

Na rozdíl od seznamu může datová tabulka obsahovat vektory libovolného datového typu, ale všechny vektory musí mít stejnou délku. Zjištění počtu sloupců a řádků se poté provádí funkcemi nrow(), ncol().

Vytvoření:

Pomocí funkce data.frame(vektor, vektor, ..., vektor).

```
> tabulka <- data.frame(c(1:2), c("Adam","Eva"))
> tabulka
  c.1.2. c..Adam....Eva..
1      1            Adam
2      2            Eva
```

5.2 Funkce v R

Existuje mnoho předprogramovaných funkcí, např. `min()`, `max()`, `sum()`, `median()`, aj. Kromě toho se mohou vytvářet i funkce vlastní.

Při vytváření funkcí vlastních se využívá klíčového slova `function(argumet){tělo}`. Vlastní tělo funkce (posloupnost příkazů pro vykonání) se uzavírá do složených závorek. Využívají se zde mechanismy podmínek, větvení a cyklů.

Podmínky se označují klíčovými slovy if, else. Příkazy ve větvi if jsou splněny, pokud je splněna podmínka. Příkazy ve větvi else pokud splněna není. Celková syntaxe je `if(podmínka){příkazy} else{příkazy}`.

Větvení se označuje klíčovým slovem switch. Na základě argumentu jsou porovnávány prvky seznamu. Celková syntaxe je `switch(argument, seznam)`.

Cykly slouží k opakovanému provádění příkazů. Existují tři typy cyklů: Cyklus **repeat** opakuje blok příkazů, dokud není požadováno ukončení. Při cyklu **while** se udává ukončovací podmínka a cyklus se opakuje do té doby, dokud je podmínka splněna. Při cyklu **for** se udává jméno proměnné a vektor procházení.

Vytvoření funkce, kde uživatelem zadané číslo je porovnáno s náhodným číslem.

```
> funkce <- function(cislo){
      nahodne <- sample(1:10, 1)
      if(nahodne < cislo){
            print("cislo je vetsi nez")
      }
      else if(nahodne > cislo){
            print("cislo je mensi nez")
      }
      else {
            print("trefa")
      }
      print(nahodne)
 }
```

5.3 Knihovny plotrix, rJava, JavaGD

Jsou zde popsány pouze knihovny, které nejsou primární součástí R jazyka. Jedná se o knihovny plotrix, rJava, JavaGD. U každé knihovny je napsáno k jakým účelům se používá, popis instalace a popis nastavení knihovny.

rJava

Za vývojem knihovny rJava [13] stojí Simon Urbanek z projektu RForge. Knihovna vytváří low-level interface[1] mezi virtuálním stojem JAVY (JVM) a mezi R jazykem. Tedy umožňuje vytvářet instance a volat funkce z jazyka Java uvnitř jazyka R.

Součástí knihovny rJava je také modul zvaný JRI. Tento modul poskytuje opačný přístup. Tedy dovoluje vytvářet instance a volat funkce jazyka R uvnitř Java kódů. Tento modul je v této práci klíčový.

Modul se skládá ze tří knihoven Java. Jedná se o knihovny *JRI.jar, JRIEngine.jar, REngine.jar*. Dále z nativní knihovny operačního systému *jri.dll* u systémů Windows a *libjri.so* v unixových systémech.

Před použitím knihovny je potřeba načíst Java knihovny do build path v Java projektu. Poté je třeba nastavit proměnné prostředí R_HOME, odkazující na domovský adresář s programem R. Dále je potřeba nastavit parametr virtuálního stroje *java.library.path* odkazující na nativní knihovnu JRI.

Načítáni knihovny v R jazyce příkazem `library()` se dále neprovádí.

JavaGD

Za vývojem knihovny JavaGD [14] stojí Simon Urbanek z projektu RForge. Knihovna umožňuje směrovat grafické operace z R jazyka do příslušné třídy Java. Tato třída se poté může použít jako grafická komponenta ve Swing nebo AWT implementaci.

Knihovna obsahuje soubor JavaGD.jar. Ten se musí nastavit do build path Java projektu.

Důležitým krokem je zde nastavení směrování grafických operací. Je potřeba v R jazyce nastavit cíl směrování, konkrétní třídu Java. To se provádí pomocí proměnných

[1] Low-level interface dovoluje programátorovi zasahovat do funkcí uvnitř SW moulu nebo uvnitř HW.

prostředí v R jazyce. Další podmínkou je, že cílová Java třída musí dědit ze třídy GDInterface.java. Tato třída je součástí balíku JavaGD.jar.

Příklad syntaxe:

```
.setenv('JAVAGD_CLASS_NAME'=tridaJava')
```

plotrix

Za vývojem knihovny plotrix [15] stojí Jim Lemon a další. Jedná se o knihovnu pro vykreslování specializovaných věci - od vykreslování tabulek, grafů, polygonů, aj. Důraz je zde věnován na jednoduché a přehledné používání funkcí.

Instalace knihovny se provádí příkazem `install.packages('plotrix')`, načítání knihovny příkazem `library(plotrix)`, smazání knihovny příkazem `remove.packages('plotrix')` s defaultní cestou ke knihovně. Možno přidat parametr lib, označující umístění knihovny na disku.

6 Vybrané analytické metody, implementace v R a JRA

Po nezbytném úvodu do R jazyka se identifikují analytické metody na základě scénářů z vlastního modelu. Tabulka četností dává uživateli přehled o zastoupení určitého znaku v určité kategorii. Modální kategorie popisuje největší kategorii. Medián a kvantily rozdělují proměnnou na určitý počet stejně velkých množin. Aritmetický průměr zjišťuje průměrnou hodnotu ze všech hodnot proměnné. Variační poměr, nominální rozptyl a entropie popisují kategorizovanou proměnnou z hlediska rozptýlení hodnot a určují míru homogenity této proměnné. Variační rozpětí značí, jaký je rozsah hodnot od minimální po maximální. Mezikvartilové rozpětí značí, jaký je rozsah hodnot od prvního po třetí kvartil. Rozptyl a směrodatná odchylka určují numerickou proměnnou z hlediska rozptýlení hodnot a určují míru homogenity této proměnné. Kontingenční tabulky popisují vzájemnou závislost dvou proměnných. Jsou to tedy metody pro základní analytické práce s datovým souborem. Dále se tyto metody teoreticky popíší, na základě literatury [7], [10] a implementují v R na základě literatury [11].

6.1 Rozdělení četností

Rozdělení četností kategoriální proměnné X značí, v jaké míře jsou jednotlivé kategorie zastoupeny. Výsledek se uvádí ve formě tabulky nebo grafu.

Tabulka četností

Pro kategoriální proměnnou X, která má x_i kategorií, kde $i = 1, \ldots, K$, se uvádí v tabulce **absolutní četnost**, tedy kolik respondentů n_i z celkového počtu n patří do příslušné kategorie. Dále se může uvádět **relativní četnost**, tedy jaký poměr respondentů patří do příslušné kategorie, vůči celkovému počtu respondentů, vzorec (1). Relativní četnost se po vynásobení hodnotou 100 může vyjádřit jako hodnota procentuální. Dále se může uvádět **kumulovaná četnost**, tedy součet aktuální relativní četnosti se součtem předchozích relativních četností, vzorec (2).

(1) $$p_i = \frac{n_i}{n}$$

(2) $$P_i = \sum_{j=1}^{j} p_i$$

Tyto výsledky se poté zanesou do tabulky. Této tabulce se poté říká tabulka četností.

Tabulka 1: Tabulka četností

X	absolutní	relativní	Kumulo-vaná
x_1	n_1	p_1	P_1
…	…	…	…
x_K	n_K	p_k	1
součet	n	1	-

U spojitých proměnných je potřeba nejprve hodnoty rozdělit do příslušných kategorií a poté udělat rozdělení četností. K tomu slouží **Sturgessovo pravidlo** (3), říkající na kolik intervalů je potřeba proměnnou rozdělit. Šíře jednotlivých intervalů se poté spočítá jako **variační rozpětí** (4) děleno počtem intervalů. Další postup je obdobný jako pro kategoriální proměnnou.

(3) $$k = 1 + 3{,}3 \log n$$

(4) $$R = x_{MAX} - x_{MIN}$$

Tabulka četností v R jazyce

K popisu rozdělení absolutních četností v R jazyce lze použít funkce `table(vector)`. Tato funkce vytvoří tabulku s příslušnými kategoriemi a počty hodnot v nich.

Např: Rozdělení absolutních četností vektoru obsahující muže, ženy, děti

```
> table(c('muz', 'zena', 'dite', 'muz'))
dite  muz  zena
   1    2    1
```

K popisu rozdělení relativních četností lze použít funkce `prop.table(table)`. Parametr této funkce není ale už vektor, nýbrž tabulka. Výsledkem této funkce jsou poté relativní četnosti v jednotlivých sloupcích tabulky.

Např: Rozdělení relativních četností vektoru obsahující muže, ženy, děti

```
> prop.table(table(c('muz', 'zena', 'dite', 'muz')))
dite  muz  zena
0.25  0.5  0.25
```

K popisu kumulativních četností lze použít funkci `cumsum(object)`. Parametrem funkce může být numerický i komplexní objekt. V tomto případě bude parametrem tabulka relativních nebo absolutních četností. Výsledkem budou poté kumulativní četnosti z dané tabulky.

Např: Rozdělení kumulativních četností vektoru obsahující muže, ženy, děti

```
> absolute <- table(c('muz', 'zena', 'dite', 'muz'))
> cumsum(prop.table(absolute))
dite  muz  zena
0.25  0.75   1
```

Pro numerické proměnné existuje funkce `hist()`. Tato funkce automaticky rozdělí numerickou proměnnou do intervalů, (hustotu intervalů lze měnit parametrem breaks) vypočte četnosti a vykreslí histogram (vykreslení lze vypnout parametrem plot=F). Výsledek funkce poté obsahuje další informace. Pod proměnnou $count je počet absolutních četností. Proměnná $breaks obsahuje hodnoty, které rozdělily numerickou proměnnou na intervaly. Proměnná $mids obsahuje střední hodnoty intervalů rozdělení.

Např: Histogram numerického vektoru a zjištění absolutní četnosti v intervalech

```
> h <- hist(c(1,7,5,9,10,6,3,6))
> h$counts
[1] 1 1 3 1 2
```

6.2 Popisné charakteristiky

Z hlediska popisných charakteristik se zkoumají **míra polohy** a **míra variability.**

Míra polohy

Pro kategoriální proměnné se určuje kategorie s největší četností a označuje se jako **modální kategorie.**

U spojitých proměnných se zjišťuje **medián** a kvantily. Kvantily se zjišťují většinou 25% a 75%. Poté se mluví o 1. (dolním) kvartilu a 3. (horním) kvartilu. 2. kvartilem se označuje sám medián. Po rozdělení do kategorií se dále zjišťuje **mediánová kategorie**, tj. kategorie, pro kterou kumulovaná četnost je > 0,5 přičemž kategorie předtím, modální kategorie, byla < 0,5. U spojitých proměnných se zjišťuje **aritmetický průměr**, dle vzorce (5)

(5) $$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Míra variability

Pro kategoriální proměnné se určuje **variační poměr (v)**, který se spočítá jako 1 - relativní četnost modální kategorie. Další funkcí je **nominální rozptyl**, tj. *„relativní počet všech dvojic, které nejsou ve stejné kategorii*“ [8]. Je dán vzorcem (6). Určuje se také **entropie**, která značí míru rozptýlení, vzorec (7).

(6) $$nomvar = 1 - \sum_{i=1}^{K} p_i^2$$

(7) $$H = -\sum_{i=1}^{K} p_i \ln p_i$$

U spojitých proměnných se zjišťují **variační** (4) **a mezikvartilové rozpětí (IRQ)**, IRQ = $x_{0,75} - x_{0,25}$. Dále se zjišťuje **rozptyl (s^2) a směrodatná odchylka (s)**. To zde určuje míru, jak hodně se liší hodnoty od průměru. Vzorec (8) a (9).

(8) $$s^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^2$$

(9) $$s = \sqrt{s^2}$$

Modální kategorie v R

Nejprve se spočítá jaká kategorie je nejčetnější pomocí funkce `max(freqTable)`, kde vstupním parametrem je patřičná tabulka četností. Po zjištění nejpočetnější kategorie se pomocí podmínky porovnání zjistí jméno této kategorie.

Např: Modální kategorie vektoru obsahující ženy, muže, děti

```
> relative <- prop.table(table(c('muz', 'zena', 'dite', 'muz')))
> mo <- relative[relative == max(relative)]
muz
0.5
```

Medián, kvantily v R

Pro zjištění kvantilů existuje funkce `quantile(x)`. Vstupním parametrem je zde numerický vektor x. Výsledkem funkce jsou hodnoty 0%, 25%, 50%, 75%, 100% kvantilu. Medián zde představuje hodnotu 50% kvantilu. Také ho lze spočítat pomocí funkce `median()`.

Aritmetický průměr v R

Pro výpočet aritmetického průměru slouží funkce `mean(x)`. Parametrem x je zde numerický nebo logický vektor. Dále může být vstupem i vektor s hodnotami date.

Variační poměr v R

Pro variační poměr neexistuje integrovaná funkce. Je nutné spočítat relativní hodnotu modální kategorie. Poté se tato hodnota odečte od hodnoty 1.

Nominální rozptyl v R

Pro určení nominálního rozptylu neexistuje integrovaná funkce. Nejprve se proto pomocí funkce `prop.table()` spočítají relativní četnosti. Poté se pomocí funkce `sum()` sečtou druhé mocniny relativních četností a výsledek se odečte od jedné.

Např: Nominální rozptyl vektoru obsahující ženy, muže, děti

```
> relative <- prop.table(table(c('muz', 'zena', 'dite', 'muz')))
> mo <- 1-sum(relative^2)
[1] 0.625
```

Entropie v R

Pro výpočet entropie neexistuje integrovaná funkce. Nejprve se tedy spočítá pomocí funkce `prop.table()` relativní četnost. Poté se pomocí funkce přirozeného logaritmu `log()` a pomocí funkce `sum()` spočítá entropie.

```
> relative <- prop.table(table(c('muz', 'zena', 'dite', 'muz')))
> -sum(relative*log(relative))
[1] 1.039721
```

Variační a mezikvartilové rozpětí v R

Pro výpočet variačního rozpětí lze využít funkce `range()`, která vrací hodnotu minimální a maximální. Nebo využít funkce `max()`, `min()` a patřičně je mezi sebou odečíst.

```
> max(data)-min(data)
> range(data)[2]-range(data)[1]
```

Pro výpočet mezikvartilového rozpětí lze použít funkce `quantile()`. Spočítat si 1. a 3. kvartil a odečíst je.

```
> quantile(data)[4]-quantile(data)[2]
```

Rozptyl a směrodatná odchylka v R

Pro určení rozptylu existuje integrovaná funkce `var(x)`. Vstupním parametrem této funkce je numerický vektor x nebo matice, nebo datová tabulka. Výsledkem je poté určující rozptyl.

Pro určení směrodatné odchylky existují dva způsoby. První je odmocnina z rozptylu, viz výše. Druhá možnost je použití integrované funkce `sd(x)`. Vstupem této funkce je numerický vektor x.

6.3 Kontingenční tabulky

Kontingenční tabulky znázorňují dvourozměrné rozdělení četností. Neboli znázorňují rozdělení četností dvourozměrného vektoru X = (Y, Z). Y značí první kategoriální proměnnou, nabývající hodnot *1 … k*. Z značí druhou kategoriální proměnnou, nabývající hodnot *1 … s.* Řádkové a sloupcové součty se zde nazývají **marginální proměnné**.

Např. Mějme výběr o rozsahu n, proměnné Y a Z nabývajících hodnot 1, 2, 3.

Tabulka 2: Kontingenční tabulka

Y	Z			součet
	1	2	3	
1	n_{11}	n_{11}	n_{11}	$n_{1.}$
2	n_{11}	n_{11}	n_{11}	$n_{2.}$
3	n_{11}	n_{11}	n_{11}	$n_{3.}$
součet	$n_{.1}$	$n_{.2}$	$n_{.3}$	n

Z této tabulky se dále dá vypočítat, zda jsou znázorněné proměnné závislé či nezávislé. K tomu se využívá např. **Chí-kvadrát test**. Vzorec číslo (10) se porovnává s podmínkou $\chi^2 \geq \chi^2_{(k-1)(s-1)}(\alpha)$. Je-li splněna, jedná se nezávislé proměnné a naopak.

(10)
$$\chi^2 = \sum_{i=1}^{k} \sum_{j=1}^{s} \frac{\left(n_{ij} \frac{n_{i.} n_{.j}}{n}\right)}{\frac{n_{i.} n_{.j}}{n}}$$

Kontingenční tabulky v R

Jako v případě rozdělení absolutních četností lze použít funkci `table()`. V tomto případě se jako vstupní parametr vloží 2 a více proměnných. V této práci se budou předávat pouze 2 proměnné. Výsledkem této funkce bude poté kontingenční tabulka, kde hodnoty budou označovat počty v každé kombinaci kategorií.

Např: Kontingenční tabulka vektoru obsahující muže, ženy, děti a vektoru obsahující počty sourozenců

```
> table(c('muz', 'zena', 'dite', 'muz') c(0,2,2,1))
      0  1  2
 dite  0  0  1
  muz  1  1  0
 zena  0  0  1
```

Pro zjištění závislosti či nezávislosti proměnných lze z této kontingenční tabulky provést **Chí-kvadrát test**. K tomu slouží funkce `chisq.test(table)`. Výsledkem této funkce je hodnota funkce chí-kvadrátu pro danou kontingenční tabulku. Dále počet stupňů volnosti (df) a hodnota pravděpodobnosti (p-value). Hodnota vypočítané pravděpodobnosti se poté porovná vůči hodnotě 0,05. Pokud bude p-value <= 0,05 jedná se o závislé proměnné (tedy výsledek jedné ovlivňuje výsledek druhé). Pokud bude p-value > 0.05 jedná se o nezávislé proměnné (tedy výsledek jedné neovlivňuje výsledek druhé).

Např: Výsledek použití funkce chisq.test() na předem známou kontingenční tabulku

```
> chisq.test(cont)
Pearson's Chi-squared test
data:  cont
X-squared = 12.2285, df = 3, p-value = 0.00664
```

7 Vybrané grafy knihoven base a plotrix

Výsledky se v R jazyce mohou interpretovat tabulkou nebo grafem. Zde se popíše interpretace pomocí grafů.

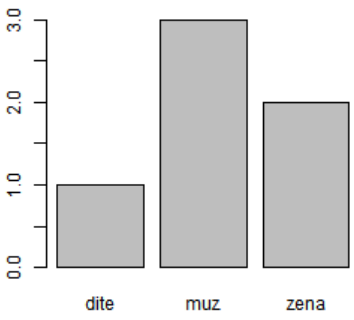
7.1 Sloupcový graf (Bar plot), Histogram

Jedná se o jednu z variant pro reprezentaci rozdělení četností. Graf je složen z jednotlivých sloupců. Každý sloupec má svou výšku úměrnou reprezentované četnosti. Tedy kategorie s největší četností má sloupec největší a naopak.

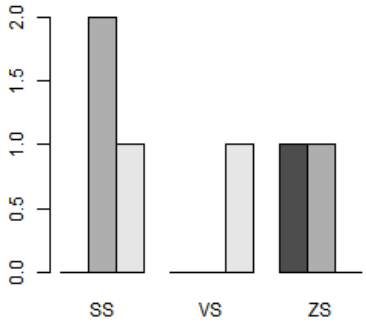
V R jazyce k tvorbě sloupcového grafu slouží funkce `barplot()` z base knihovny. Vstupními parametry funkce jsou parametry:

- height – reprezentuje vstupní vektor dat. Tedy vektor hodnot, nebo matici hodnot
- main – reprezentuje titulek, název grafu. Ten se vypíše tučnými písmeny, uprostřed nahoře
- xlab – reprezentuje vlastní název, popř. popis osy x
- ylab – reprezentuje vlastní název, popř. popis osy y
- beside – pokud je parametr na hodnotě TRUE a vstupním parametrem je kontingenční tabulka, jsou sloupce kresleny po skupinách daných touto tabulkou. Pokud je na parametr na hodnotě FALSE, sloupce jsou kresleny ve formě fronty, uvnitř sebe
- col – vektor reprezentující barvu sloupců. Pokud má vektor barev méně hodnot než je počet sloupců, použijí se hodnoty opakovaně

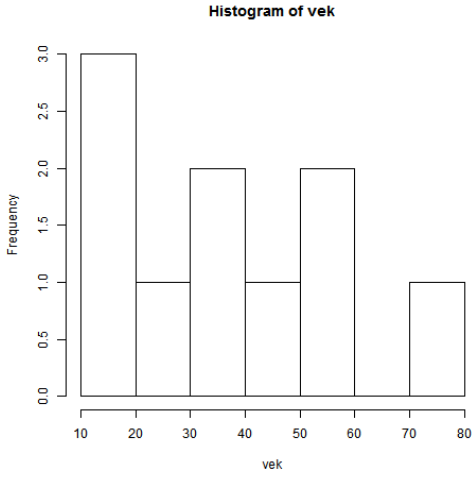
Speciální variantou reprezentace rozložení četností, pro numerickou proměnnou je histogram. V R jazyce pro něj slouží funkce `hist(x)`, kde x je numerická proměnná.



Obrázek 6: Sloupcový graf rozdělení četností



Obrázek 7: Sloupcový graf kontingenční tabulky



Obrázek 8: Histogram věkového rozdělení

7.2 Kruhový graf (Pie plot)

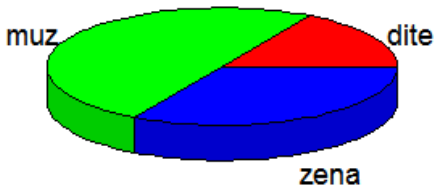
Jedná se o další variantu reprezentace rozdělení četností. Graf je složen z jednotlivých kruhových výsečí. Velikost každé kruhové výseče je závislá na četnosti dané kategorie. Celkově vyváří kruh.

V R jazyce souží k vykreslení funkce `pie()` z knihovny base. Tato funkce vykreslí jednoduchý 2D graf.

V této práci se bude používat externí knihovna plotrix a její funkce pie3D(). Tato funkce kreslí 3D kruhové grafy. Vstupními parametry funkce jsou:

- x – vektor vstupních dat
- labels – vektor obsahující názvy jednotlivých kruhových výsečí

- main – reprezentuje titulek, název grafu. Ten se vypíše tučnými písmeny, uprostřed nahoře
- col – vektor reprezentující barvu sloupců. Pokud má vektor barev méně hodnot než je počet sloupců, použijí se hodnoty opakovaně
- height – reprezentuje šířku grafu
- start – reprezentuje úhel, ze kterého se začínají vykreslovat výseče
- theta – reprezentuje úhel pohledu
- explode – reprezentuje velikost mezery mezi jednotlivými výsečemi
- labelcex – reprezentuje faktor zvětšení písma vektoru labels



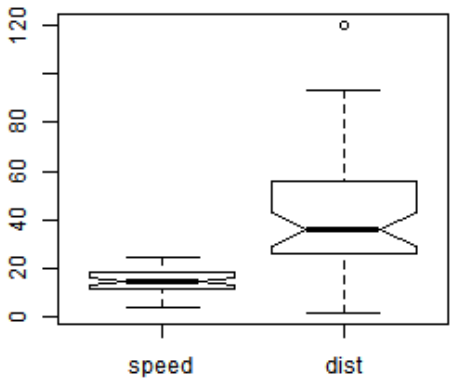
Obrázek 9: Kruhový graf rozdělení četností

7.3 Krabicový graf (Box plot)

Jedná se o variantu reprezentace numerických dat. Graf vychází z kvartilů dat. Dále znázorňuje extrémní hodnoty, míru rozptylu a šikmosti. Dále může zobrazovat aritmetický průměr a variační rozptyl.

V R jazyce je implementován pomocí funkce `boxplot()` z knihovny base. Vstupními parametry funkce jsou:

- x – vstupní vektor s daty.
- names – v případě více proměnných, indikují názvy jednotlivých „krabic“
- notch – pokud je nastaven na hodnotě TRUE, indikuje tzv. „zářezy“ kolem mediánu



Obrázek 10: Krabicový graf

8 Návrh a realizace vlastního řešení

Hlavní částí práce je vytvoření vlastního softwaru pro analýzu datového souboru. Návrh tohoto softwaru vychází z vlastního modelu, popsaného v první části práce. Dále z identifikovaných a popsaných analytických metod. Software nese označení JRA (Java-R Analysis).

Nejprve bude JRA představen z hlediska grafického rozhraní, včetně ukázek a možností použití. Poté se software rozebere z hlediska vnitřní struktury (tj. z hlediska programátorského). Pro ukázky jsou použity datové soubory z R [12]. Vlastní testování softwaru probíhá stažením veřejně dostupného datového souboru, včetně výsledků analýzy. Tento datový soubor se otevře v aplikaci JRA, provede se analýza a výsledky se porovnají s originálními.

8.1 Grafické rozhraní JRA, ukázky a možnosti použití

Software se skládá ze dvou hlavních oken. Každé okno obsahuje nabídku menu. Pro různá okna obsahuje menu různé položky.

První okno slouží pro načtení datového souboru. Skládá se tlačítka pro vybrání souboru a informačního textboxu, obsahujícího cestu k němu. Po načtení souboru se v okně zobrazí obsah datového souboru. Dále je v tomto okně možnost zvolit chybějící hodnoty (dvojklik na záhlaví sloupce). Pro vybrání určitých proměnných k analýze dále slouží tlačítko detail. V menu jsou nejvýraznějšími prvky možnosti pro změnu jazyka.

Druhé okno slouží jako okno pro detailní zobrazení vybraných proměnných. Hlavní částí tohoto okna je panel se záložkami. Jako první záložka jsou zobrazeny vybrané proměnné. Dále se zde provádí veškeré analytické operace s vybranými proměnnými. K tomu slouží patřičná tlačítka v menu. Pod těmito tlačítky se skrývají analytické metody, popsané v první části této práce. Po zvolení okruhu analýzy se zobrazí dialog s upřesňujícími možnostmi tohoto okruhu analýzy. Výstupy analýzy se zobrazí v nové záložce, pod jménem proměnné, pro kterou je analýza počítána. Grafická reprezentace výstupů analýzy je možná uvnitř okna (plní funkci náhledu), nebo

možnost v novém okně, přičemž ve výchozím stavu je nastavena možnost V novém okně. Podrobnější informace obsaženy v uživatelské příručce, viz příloha 2.

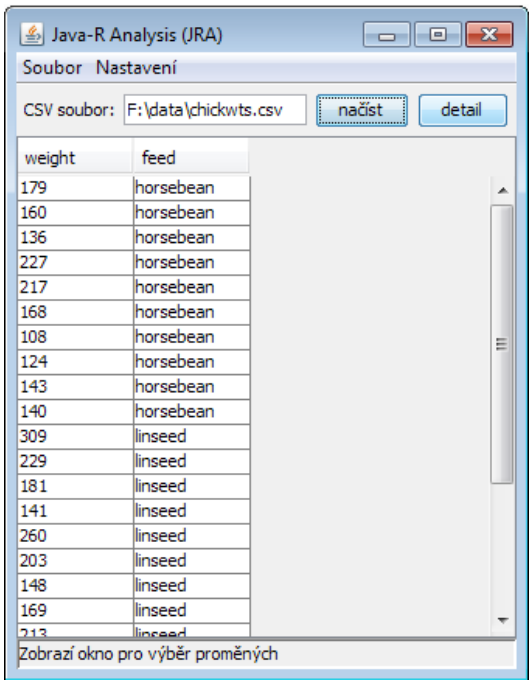
Ukázky a možnosti použití

Datový soubor chickwts

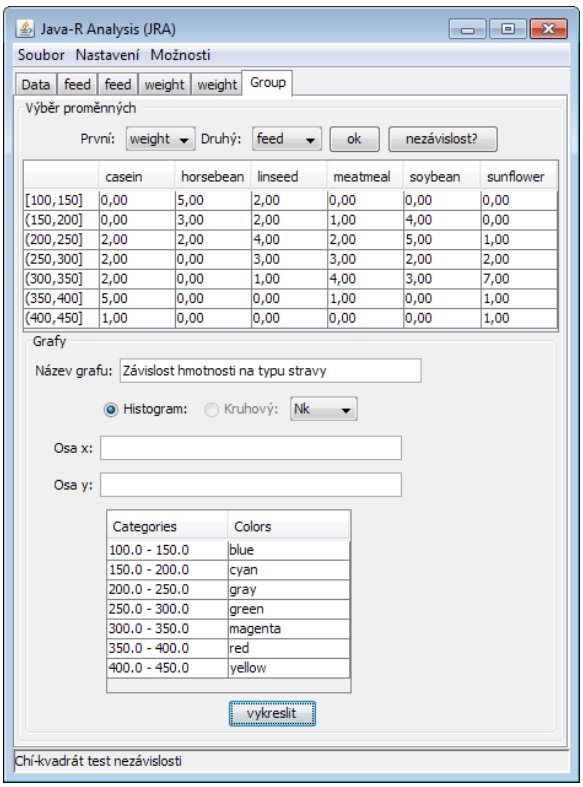
Jedná se o datový soubor popisující váhu kuřat a typ stravy kuřat. Obsahuje dvě proměnné. První, numerická, proměnná obsahuje hodnotu váhy kuřete. Druhá faktorová proměnná obsahuje typ stravy. Typ stravy může nabývat hodnot *horsebean, linseed, soybean, sunflower, meatmeal, casein.* Celkově obsahuje 71 záznamů. Datový soubor je součástí knihovny datasets, která je součástí knihovny base R jazyka.

U tohoto souboru se bude analyzovat:

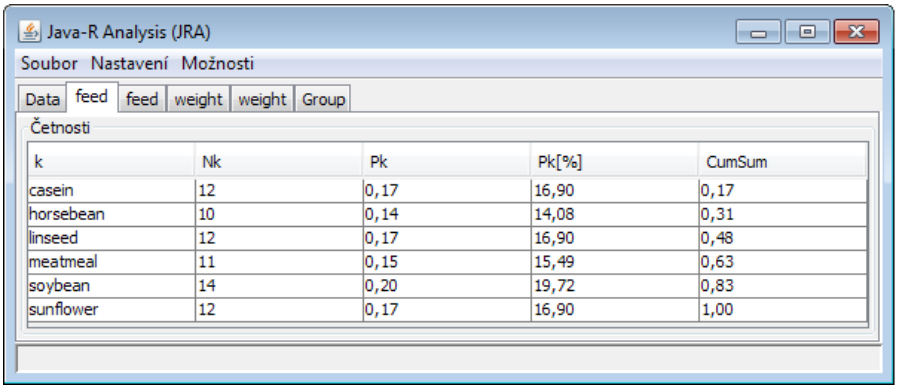
- tabulka četností pro typy stravy
- tabulka četností a vykreslení histogramu uvnitř aplikace pro váhu kuřat
- vzájemná závislost obou proměnných s grafickou reprezentací i výsledkem testu nezávislosti



Obrázek 11: Načtení datového souboru chickwts

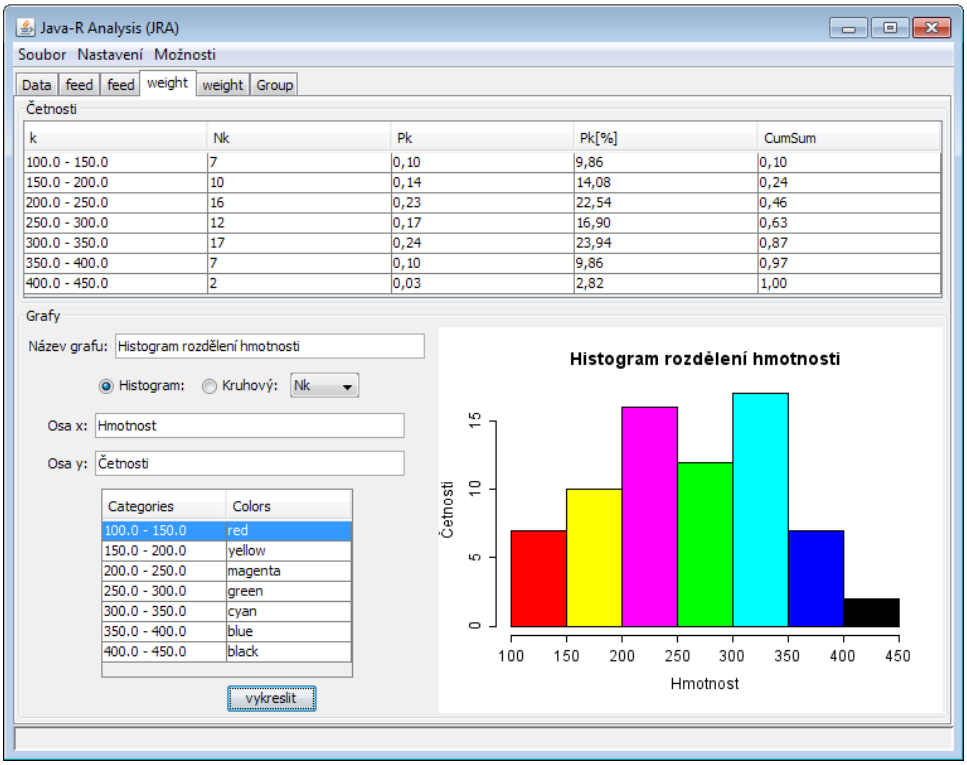


Obrázek 12: Kontingenční tabulka proměnných feed, weight souboru chickwts



k	Nk	Pk	Pk[%]	CumSum
casein	12	0,17	16,90	0,17
horsebean	10	0,14	14,08	0,31
linseed	12	0,17	16,90	0,48
meatmeal	11	0,15	15,49	0,63
soybean	14	0,20	19,72	0,83
sunflower	12	0,17	16,90	1,00

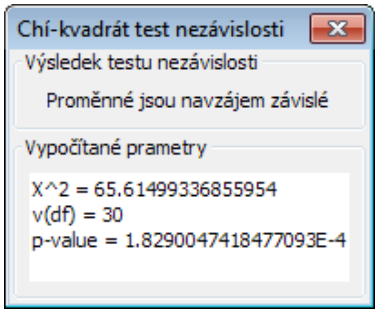
Obrázek 13 Tabulka četností proměnné feed souboru chickwts



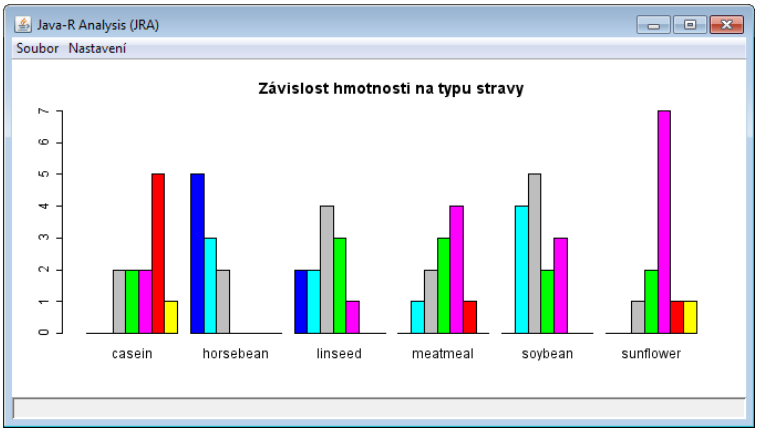
k	Nk	Pk	Pk[%]	CumSum
100.0 - 150.0	7	0,10	9,86	0,10
150.0 - 200.0	10	0,14	14,08	0,24
200.0 - 250.0	16	0,23	22,54	0,46
250.0 - 300.0	12	0,17	16,90	0,63
300.0 - 350.0	17	0,24	23,94	0,87
350.0 - 400.0	7	0,10	9,86	0,97
400.0 - 450.0	2	0,03	2,82	1,00

Categories	Colors
100.0 - 150.0	red
150.0 - 200.0	yellow
200.0 - 250.0	magenta
250.0 - 300.0	green
300.0 - 350.0	cyan
350.0 - 400.0	blue
400.0 - 450.0	black

Obrázek 14: Tabulka četností histogram proměnné weight souboru chickwts



Obrázek 15: Test nezávislosti proměnných feed, weight souboru chickwts



Obrázek 16: Graf závislosti proměnných feed, weight souboru chickwts

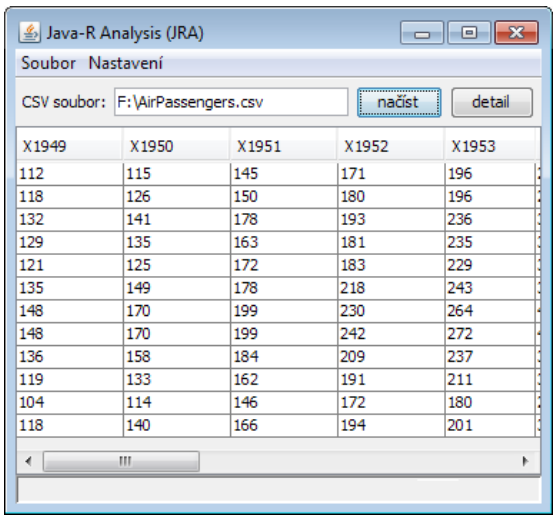
Závěry plynoucí z této analýzy jsou tedy, že typ stravy má vliv na váhu kuřat (závislost proměnných). Detaily jsou zobrazeny na grafu, viz Obrázek 16. Dále je možné vyčíst z histogramu, že váha se blíží normálnímu rozdělení, viz Obrázek 14.

Datový soubor AirPassengers

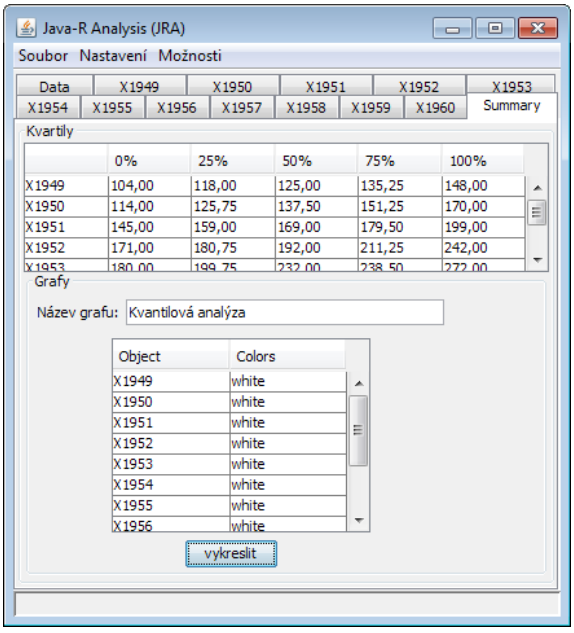
Jedná se o datový soubor popisující měsíční počet cestujících v letecké dopravě z let 1949 až 1960. Obsahuje 12 proměnných. Každá reprezentuje jeden rok. Každá proměnná (rok) obsahuje 12 hodnot, které určují počet přepravených cestujících od ledna do prosince. Datový soubor je součástí knihovny datasets, která je součástí knihovny base R jazyka.

U tohoto souboru se bude analyzovat:

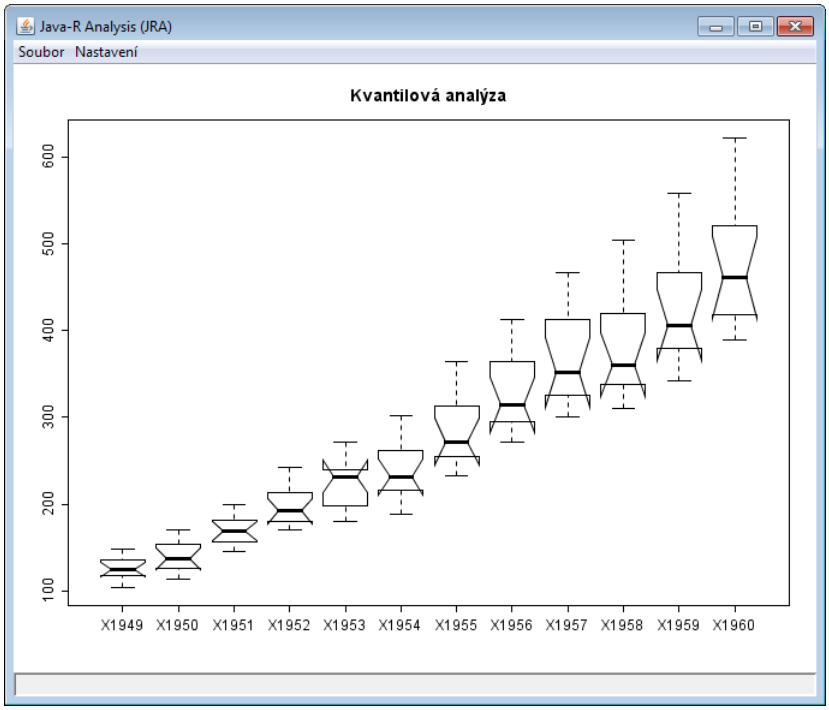
- tabulka kvantilů pro jednotlivé proměnné
- tabulka kvantilů pro celý soubor
- grafické znázornění jednotlivých proměnných datového souboru promocí krabicového grafu



Obrázek 17: Načtení datového souboru AirPassengers



Obrázek 18: Detail proměnných souboru AirPassengers



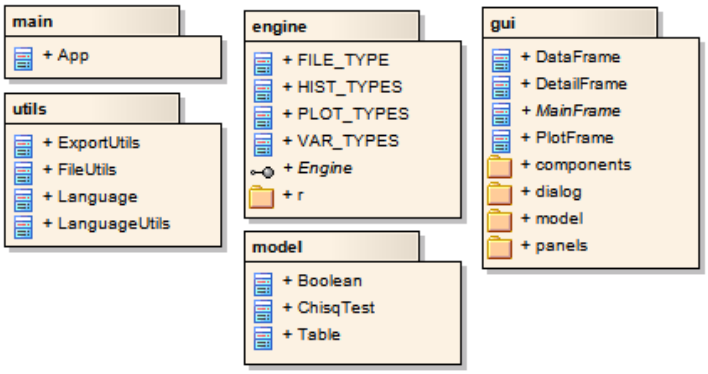
Obrázek 19: Krabicový graf pro proměnné souboru AirPassengers

Závěry plynoucí z této analýzy lze vyčíst z krabicového grafu pro proměnné, viz Obrázek 19. Například při porovnání hodnot z roku 1949 a 1960 lze jednoznačně říct, že hodnoty v roce 1960 mají daleko větší variační rozpětí než hodnoty roku 1949. Další informace lze vyčíst i z tabulky kvantilů všech proměnných, viz Obrázek 18.

8.2 Implementace JRA v Java

Popis implementace programu v jazyce Java z hlediska programátorského se bude skládat ze sekcí určujících strukturu balíčků v projektu. Začátek každé sekce bude obsahovat grafický souhrn v UML notaci. Pro zjednodušení tento souhrn obsahuje pouze třídy s metodami, u modelových tříd jsou zobrazeny i atributy. Tyto metody jsou poté rozepsány podrobně. Detailnější struktura tříd je obsažena v přiloženém CD v adresáři se zdrojovými kódy.

K implementaci JRA je použita Java SDK ve verzi 1.8.0_25 a R ve verzi 3.1.1. Celková struktura tříd je rozdělena do balíčků určující jejich použití. Z java objektů jsou zde použity výčtové typy (enum), třídy (class) i rozhraní (interface).

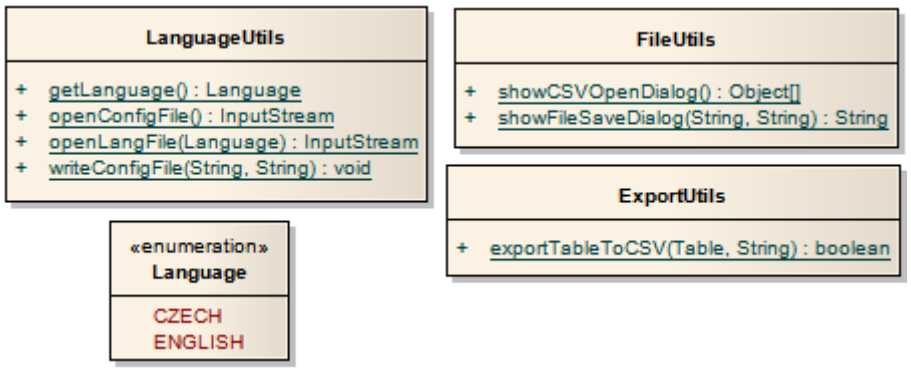


Obrázek 20: Struktura balíčků projektu

Základními balíčky na root (hlavní) úrovni, jsou balíčky:

- main – obsahuje třídu App. Tato třída slouží pouze pro spuštění aplikace
- utils – obsahuje pomocné třídy pro práci s ukládáním a načítáním souborů
- model – obsahuje modelové třídy. Jedná se o třídy s největší mírou znovupoužitelnosti
- engine – obsahuje třídy a rozhraní, sloužící k oddělení aplikační a prezentační logiky, neboli k oddělení technologie Java (prezentační logika) a technologie R (aplikační logika)
- gui – obsahuje třídy a rozhraní pro zobrazování výsledků a skládání grafického rozhraní

Balíček utils



Obrázek 21: Balíček utils

Language

Jedná se o výčtový typ určující jazykovou lokalizaci aplikace. Může nabývat hodnot CZECH, ENGLISH.

LanguageUtils

Třída slouží jako kontejner pro statické metody zabývající se prací s jazykovými a konfiguračními soubory.

`getLanguage()`

Načte z konfiguračního souboru záznam o jazykové lokalizaci a vrátí uživateli patřičnou hodnotu z výčtového typu Language.

`openConfigFile()`

Otevře konfigurační soubor a vrátí uživateli instanci třídy InputStream, bitový proud reprezentující data ze souboru.

`openLangFile()`

Otevře příslušný jazykový soubor a vrátí instanci třídy InputStream.

`writeConfigFile()`

Přepíše záznam o jazykové lokalizaci na hodnotu danou uživatelem.

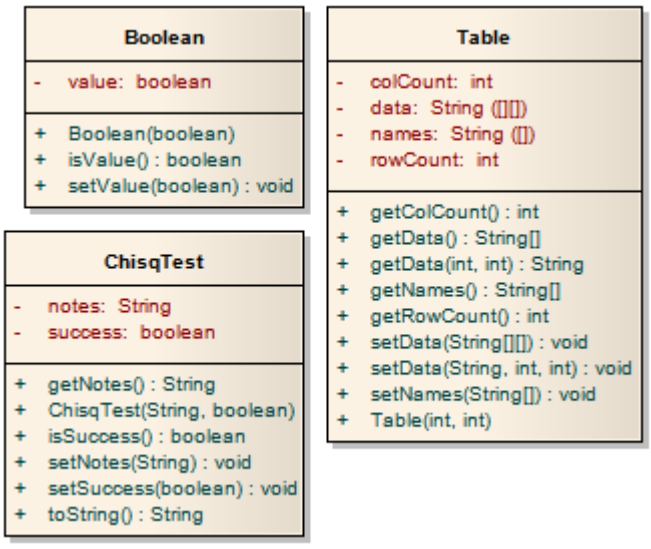
ExportUtils

Třída slouží jako kontejner pro statické metody zabývající se exportem dat do souborů.

`exportTableToCSV()`

Převádí data z instance třídy Table do souboru CSV odděleného středníkem.

Balíček model



Obrázek 22: Balíček model

Boolean

Jedná se o objektový typ zapouzdřující primitivní datový typ *boolean*. Díky tomu lze využít objektových vlastností pro primitivní datový typ.

Table

Jedná se o třídu reprezentující obecně jakoukoli tabulkovou strukturu pro jazyk Java. Třída obsahuje atribut data, dvourozměrné pole typu String pro ukládání dat v řetězcové formě. Dále obsahuje atribut names, jednorozměrné pole typu String pro ukládání názvů sloupců.

`Table()`

Konstruktor vytvářející instanci počtem řádků a počtem sloupců.

`getColCount()`

Vrací počet sloupců instance třídy Table.

`getRowCount()`

Vrací počet řádků instance třídy Table.

`getData()`

Přetížená metoda vrací buď celkový soubor dat, nebo hodnotu určité buňky.

`setData()`

Přetížená metoda nastavující soubor hodnot do tabulky, nebo určitou hodnotu na určité místo do tabulky.

`getNames()`

Vrácení názvů sloupců.

`setNames()`

Nastavení názvů sloupců.

ChisqTest

Jedná se o třídu reprezentující výsledky Pearsonova Chí kvadrát testu.

`ChisqTest()`

Konstruktor vytvářející instanci danou parametry reprezentující popis a hodnoty testu a parametru reprezentující úspěch či neúspěch (nezávislost či závislost).

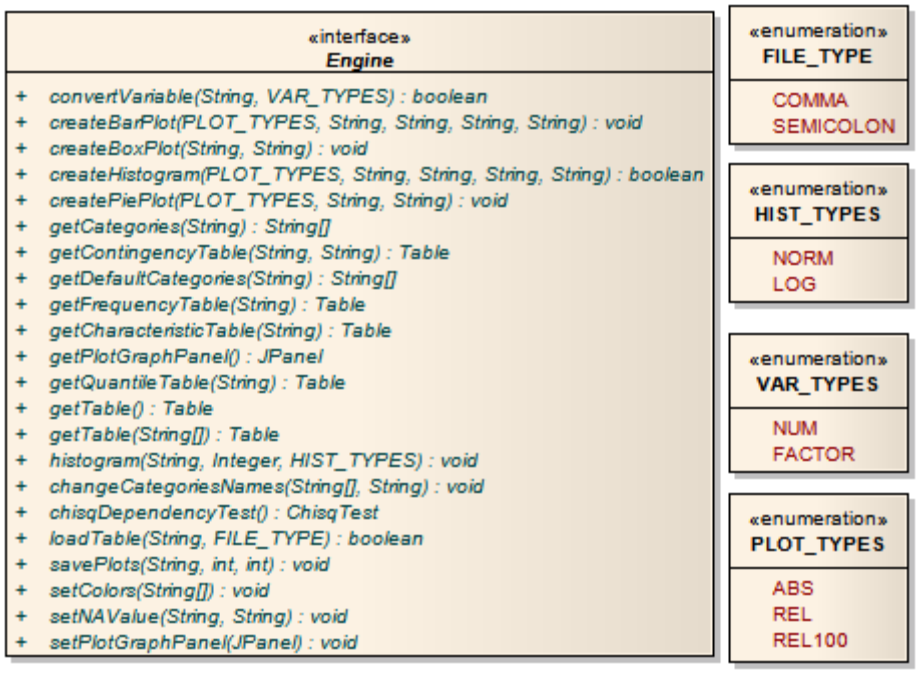
`getNotes()`

Vrací informace o testu.

`isSuccess()`

Vrací informaci o úspěšnosti testu.

Balíček engine



Obrázek 23: Balíček engine

Základní obsah balíčku engine

Zde se popisují třídy a jejich metody nacházející se v kořenovém balíčku (engine).

FILE_TYPE

Výčtový typ určující typ oddělovače načítaného datového souboru.

HIST_TYPE

Výčtový typ určující typ histogramu.

PLOT_TYPES

Výčtový typ určující typ vykreslovaného grafu.

VAR_TYPES

Výčtový typ určující typ proměnné (sloupce tabulky).

Engine

Rozhraní obsahující veřejné metody pro hlavní práci aplikace. Toto rozhraní odděluje třídy a implementaci aplikační logiky a prezentační logiky (gui).

`convertVariable()`

Převod proměnné na faktorovou nebo numerickou. Výsledek je logická hodnota určující úspěch či neúspěch převodu.

`createBarPlot()`

Vykreslení sloupcového grafu pro kontingenční tabulku nebo histogramu pro faktorovou proměnnou. Přepínačem PLOT_TYPE se určí, zda vykreslit pro absolutní či relativní četnost.

`createBoxPlot()`

Vykreslení krabicového grafu.

`createHistogram()`

Vykreslení histogramu pro numerickou proměnnou. Funkce přepínače PLOT_TYPE, viz createBarPlot(). Výsledek je logická hodnota určující úspěch či neúspěch vykreslení histogramu.

`createPiePlot()`

Vykreslení kruhového grafu. Funkce přepínače PLOT_TYPE, viz createBarPlot().

`getCategories()`

Zjištění jmen kategorií pro danou proměnnou, po provedení kategorizace.

`getDefaultCategories()`

Zjištění jmen kategorií pro danou proměnnou, před provedením kategorizace.

`getContingencyTable()`

Vytvoření kontingenční tabulky ze dvou zadaných proměnných.

`getFrequencyTable()`

Vytvoření frekvenční tabulky pro zadanou proměnnou. Vrací instanci Table.

`getCharacteristicTable()`

Vytvoření tabulky charakteristik pro zadanou proměnnou. Vrací instanci Table.

`getPlotGraphPanel()`

Metoda vracející instanci plátna, na které se budou vykreslovat grafy.

`getQuantileTable()`

Vytvoření tabulky kvantilů pro zadanou proměnnou. Vrací instanci Table.

`getTable()`

Zobrazení tabulky se všemi daty. Pokud je zadán parametr určující název sloupce, zobrazí se data pouze z tohoto sloupce.

`histogram()`

Vytvoření histogramu pro danou proměnnou. Přepínač HIST_TYPE nastaví výpočet na logaritmický nebo normální. Metoda nevykresluje histogram.

`changeCategoriesNames()`

Nastavení jmen kategorií zadané proměnné.

`chisqDependencyTest()`

Vypočtení testu nezávislosti. Vrací instanci třídy ChisqTest.

`loadTable()`

Načtení a vytvoření tabulky ze souboru se všemi daty. Přepínač FILE_TYPE značí, jaké znaky jsou použity jako oddělovače.

`savePlots()`

Uložení grafu na dané umístění s danými rozměry.

`setColors()`

Nastavení barev pro vykreslování grafů.

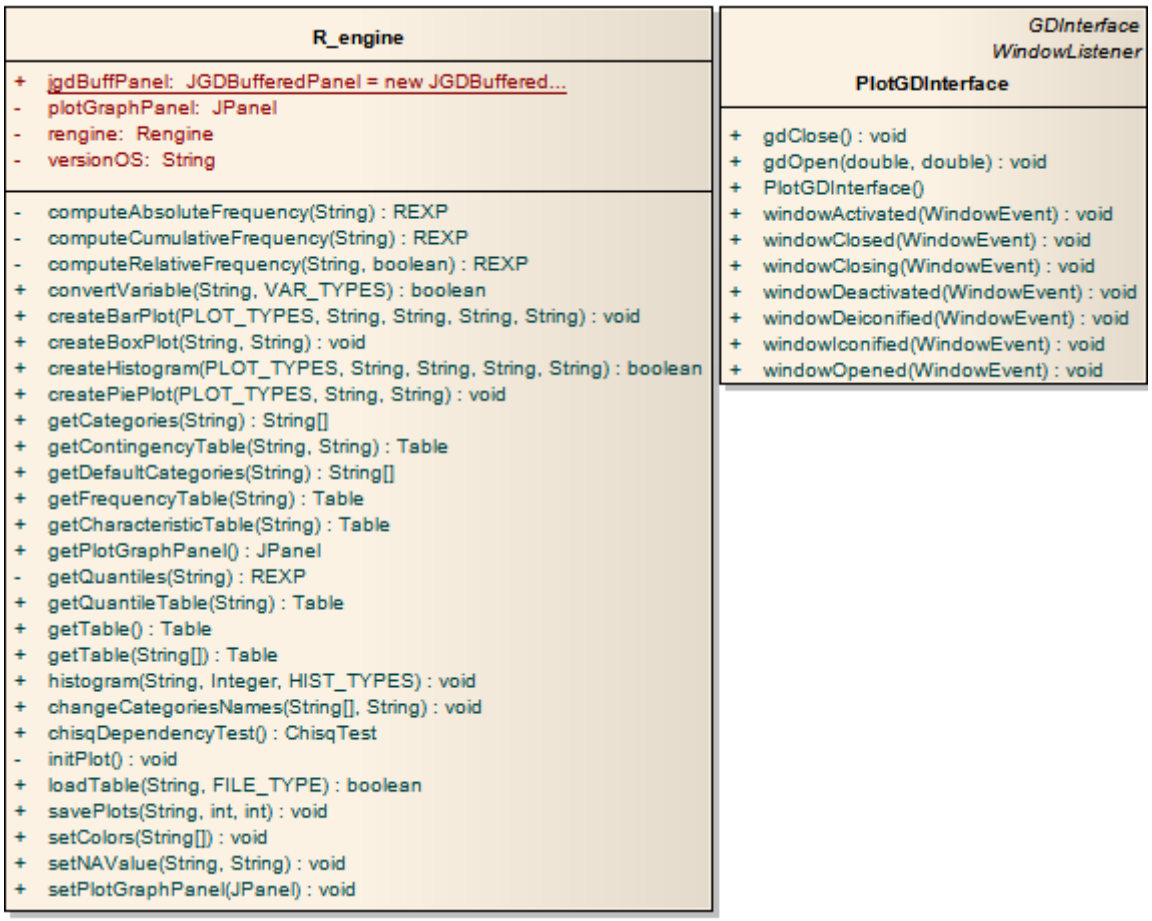
`setNAValue()`

Nastavení chybějících hodnot v určitém sloupci.

`setPlotGraphPanel()`

Nastavení panelu pro vykreslování grafů.

Obsah podbalíčku r



Obrázek 24: Podbalíček r

R_engine

Jedná se o implementaci rozhraní Engine. Tato implementace volá z jazyka Java kód a příkazy jazyka R. K tomu slouží třída Rengine z knihovny JRI. Instancí této třídy je zde atribut engine. Dále třída obsahuje jeden statický veřejný atribut jgdBuffPanel, ten reprezentuje plátno, na které se vykreslují grafické prvky jazyka R. Tento atribut je dále sdílen s atributem plotGraphPanel, který předepisuje rozhraní Engine.

`R_engine()`

V konstruktoru se vytváří instance třídy Rengine s parametrem *--vanilla*. Tento parametr naznačuje, že R prostředí se má spustit v čistém režimu, bez předchozích uložených věcí. Také po ukončení se prostředí neukládá. Třída Rengine disponuje metodou **REXP eval(String)**. Tato metoda spouští příkaz z parametru v prostředí R a vrací objekt REXP. Objekt REXP dále disponuje

metodami pro převod R objektu na objekt třídy Java. Poté se, pomocí výše zmíněné funkce, načtou patřičné knihovny do R prostředí, **library(javaGD)** a **library(plotrix)**.

`computeAbsoluteFrequency()`

Pomocná metoda pro výpočet absolutní četnosti v objektu typu REXP. Samotný výpočet se rozděluje na dva typy, které jsou zvoleny na základě typu proměnné. Pro faktorovou proměnnou je použita funkce **table(data$colName)**. Pro numerickou proměnnou jsou použita data z vypočítaného histogramu. K tomu je využita funkce **hiscolName$counts**. Části colName jsou dodány uživatelem při volání funkce a obsahují názvy konkrétních proměnných v datové tabulce.

`computeCumulativeFrequency()`

Pomocná metoda pro výpočet kumulativní četnosti v objektu typu REXP. Výpočet se dělí na dva typy odpovídající typu proměnné. Pro faktorovou proměnnou použita funkce **cumsum(prop.table(table(data$colName)))**. Pro numerickou proměnnou použita funkce **cumsum(hiscolName$counts /sum(hiscolName$counts))**.

`computeRelativeFrequency()`

Pomocná metoda pro výpočet relativní četnosti v objektu typu REXP. Výpočet se dělí na dva typy odpovídající typu proměnné. Pro faktorovou proměnnou je použita funkce **prop.table(table(data$colName))**. Pro numerickou proměnnou funkce **hiscolName$counts/sum(hiscolName$counts)**.

`getQuantiles()`

Pomocná metoda pro výpočet kvantilů 0, 25, 50, 75 a 100% pro numerické proměnné. Je zde využita funkce **quantile(data$colName, na.rm=TRUE)**. Parametr na.rm=TRUE značí, že se chybějící hodnoty vynechávají.

`initPlot()`

Pomocná metoda pro nastavení proměnné prostředí a nastavení a inicializaci JavaGD třídy. Nastavení prostředí **.setenv <- if (exists('Sys.setenv')) Sys.setenv else Sys.putenv**. Nastavení JavaGD třídy **.setenv('JAVAGD_CLASS_NAME'='engine/r/PlotGDInterface')**. Ve finále se JavaGD třída inicializuje **JavaGD(width=400, height=300)**.

`convertVariable()`

Při převodu na faktorovou proměnnou je využita funkce **as.character(variable)**. Při převodu na numerickou proměnnou se odstraní případné volné mezery funkcí **gsub('\\s', '', variable)** a proměnná se převede **as.numeric(variable)**.

`createBarPlot()`

Pro vykreslení sloupového grafu se využívají funkce **table()** pro absolutní četnost, **prop.table(table())** pro relativní četnost. Pro samotný graf poté funkce **barplot(x, col=plotColors, xlab=xLabel, ylab=yLabel, main=title, beside=TRUE)**, kde col znamená barvy z proměnné plotColors, xlab a ylab popsiky na ose x a y dodané uživatelem z parametru.

`createBoxPlot()`

Pro vykreslení krabicového grafu se využívá funkce **boxplot(cols,... , notch=TRUE)**. Cols jsou jména numerických proměnných obsažených v grafu. Sumární informace jsou do grafu vepsány funkcí **mtext(), summary()** a for cyklu.

`createHistogram()`

Pro vykreslení histogramu s absolutními četnostmi je využita funkce **plot(hiscolName, ...)**. Pro vykreslení histogramu s relativními četnostmi je potřeba nejprve relativní četnosti spočítat a uložit do pomocné proměnné **hisTmp$counts <- hiscolName$counts/sum(hiscolName$counts)**. O zbytek se postará opět funkce **plot()**.

`createPiePlot()`

K vykreslení použita funkce **pie3D()**. Typ vykreslení (absolutní nebo relativní četnosti) popsány ve funkci createBarPlot().

`getDefaultCategories()`

Zjištění jmen se provádí pomocí funkce **levels()**. Zjištění kategorií pomocí funkce **factor()**. Tedy celkově **levels(factor(data$colName))**.

`getCategories()`

Obdoba getDefaultCategories(). Navíc pro numerickou proměnnou se názvy kategorií skládají z hodnot **hiscolName$breaks**.

`getContingencyTable()`

Vytvoření kontingenční tabulky absolutních četností. Využívá funkci **table(first, second)**. Tato funkce vrátí objekt REXP, který se dále pomocí java funkce asMatrix() převede na dvourozměrné pole. Metoda vrací poté instanci třídy Table s daty z dvourozměrného pole.

`getFrequencyTable()`

Vytvoření frekvenční tabulky. Data se získají pomocí metod getCategories(), computeAbsoluteFrequency(), computeRelativeFrequency(), computeCumulativeFrequency(). Výsledky metod se poté nastaví do atributu data instance třídy Table, kterou vrátí uživateli.

`getCharacteristicTable()`

Charakteristiky pro zadanou proměnnou se rozdělují na část společnou pro všechny typy a část pouze pro numerické proměnné. Z výsledků funkcí se vytvoří instance třídy Table. Ta se poté vrátí uživateli.

`getQuantileTable()`

Vrátí instanci třídy Table obsahující hodnoty vypočítané pomocnou funkcí getQuantiles().

`getTable()`

Vrátí instanci třídy Table. Ta bude obsahovat data vypsaná z R datové proměnné **data**. Metoda s parametrem columnNames bude obsahovat data vypsaná z R proměnné **data$columName.**

`histogram()`

Vytvoří histogram pro zadanou numerickou proměnnou, dále vytvoří počet intervalů podle vstupního parametru breaks. Pokud je parametr breaks null, vytvoří se počet intervalů podle Sturgessova pravidla. Histogram vytvoří proměnnou pro uložení informací **hiscolName <- hist(data$colName, breaks=breaksStr, plot=F)**.

`changeCategoriesNames()`

Pomocí funkce **levels()** změní názvy kategorií pro zadanou proměnnou.

`chisqDependencyTest()`

Funkce pro výpočet testu nezávislosti. Výsledek testu uložen do instance třídy ChisqTest a instance vrácena uživateli.

`loadTable()`

Načítá data do datové proměnné pomocí funkce - **data <- read.table('filePath', header=T, sep=';')**. Podle přepínače FILE_TYPE se parametr **sep** mění mezi čárkou a středníkem.

`savePlots()`

Převádí grafický výstup do souboru. K tomu slouží funkce **dev.print(png, file = 'filePath', width =width, height =height)**, kde filePath, width, height jsou parametry dané uživatelem.

`setColors()`

Vytváří R proměnnou typu vektor řetězcových hodnot **plotColors**.

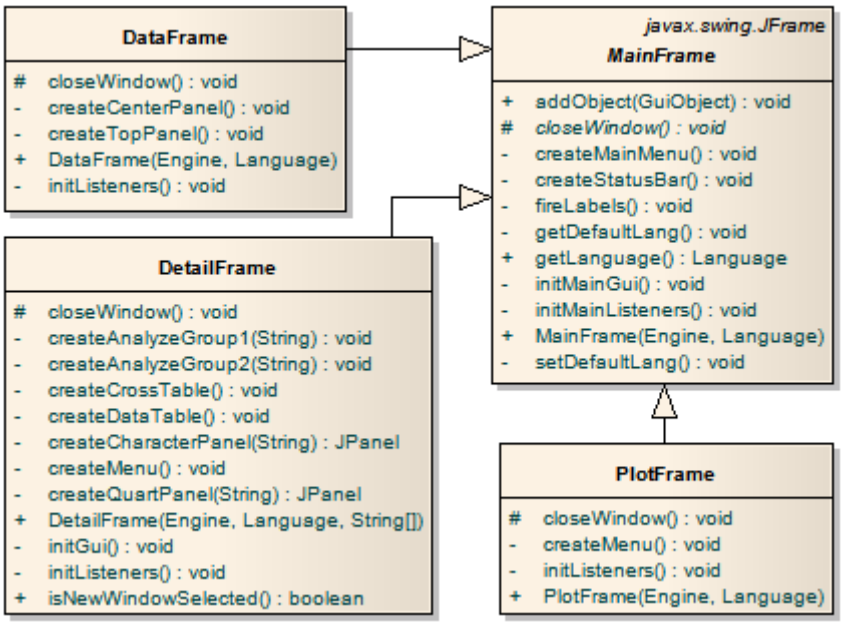
`setNAValue()`

Označuje kategorii danou uživatelem jako kategorii s chybějícími hodnotami - **data$colName[data$colName == 'value'] <- NA**.

PlotGDInterface

Třída implementující rozhraní GDInterface. Tato třída je důležitá pro propojení grafického výstupu z R jazyka.

Balíček gui



Obrázek 25: Balíček gui

Základní obsah balíčku gui

Balíček a jeho podbalíčky obsahují základní třídy pro konstrukci grafického rozhraní.

MainFrame

Jedná se o abstraktní třídu reprezentující základní kostru okna. Abstraktní metodou je zde metoda closeWindow(). Tato metoda se musí implementovat v potomcích.

`addObject()`

Slouží pro přidání grafického objektu implementující rozhraní GUIObject. Tyto objety se ukládají do kolekce **guiObjects**.

`getLanguage()`

Vrací aktuální jazykovou lokalizaci textů v okně.

`createMainMenu()`

Vytváří panel pro menu a základní prvky menu. Prvky pro změnu lokalizace a ukončení aplikace.

`createStatusBar()`

Vytváří grafický prvek obsahující popisy jednotlivých grafických prvků.

`fireLabels()`

Umožňuje změnit jazykovou lokalizaci za běhu programu. Pomocí for cyklu se projde kolekce **guiObjects**, poté se na každém objektu (který implementuje rozhraní GuiObject) zavolá metoda setLanguage, která změní jazykovou lokalizaci objektu. Spolu s metodou AddObject() umožňuje univerzální změnu lokalizace pro potomky této třídy.

`getDefaultLang()`

Zjistí z konfiguračního souboru výchozí jazykovou lokalizaci.

`initMainGui()`

Sdružuje všechny metody související s vykreslováním grafických prvků.

`initMainListeners()`

Sdružuje všechny vlastní listenery pro grafické objekty.

`setDefaultLang()`

Zapíše aktuální jazykové nastavení do konfiguračního souboru.

PlotFrame

Třída je potomkem třídy MainFrame a slouží pro vytvoření okna, do kterého se vykreslí libovolný graf. Třída obsahuje instanci na plátno, neboli statickou proměnnou **jdgBuffPanel** z třídy R_engine. Okno má oproti svému rodiči, třídě MainFrame, navíc menu akce pro práci s grafy.

`createMenu()`

Vytvoří menu akce pro práci s grafy a přidá je.

`initListeners()`

Vytvoří dodatečné listenery pro nově přidané prvky.

DataFarme

Třída je potomkem třídy MainFrame a slouží pro vytvoření okna zobrazující celou datovou tabulku. Instance této třídy se vytváří po startu aplikace.

`createCenterPanel()`

Vytváří grafické prvky pro zobrazení datové tabulky načtené ze souboru.

`createTopPanel()`

Vytváří grafické prvky, přes které se načítá soubor s datovou tabulkou.

`initListeners()`

Vytváří dodatečné listenery pro nově přidané prvky.

DetailFrame

Třída je potomkem třídy MainFrame a slouží pro vytvoření okna obsahující vybrané proměnné, analytické metody a výsledky analýz.

`createAnalyzeGroup1()`

Vytvoří záložku pro určitou proměnnou. Tato záložka může obsahovat panel s tabulkou četností a panel s grafickými možnostmi.

`createAnalyzeGroup2()`

Vytvoří záložku pro určitou proměnnou. Tato záložka může obsahovat panel s panelem charakteristik a panelem kvantilů.

`createCrossTable()`

Vytvoří záložku pro kontingenční tabulky. V této záložce lze poté vybírat pro které dvě proměnné se má tabulka sestrojit.

`createDataTable()`

Vytvoří záložku obsahující tabulku s vybranými sloupci (proměnnými). Tyto proměnné si uživatel vybírá v dialogu ChooseVariableDialog před zobrazením okna pro detail.

`createCharacterPanel()`

Vytvoří panel obsahující tabulku charakteristik pro danou proměnnou.

`createMenu()`

Vytvoří dodatečné menu akce a přidá je.

`createQuartPanel()`

Vytvoří panel obsahující tabulku kvantilů.

`initGui()`

Sdružuje všechny ostatní metody související s dodatečným vykreslováním grafických prvků.

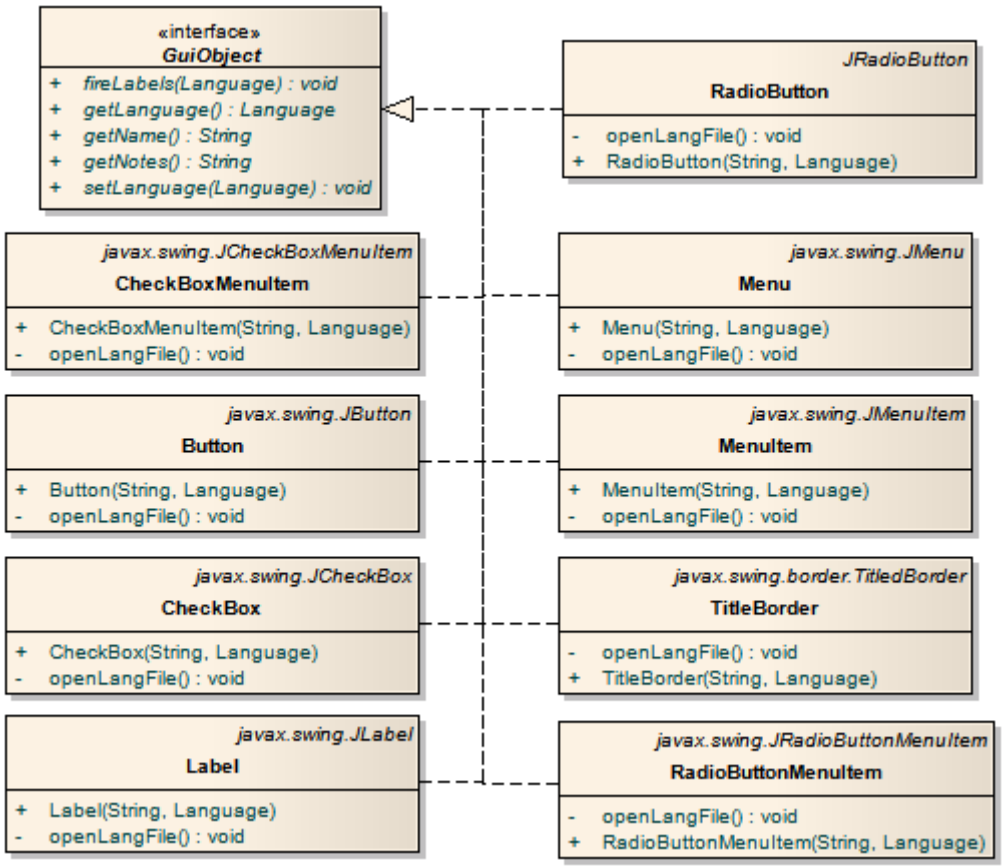
`initListeners()`

Vytvoří dodatečné listenery pro nově přidané prvky.

`isNewWindowSelected()`

Pokud metoda vrátí logickou hodnotu false, graf se vykreslí jako náhled v patřičném panelu. Pokud metoda vrátí hodnotu true, graf se vykreslí v novém okně jako plnohodnotný graf. Ve výchozím stavu je nastavena hodnota true.

Obsah podbalíčku components



Obrázek 26: Podbalíček components

Obsahuje základní grafické komponenty. Třídy jsou specializací svých protějšků z knihovny javax.swing. Rozdíl těchto tříd je v možnosti dynamické změny jazykové lokalizace a všech jejich textů. Každá třída má speciální řetězcový ID identifikátor, pod tímto identifikátorem jsou hledány příslušné překlady v externích souborech.

GuiObject

Základní rozhraní sdružující všechny grafické prvky tohoto podbalíčku.

`fireLabels()`

Slouží pro načtení textů ve zvoleném jazyce a nastavení těchto textů.

`getLanguage()`

Slouží pro zjištění aktuální jazykové lokalizace prvku.

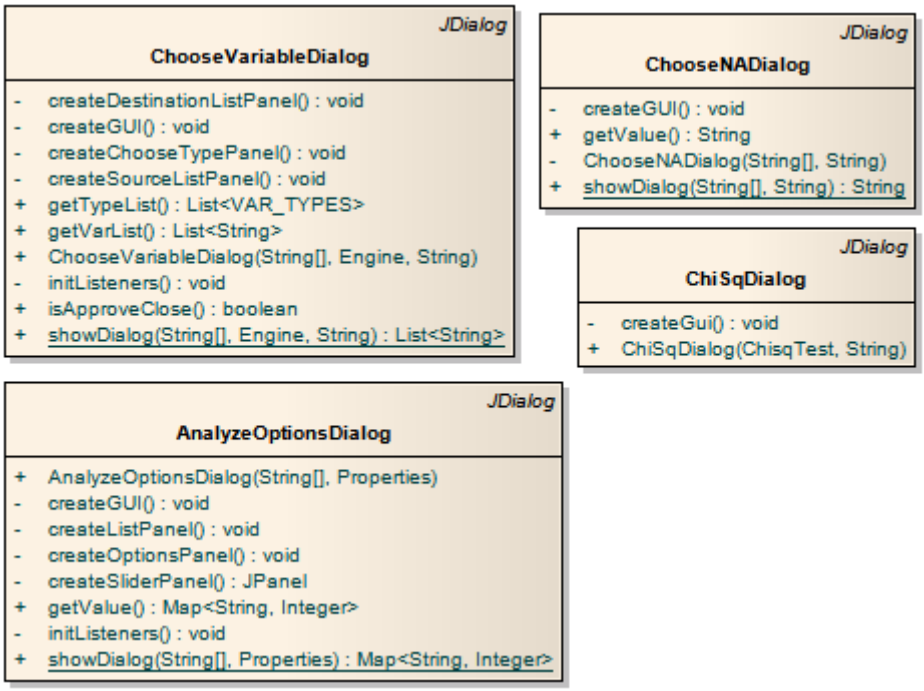
`getName()`

Vrací jméno prvku pro zobrazení. Nejedená se o ID prvku.

`getNotes()`

Vrací popis grafického prvku. Tento popis je pak využit pro status bar.

Obsah podbalíčku dialog



Obrázek 27: Podbalíček dialog

Podbalíček dialog obsahuje dialogové grafické třídy. Jedná se modální dialogy, tedy pokud je dialog spuštěn, práce s ostatními okny je zakázána dokud se dialog nezavře.

ChooseNADialog

Třída reprezentující modální okno (dialog) sloužící pro výběr chybějících hodnot. Třída má neveřejný konstruktor. Instance třídy se vytváří pomocí statické metody showDialog(), která zároveň vrací výsledek z dialogu.

`createGUI()`

Slouží pro vytvoření grafické struktury okna. Vytvoření panelu pro změnu typu sloupce a panelu pro výběr chybějících hodnot.

`getValue()`

Metoda vrací hodnotu, která se má označit za chybějící.

AnalyzeOptionsDialog

Třída reprezentující modální okno (dialog) sloužící pro výběr analytických metod.

`createDestinationListPanel()`

Grafický seznam proměnných vybraných z celkového datového souboru. Jedná se o instanci JList.

`createGUI()`

Souží pro vytvoření grafické struktury okna.

`createChooseTypePanel()`

Panel s ovládacími tlačítky a přepínacími tlačítky určující typ vybírané proměnné.

`createSourceListPanel()`

Grafický seznam všech proměnných datového souboru. Jedná se o instanci JList.

`getTypeList()`

Vrací seznam typů vybraných proměnných.

`getVarList()`

Vrací seznam názvů vybraných proměnných.

`initListeners()`

Vytvoří dodatečné listenery pro nově přidané prvky.

`isApproveClose()`

Vrací logickou hodnotu, reprezentující zda byl dialog zavřen v režimu zrušení, či pokračování.

ChooseVariableDialog

Třída reprezentující modální okno (dialog) sloužící pro výběr možností analýzy pro určitou proměnnou. Třída má veřejný konstruktor. Instance třídy se vytváří tedy pomocí konstruktoru, nebo pomocí předprogramované statické metody showDialog(), která zároveň vrací výsledek z dialogu - mapu vybraných analytických možností.

`createGUI()`

Slouží pro vytvoření grafické struktury okna.

`createListPanel()`

Panel s grafickým seznamem proměnných k dispozici.

`createOptionsPanel()`

Panel s tlačítky pro nastavení typů analýz.

`createSliderPanel ()`

Panel s posuvným prvkem, sloužící pro výběr hodnoty intervalu rozdělení histogramu.

`getValue ()`

Vrací mapu uživatelem nastavených a vybraných možností.

`initListeners()`

Vytvoří dodatečné listenery pro nově přidané prvky.

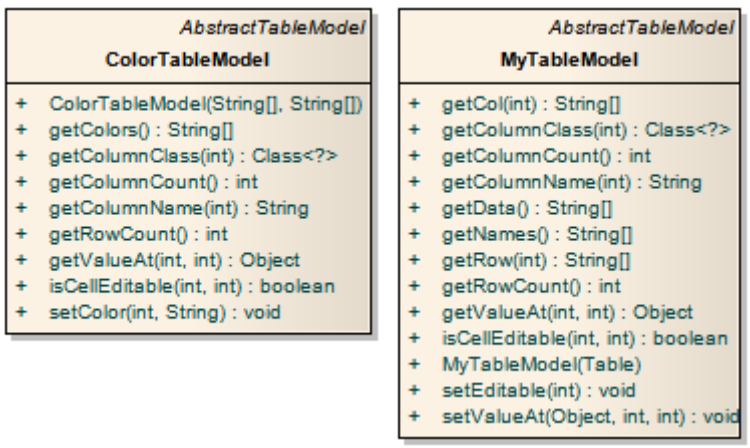
ChiSqDialog

Třída reprezentující modální okno (dialog), sloužící k zobrazení informací o Chí kvadrát testu nezávislosti. Instance se vytváří pomocí konstruktoru.

`createGUI()`

Slouží pro vytvoření grafické struktury okna.

Obsah podbalíčku model



Obrázek 28: Podbalíček model

ColorTableModel

Třída reprezentující strukturu pro tabulku, v níž lze nastavovat barvy jednotlivým kategoriím. Instance se vytváří pomocí konstruktoru s parametry určující jména kategorií a jména barev, tedy tabulka o 2 sloupcích.

`getColors()`

Vrací seznam nastavených barev.

`setColor()`

Nastaví hodnotu barvy na příslušný řádek.

`getColumnClass()`

Vrací typ formátování sloupce (celá čísla, desetinná, řetězec, atd.).

`getColumnCount()`

Vrací počet prvků ve sloupci.

`getColumnName()`

Vrací název požadovaného sloupce.

`getRowCount()`

Vrací počet prvků v řádku.

`getValueAt()`

Vrací hodnotu na souřadnicích zadaných uživatelem

`isCellEditable()`

Vrací logickou hodnotu určující, zda je buňka editovatelná, či nikoli.

MyTableModel

Třída reprezentující strukturu pro komponentu JTable, reprezentující tabulku hodnot. Metody getColumnClass(), getColumnCount(), getColumnName(), getRowCount(), getValueAt(), IsCellEditable(), jsou popsány výše.

`getCol()`

Vrací požadovaný sloupec hodnot.

`getData()`

Vrací data obsažená v tabulce.

`getNames()`

Vrací jména všech sloupců.

`getRow()`

Vrací požadovaný řádek hodnot.

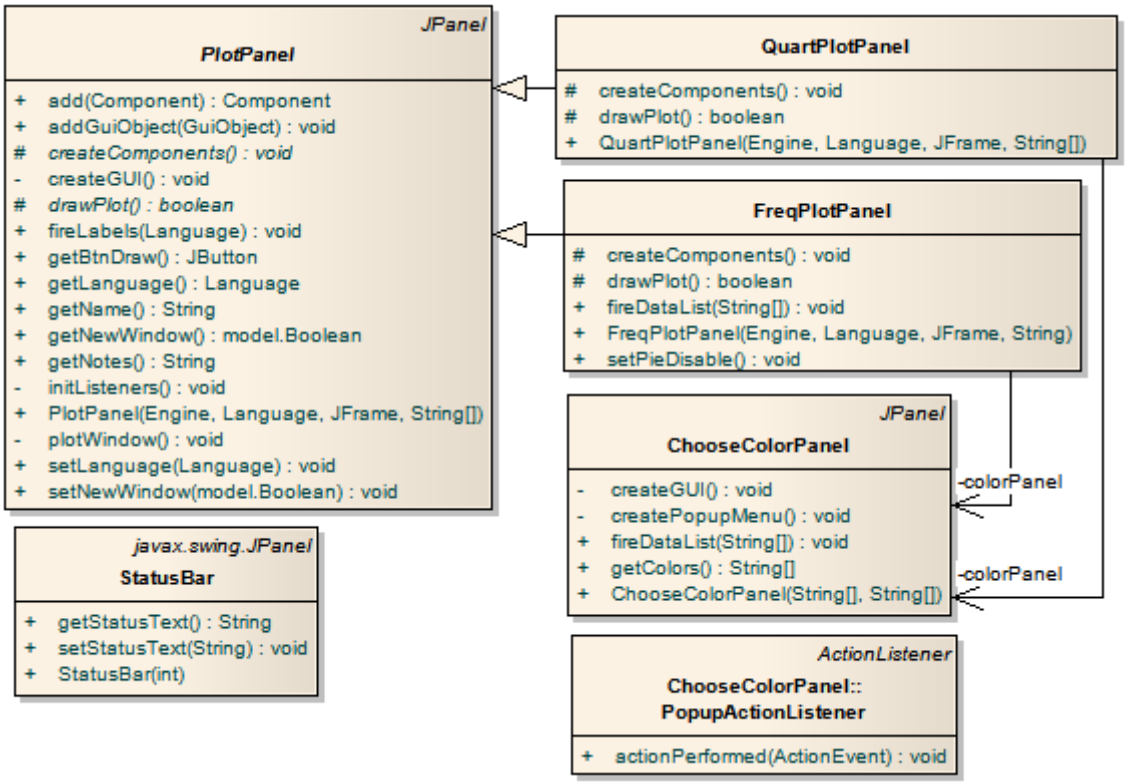
`setEditable()`

Nastaví sloupec buněk příznakem editovatelný.

`setValueAt()`

Nastaví hodnotu na souřadnice zadané uživatelem.

Obsah podbalíčku panels



Obrázek 29: Podbalíček panels

StatusBar

Třída reprezentuje komponentu stavového řádku. Jedná se o specializaci komponenty JPanel.

`getStatusText()`

Vrací zobrazovaný text ve stavovém řádku.

`setStatusText(text)`

Nastavení zobrazovaného textu.

ChooseColorPanel

Třída reprezentuje tabulku pro výběr barev. Tento panel se dále skládá z rolovací nabídky obsahující názvy jednotlivých barev. Dále obsahuje pro tuto rolovací nabídku vnitřní třídu PoupupActionListener, implementující akce.

`createGui()`

Souží pro vytvoření grafické struktury.

`createPopupMenu()`

Vytváří rolovací nabídku s názvy barev.

`fireDataList()`

Přepisuje hodnoty barev v tabulce pro jednotlivé kategorie.

`getColors()`

Vrací seznam názvů barev.

PlotPanel

Abstraktní třída reprezentující základ panelu pro vykreslování grafů. Implementuje rozhraní GuiObject. Obsahuje referenci na jgdBuffPanel z třídy R_engine a další ovládací prvky. Dále obsahuje abstraktní metody createcomponents() a drawPlot(). Metody addGuiObject(), fireLabels(), setLanguage(), getLanguage() jsou popsány ve třídě MainFrame.

`createGui()`

Slouží pro vytvoření grafické struktury.

`getBtnDraw()`

Vrátí referenci na tlačítko pro vykreslování grafů.

`getName(), getNotes()`

Metody zděděné z rozhraní GuiObject. V této třídě neimplementované.

`getNewWindow(), setNewWindow()`

Nastavení a zjištění parametru, určujícího zda má být graf vykreslen v této třídě, nebo ve třídě PlotFrame.

`initListeners()`

Vytvoří listenery pro grafické prvky.

`plotWindow()`

Vykreslí graf v aktuální třídě, nebo ve třídě PlotFrame.

QuartPlotPanel

Potomek třídy PlotPanel. Slouží pro vykreslování krabicového grafu.

`drawPlot()`

Nastaví barvy grafu a vykreslí krabicový graf.

`createComponents()`

Vkládá panel pro výběr barev. ChooseColorPanel.

FreqPlotPanel

Potomek třídy PlotPanel. Obsahuje navíc grafické prvky pro výběr typu grafu pro tabulky četností, popř. kontingenční tabulky.

`drawPlot()`

Nastaví barvy grafu a vykreslí kruhový graf nebo histogram.

`createComponents()`

Vkládá panel pro výběr barev, ChooseColorPanel. Dále vkládá tlačítka pro výběr typu grafu (kruhový/histogram) a nabídku pro typ hodnot (absolutní/relativní).

`setPieDisable()`

Nastavuje kruhový graf na disable. Slouží při použití FreqPlotPanel pro kontingenční tabulky.

9 Zhodnocení vlastního řešení

Software JRA je plně funkční aplikace. Svojí funkcionalitou zcela odpovídá původně vytčenému cíli práce a také specifikovaným požadavkům v navrhovaném modelu.

Mezi výhody bezpochyby patří dostupnost vícejazyčného rozhraní, konkrétně je podporována angličtina a čeština. Další výhodou jsou externí jazykové soubory, jejichž dostupnost nebyla původně zahrnuta v cílech práce. Možnost přidání jiné jazykové lokalizace by vyžadovala přepsat pouze jeden z přiložených jazykových souborů. Díky externímu umístění souborů není potřeba rekompilace softwaru. Uživatelské prostředí se dále snaží být intuitivní a univerzální.

Oproti tomu větší univerzálnost je zároveň i nevýhodou. Obsahuje pouze základní analytické metody. Pro složitější a profesionálnější analýzu je nutnost použít software jiný. Ač se snaží být prostředí intuitivní, styl a možnosti ovládání jsou podstatně jiné než u ostatních softwarů, což může zprvu působit negativně.

Z hlediska rozšiřování softwaru o další funkcionality je nutné doprogramovat patřičné metody do rozhraní **engine**. Dále implementovat ovládací prvky do GUI tříd. Pokud nevyhovuje implementace metod v jazyce R, může se napsat implementace vlastní. Je zde ale nutné, aby nově vzniklá třída implementovala rozhraní **engine.** Dále stačí pouze zaměnit referenci z **R_engine** na třídu novou. Pokud se bude rozšiřovat grafické rozhraní o GUI prvky s možností dynamické změny textu, pak stačí nově vytvořenému prvku implementovat rozhraní **GuiObject**.

Obtížnost řešené implementace JRA je především v propojení technologie R a Java. Je zde totiž nutná transformace datových objektů mezi technologiemi, zachycování nežádoucích a výjimečných stavů, ošetření vstupních objektů z hlediska správného formátování, aj. Další obtížností implementace je přepínání jazykové lokalizace za běhu aplikace. To vyžaduje externí jazykové soubory a rozšíření zabudovaných grafických prvků jazyka Java o dodatečné atributy a metody.

Význam vlastního řešení vidím v možnosti rozšíření vlastního povědomí o problematice analýzy dat a možnostech speciálních typů softwaru pro statistickou analýzu. Cenná byla také zkušenost a praxe propojení dvou různých technologií a zjištění úskalí tohoto propojení. Získané poznatky o struktuře a využití softwaru R.

10 Shrnutí výsledků

V práci se úspěšně povedlo propojit dvě různé technologie (R, Java). Dále se povedlo v těchto technologiích implementovat všechny vytyčené funkcionality.

Výsledný software JRA je tedy plně funkční na platformě Windows. Navíc obsahuje možnost změny jazyku za běhu softwaru.

Naopak se nepovedlo vytvořit software platformě nezávislý. Vzhledem k odlišné syntaxi na Unix a Windows systémech, je problém s inicializací knihovny JavaGD, která se stará o grafické plátno. Zatímco na Windows proběhla inicializace úspěšně, na Unix se inicializace nezdařila. Ostatní funkce fungují na obou systémech. Dalším problémem stojícím v platformní nezávislosti jsou knihovny jazyka R. Struktura balíčků a nativních knihoven se mezi systémy liší.

Výsledky této práce přinesly základní znalost jazyka R, jeho datových struktur, funkcí a konstrukce vlastních funkcí. Dále znalost a používání analytických metod, konkrétně rozdělení četností, popisných charakteristik, kontingenční tabulky, chí - kvadrátu testu nezávislosti. Tyto základy jsou obecně použitelné i mimo tuto práci. Implementace těchto metod v softwaru JRA může sloužit jako příklad jejich použití. Znalost těchto základů dále dovoluje i jejich využití v aplikaci RExcel. RExcel umožňuje psát příkazy R uvnitř excelové tabulky, proto se zde využije základní popis jazyka R, funkce, datové struktury, proměnné. Dále se v něm využijí popsané analytické metody a grafy. Například pro zjištění směrodatné odchylky stačí napsat do buňky excelu příkaz `sd(data)`, kde proměnná data označuje oblast excelových dat. Výsledek této funkce se vloží do označení excelové buňky. Obdobný postup funguje při zadání jakéhokoli příkazu. Rozdíl mezi JRA a RExcel je tedy v tom, že JRA nepředpokládá znalost příkazů R (a neumožňuje ji využívat), kdežto RExcel tuto znalost předpokládá a uživateli umožní pracovat s R v prostředí tabulkového procesoru, které zpravidla dobře ovládá.

11 Závěry a doporučení

Po zhodnocení vlastního řešení (softwaru JRA) z hlediska použití, dostupných funkcí, univerzálnosti a uživatelské přívětivosti a porovnání těchto výsledků s existujícími softwary jsem dospěl k následujícím závěrům.

Z hlediska obecných funkcionalit, jako jsou import datového souboru, úprava datového souboru, transformace proměnných aj., vyšlo vlastní řešení JRA na srovnatelné úrovni s ostatními řešeními. JRA nedovoluje úpravu datového souboru, ani to nebylo cílem práce. Oproti tomu MS Excel nezahrnuje transformaci proměnných. Existující analytické softwary však řeší většinou obě funkcionality.

Software JRA lze využít pro základní typy analýzy datového souboru. Například pro výpočty a grafické zobrazení rozdělení četností, popisné charakteristiky (průměr, modus, medián) a závislosti dvou proměnných (kontingenční tabulka). Tabulkový editor ale poskytuje více možností pro uživatelský zásah, je zde ovšem nutnost znalosti metod a jejich použití. JRA k těmto funkcím přidává ovládací prvky, proto zde odpadá hlubší znalost těchto metod a jejich použití.

Plugin RExcel nebo ostatní komerčně dostupné produkty jsou vhodné také pro pokročilejší analýzu. Například pro regresní analýzu, clusterovou analýzu, ANOVA testy, vícerozměrné grafy aj. Plugin RExcel kombinuje složitější analytické metody R jazyka a uživatelskou přívětivost Excelu. Pro velkou uživatelskou variabilitu je vhodné použít R jazyk. Zde je ale nutnost znát základní syntaxi.

Pro pokročilejší analytické metody při použití JRA by bylo nutné doprogramovat přístup na příslušné analytické metody R a výstupy jejich výsledků. To by znamenalo připsání nových metod k rozhraní **engine** a implementování ovládacích prvků. Z hlediska programátorského je dále možný vývoj skriptů v R jazyce, či vytváření vlastních funkcí v makro jazyce Visual Basic pro MS Excel, či vytváření funkcí v R jazyce pro prostředí RExcel. S každou z těchto metod souvisí testování funkcí pro různé případy použití.

Pro R jazyk existuje mnoho projektů zabývající se grafickým uživatelským rozhraním pro jednotlivé metody. Studium a porovnání těchto projektů nebyl cíl ani účel práce, cílem bylo spojení technologií R a Java. Tento odstavec tedy ukazuje paralelní možnosti využití grafického rozhraní v R. Z těchto projektů jsem vybral pro porovnání R Commander [16]. Grafické rozhraní jednotlivých metod vlastního řešení je integrální součástí celé aplikace. To způsobuje sice přehlednost dostupných funkcí a jejich vstupů, ale pro každou funkci zvlášť je nutné sestavení grafických prvků, validace vstupních hodnot, patřičné zobrazování výsledků uvnitř aplikace. Oproti tomu R Commander obsahuje grafické dialogy pro jednotlivé funkce. Tyto dialogy umožní nastavování vstupních parametrů přímo v dialogu, odpadá tím nutnost psát hodnoty přímo do R funkce. Například pro načtení datového souboru se zobrazí dialog pro zadání názvu, umístění, specifikování oddělovače, atd. Výsledky funkcí zobrazuje R Commander ve formě grafů nebo textových výstupů do konzole. R Commander existuje jako knihovna jazyka R, odpadá zde tedy instalace dodatečného softwaru. Pro budoucí využití R jazyka by bylo vhodné využít těchto externích knihoven pro grafické nástavby.

Po získaných zkušenostech bych pro práci s grafickým rozhraním R jazyka zvolil možnost využití knihovny R Commander. Hlavním rozdílem mezi pluginem RExcel a knihovnou R Commander je totiž řešení integrace. K práci v RExcel je potřeba instalace tří nezávislých softwarů. Jedná se o R, MS Excel a samotný RExcel plugin. K práci v R Commander stačí pouze software R a instalace knihovny jako integrálního modulu.

12 Seznam použité literatury

[1] Provalis Research: SimStat. *Provalis Research* [online]. [cit. 2015-01-25]. Dostupné z: http://provalisresearch.com/products/simstat/.

[2] AASLAND, Aadne. *A User Manual for SPSS Analysis: CNAS 2008 Survey Data*. NIBR: Norwegian Institute for Urban and Regional Research, 2008.

[3] GOODDATA CORPORATION. *GoodData Reference Guide*. 3.1.2015. [cit. 2015-01-25]. Dostupné z: http://help.gooddata.com/doc/public/pdf/GoodData%20Reference%20Guide.pdf.

[4] Baier Thomas, & Neuwirth Erich (2007). Excel :: COM :: R. Computational Statistics, Volume 22, Number 1/April 2007. PhysicaVerlag.

[5] SKALSKÁ, Hana. Data mining a klasifikační modely. Vyd. 1. Hradec Králové: Gaudeamus, 2010, 154 s. Recenzované monografie. ISBN 978-80-7435-088-7.

[6] S-PLUS 4 Guide to Statistics, Data Analysis Products Division, MathSoft, Seattle.

[7] ŘEZANKOVÁ, Hana. *Analýza dat z dotazníkových šetření*. Professional Publishing, Praha 2007, 212 s. ISBN 978-80-86946-49-8.

[8] MYATT, Glenn J. *Making sense of data: a practical guide to exploratory data analysis and data mining*. Hoboken, N.J.: Wiley-Interscience, c2007, xii, 280 p. ISBN 9780470074718.

[9] CRAWLEY, Michael J.. *The R Book.* 2nd Edition. John Wiley & Sons, Ltd, 2012. 1076 s. ISBN: 978-0-470-97392-9

[10] ANDĚL, Jiří. *Matematická statistika*. Brno: SNTL, 1978, s. 210-211.

[11] COHEN, Yosef a Jeremiah Y COHEN. *Statistics and data with R: an applied approach through examples*. Chichester, U.K.: Wiley, 2008, xviii, 599 p. ISBN 04-707-5805-8.

[12] R Core Team (2014). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

[13] Simon Urbanek (2013). rJava: Low-level R to Java interface. R package version 0.9-6. http://CRAN.R-project.org/package=rJava.

[14] Simon Urbanek (2012). JavaGD: Java Graphics Device. R package version 0.6-1. http://CRAN.R-project.org/package=JavaGD.

[15] Lemon, J. (2006) Plotrix: a package in the red light district of R. R-News, 6(4): 8-12.

[16] Fox, J. (2005). The R Commander: A Basic Statistics Graphical User Interface to R. Journal of Statistical Software, 14(9): 1--42.

13 Přílohy

1) Java-R Analysis (JRA) – Uživatelská příručka.
2) CD – Obsahuje instalátor softwaru JRA, R verze 3.1.1., zdrojový kód s dokumentací.

Příloha č. 1

Java-R Analysis (JRA)

Uživatelská příručka

Vojtas Pavel

Požadavky na systém

Doporučené požadavky:

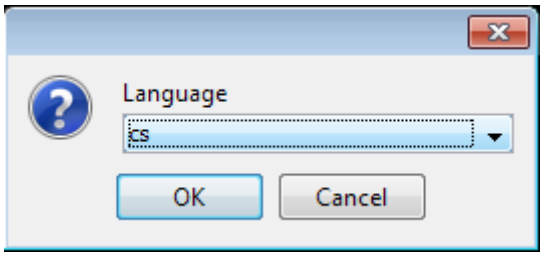
- Operační systém: MS Windows
- R: min. verze 3.1.x
- Java Runtime Environment: 1.8.0_25

Instalace a odinstalování

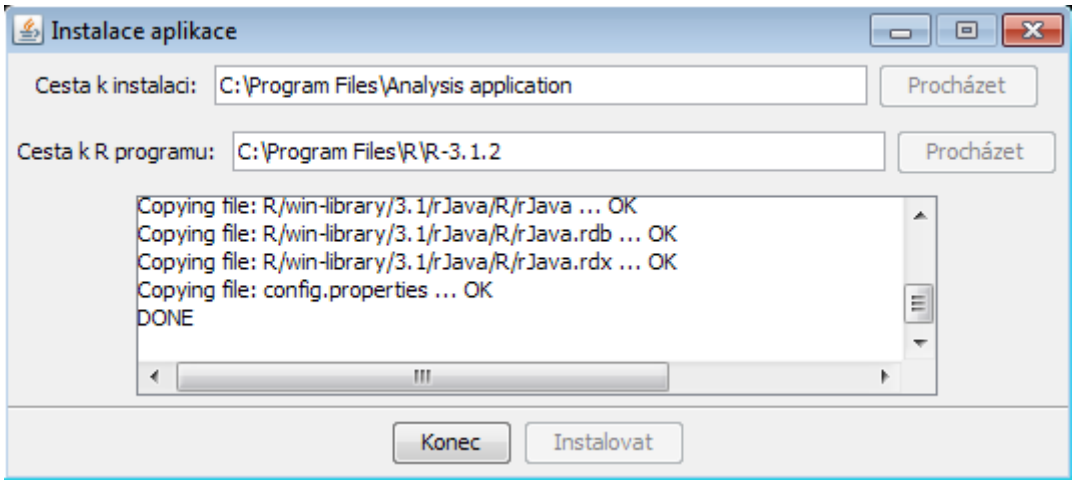
Instalace

> Instalace se spouští souborem jra_setup.exe v režimu správce.

> Po spuštění se vybere jazyk instalace.



> V následujícím okně se vybere adresář pro instalaci, adresář s nainstalovaným R programem. Stiskne se tlačítko **Instalovat** pro instalaci aplikace.



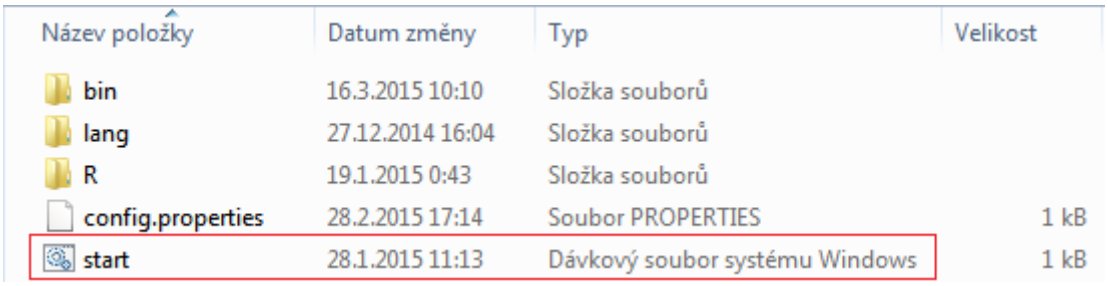
> Po nainstalování se stiskne tlačítko **Konec**.

Odinstalování

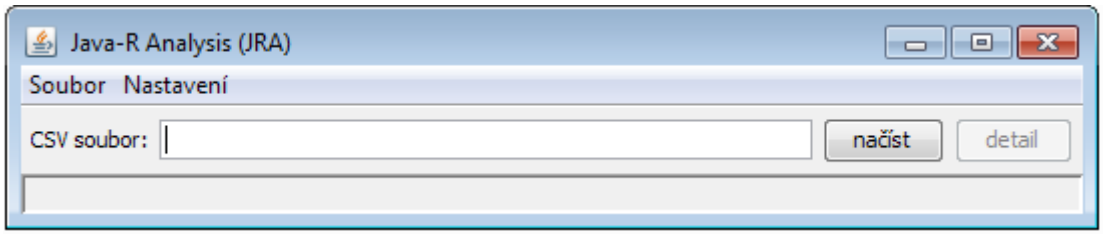
> Pro odinstalování se smaže adresář s nainstalovanou aplikací.

Spuštění a popis aplikace

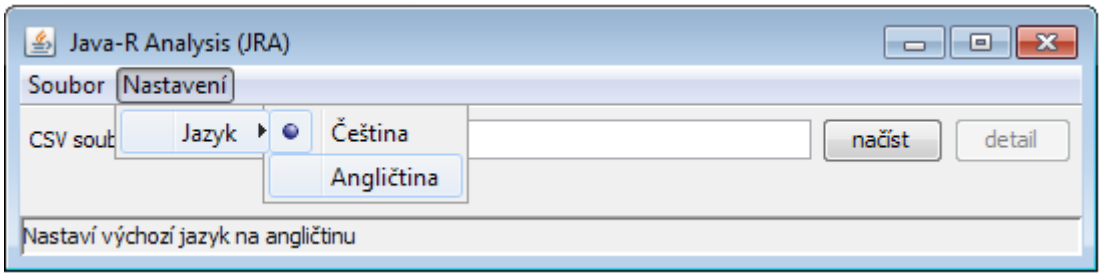
> Spuštění aplikace se provede spuštěním dávkového souboru start.bat v adresáři s instalací aplikace.



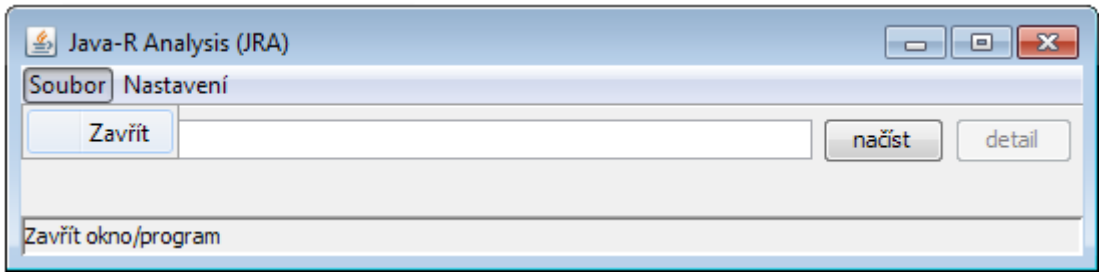
> Po spuštění aplikace se zobrazí okno pro načtení souboru.



> Pro změnu jazyku aplikace slouží menu **Nastavení -> Jazyk -> Čeština/Angličtina**.

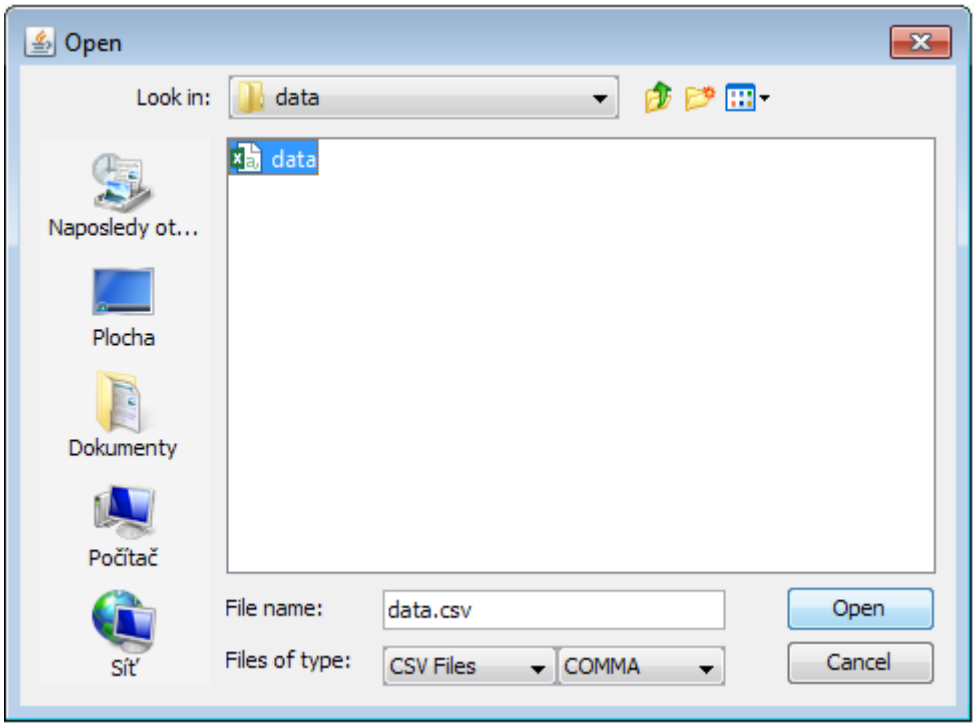


> Pro ukončení aplikace slouží menu **Soubor -> Zavřít**.

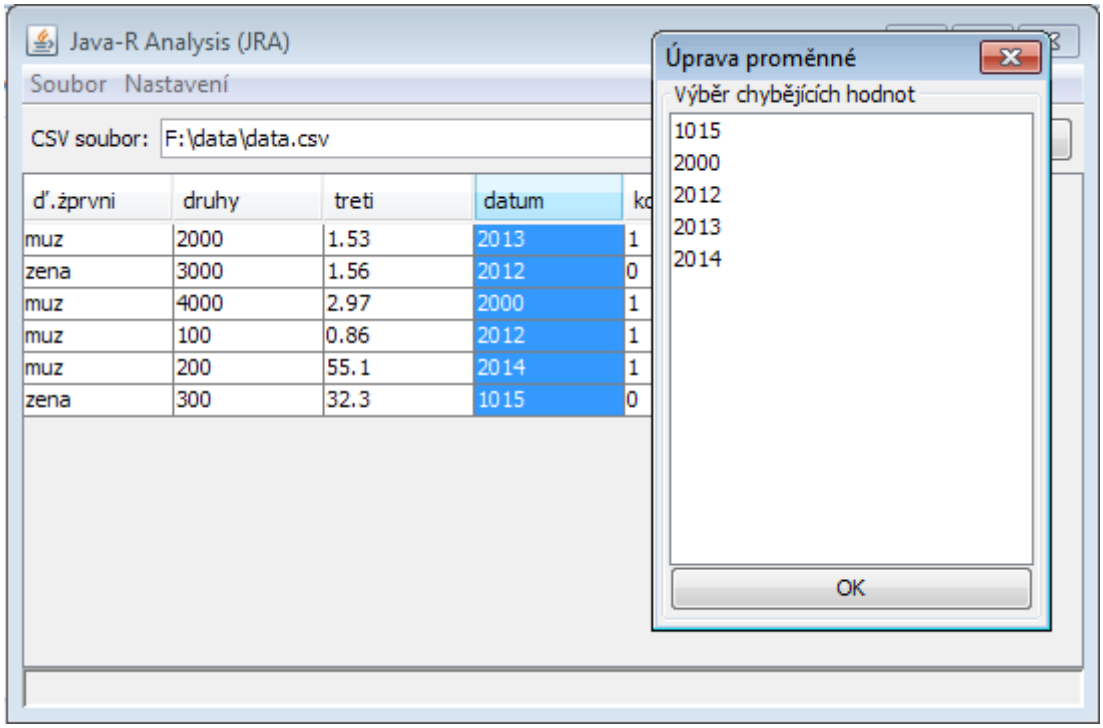


Načtení datového souboru

> Načtení se provede tlačítkem **načíst**. Zobrazí se dialog. V dolní části dialogu je typ oddělovače. Na výběr je **COMMA** (čárkou), **SEMICOLON** (středník).

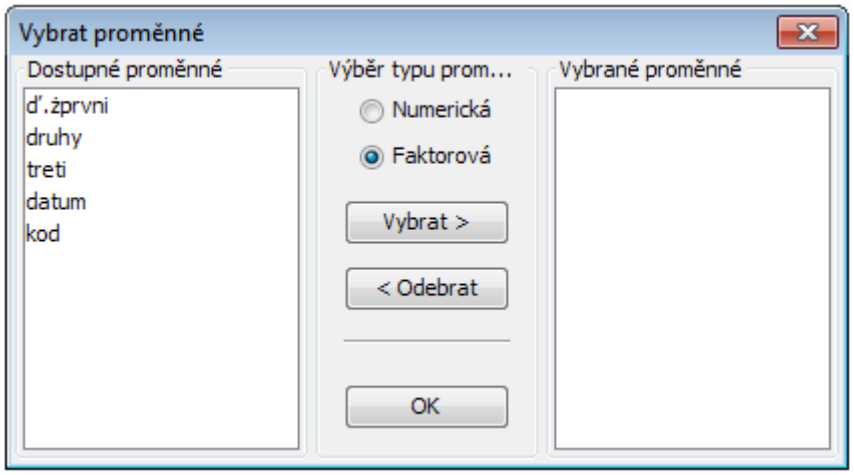


> Po načtení datového souboru se zobrazí tabulka s daty. Dále po dvojkliku na název sloupce lze označit chybějící hodnoty.

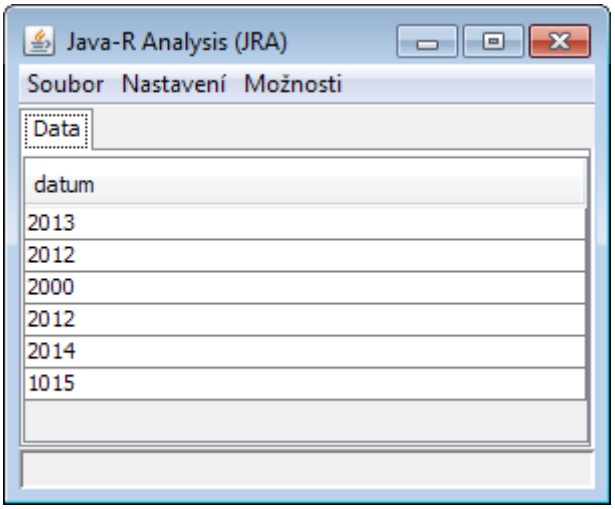


Detail vybraných proměnných

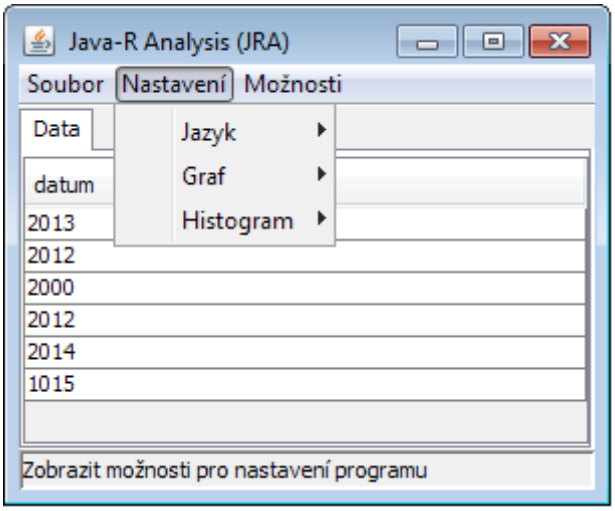
> Po stisknutí tlačítka **detail** se otevře dialog pro výběr proměnných a určení typu. Pomocí tlačítka **Vybrat** se proměnná vybere k analýze. Tlačítkem **Odebrat** odebere.



> Po výběru proměnných a stisku tlačítka **OK** se otevře okno pro jejich analýzu.

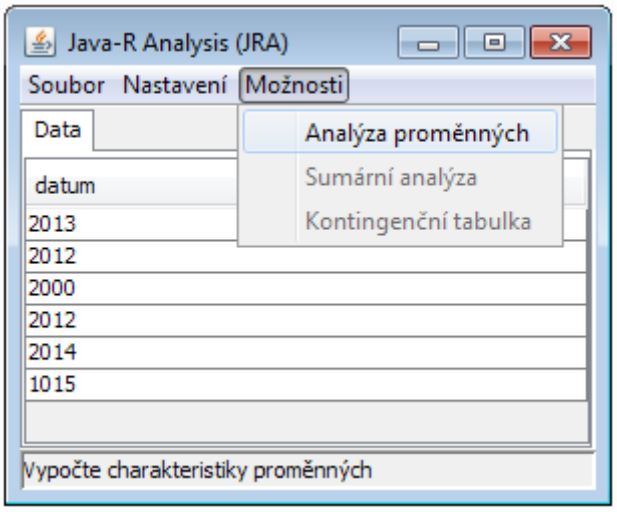


> Nastavení obsahuje možnost vykreslovat graf v novém okně a nastavení histogramu.



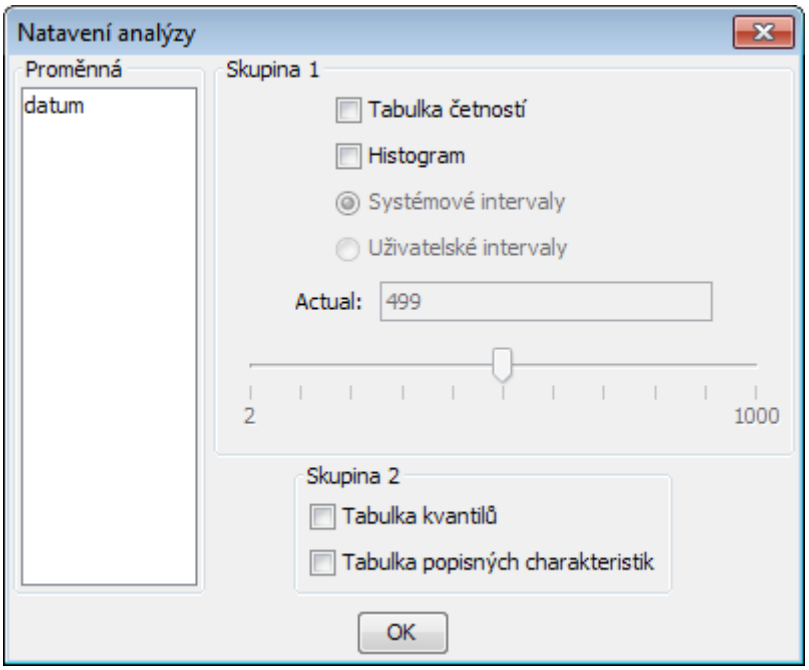
Analytické metody

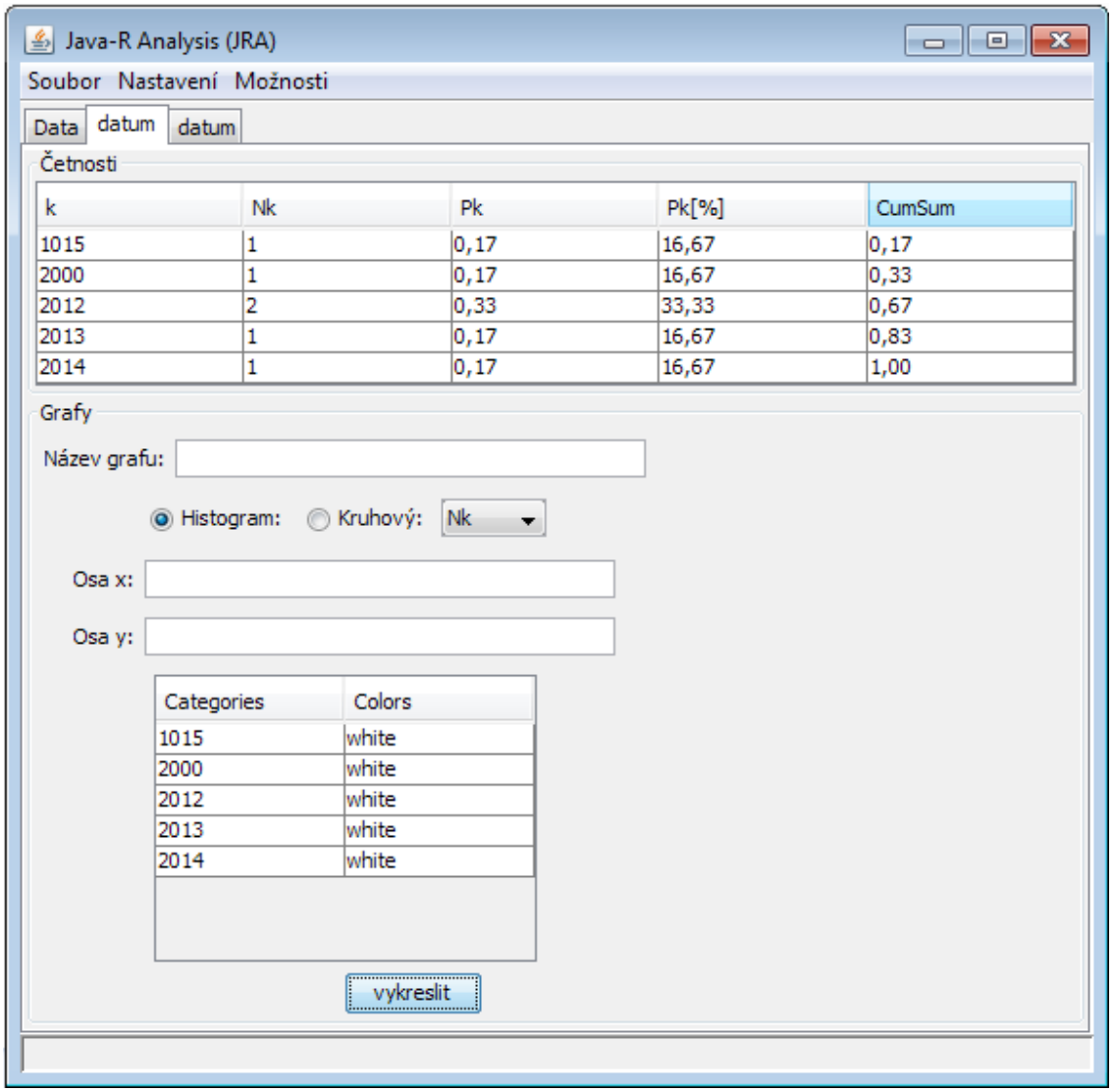
> Pro výběr analytických metod slouží nabídka **Možnosti**.



> Po výběru možnosti **Analýza proměnných** se zobrazí dialog pro výběr proměnné a možnosti dostupných analýz. Skupina 1 obsahuje na výběr možnost analýzy ve formě **Tabula četností** (zvolením této volby se vypočítá a zobrazí tabulka četností), **Histogram** (zvolením této volby se zobrazí možnost vykreslení grafu, histogramu). Počet intervalů histogramu se volí podle systému nebo podle uživatele, přepínač **Systémové/uživatelské intervaly**. Skupina 2 obsahuje na výběr možnost analýzy ve formě **Tabulka kvantilů** (zvolením této možnosti se pro numerickou proměnnou vypočítá tabulka kvantilů), **Tabulka popisných charakteristik** (zvolením této možnosti se spočítají a zobrazí popisné charakteristiky).

> Po stisknutí tlačítka **OK** se vypočítají zvolené okruhy analýzy a zobrazí se ve formě záložek s názvem dané proměnné.





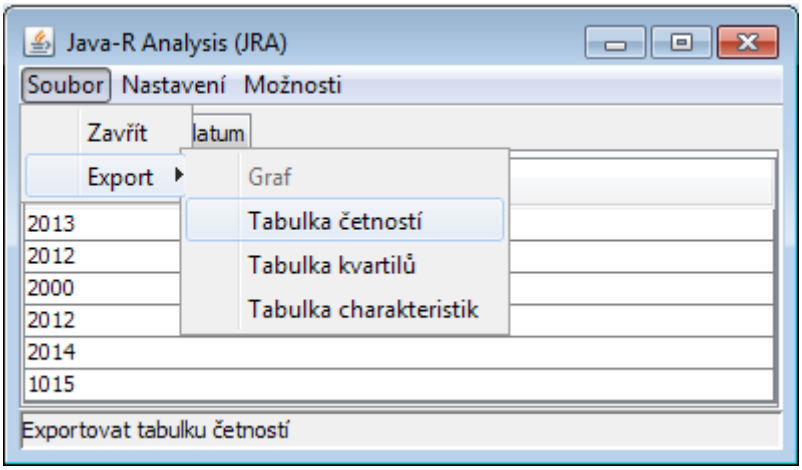
> Pro vykreslení grafu dané analýzy slouží tlačítko **vykreslit**. Předtím lze v tabulce Colors změnit barvu jednotlivých kategorií **Pravé tlačítko myši na buňku s barvou -> výběr barvy z přednastavených**. Dále lze vybrat typ grafu z možností **Histogram** a **Kruhový**. Dále lze specifikovat typ hodnot z možností **Nk** (absolutní četnost), **Pk** (relativní četnost), **Pk%** (relativní četnost v procentech). Po stisknutí tlačítka vykreslit se vykreslí graf.

> Po výběru možnosti **Sumární analýza**, se zobrazí záložka s tabulkou charakteristik pro všechny dostupné proměnné, popř. tabulkou kvantilů pro všechny numerické.

> Po výběru možnosti **Kontingenční tabulka**, se zobrazí záložka s možností výběru dvou proměnných, pro kterou se má kontingenční tabulka spočítat.

Export výsledků

> Pro export výsledků se použije možnost v menu **Soubor -> Export -> Typ exportu**.



> Po výběru **Tabulka četností,** se exportuje tabulka četností pro proměnnou, která má aktuálně otevřenou záložku. Pokud uživatel nezvolil možnost vytvoření tabulky četností, nic se neexportuje. Po výběru **Tabulka kvantilů** a **Tabulka charakteristik** je funkce obdobná jako u tabulky četností.

> Výběr exportu **Graf** je k dispozici, pouze pokud je odznačena možnost v menu **Nastavení -> Graf -> V novém okně**. Poté se graf exportuje do souboru PNG. Pokud je možnost **V novém okně** zvolena, export probíhá skrz stejné menu v okně grafu.

Univerzita Hradec Králové
Fakulta informatiky a managementu
Akademický rok: 2014/2015

Studijní program: Aplikovaná informatika
Forma: Prezenční
Obor/komb.: Aplikovaná informatika (ai3-p)

Podklad pro zadání BAKALÁŘSKÉ práce studenta

PŘEDKLÁDÁ:	ADRESA	OSOBNÍ ČÍSLO
Vojtas Pavel	Lhotka 188, Česká Třebová - Lhotka	I1201039

TÉMA ČESKY:

Analýza dat s využitím jazyků R a Java

NÁZEV ANGLICKY:

Data analysis using the R language and Java

VEDOUCÍ PRÁCE:

Prof. RNDr. Hana Skalská, CSc. - KIKM

ZÁSADY PRO VYPRACOVÁNÍ:

Cíl práce: Práce se bude zabývat problematikou analýzy dat a možnostmi jejich zpracování. Zaměří se na metody a nástroje jazyka R a vytvoření uživatelského rozhraní v Java, implementujícího tyto metody.

Osnova práce:
- Rešerše literatury a vybraných produktů
- Analýza požadovaných funkcionalit uživatelského rozhraní
- Studium jazyka R a vybrané knihovny
- Vybrané analytické metody a jejich implementace v R jazyce
- Interpretace výsledků a grafy v R jazyce
- Vytvoření uživatelského rozhraní programu v jazyce Java a jeho popis
- Popis implementace programu v jazyce Java

SEZNAM DOPORUČENÉ LITERATURY:

[1] ŘEZÁNKOVÁ, Hana. Analýza dat z dotazníkových šetření. Professional Publishing, Praha 2007, 212 s. ISBN 978-80-86946-49-8.
[2] COHEN, Yosef a Jeremiah Y COHEN. Statistics and data with R: an applied approach through examples. Chichester, U.K.: Wiley, 2008, xviii, 599 p. ISBN 04-707-5805-8.
[3] CRAWLEY, Michael J.. The R Book. 2nd Edition. John Wiley & Sons, Ltd, 2012. 1076 s. ISBN: 978-0-470-97392-9.
[4] Webová stránka R projektu. [online] Dostupné z: http://www.r-project.org/

Podpis studenta: Vojtas ..... **Datum:** 13.10.2014

Podpis vedoucího práce: I. Skalská ..... **Datum:** 13.10.2014