Vysoká škola ekonomická v Praze

Fakulta informatiky a statistiky

Studijní program: Kvantitativní metody v ekonomice

Studijní obor: Statistika a ekonometrie

POPISNÁ STATISTIKA V R
S APLIKACÍ NA REÁLNÁ DATA

Bakalářská práce

Autorka bakalářské práce: Eva Pirohová

Vedoucí bakalářské práce: Mgr. Milan Bašta, Ph.D.

Akademický rok 2016/2017

Prohlášení

Prohlašuji, že jsem bakalářskou práci zpracovala samostatně a že jsem uvedla všechny použité prameny a literaturu, ze kterých jsem čerpala.

V Praze dne 22. května 2017

…….
podpis

Poděkování

Chtěla bych poděkovat panu Mgr. Milanu Baštovi, Ph.D. za vedené mé práce, ochotu a pomoc se získáním dat, za cenné rady a poskytnuté materiály, díky kterým jsem nabyla část znalostí potřebné k psaní bakalářské práce.

Rovněž děkuji panu Ing. Jiřímu Procházkovi za ochotu pomoci s nejasnostmi týkajících se jazyka R.

Abstrakt

Cílem práce je představit možnosti popisné statistiky ve statistickém softwaru R a vysvětlit princip zadávání příkazů a funkcí v tomto softwaru. V teoretické části práce bude uživatel seznámen s informacemi, co je to popisná statistika a na jakém principu funguje software R. Důležitou součástí jsou uvedené funkce a kódy, které se zadávají do příkazového řádku či skriptu v R. Praktická část obsahuje aplikaci principů popisné statistika a příslušných příkazů a funkcí v R na reálná data včetně ilustrací v podobě jednotlivých typů grafů. Pod každou konkrétní mírou popisné statistiky či grafickým výstupem bude uveden příkaz v podobě funkce v R s vysvětleným postupem a komentářem.

Práce obsahuje vše podstatné, co by měl znát každý, kdo se statistikou a s prací v R začíná. I přes to, že je práce s R pro ty, kteří neprogramují, ze začátku složitá, bude práce koncipována tak, aby i začátečník byl schopen analyzovat data z hlediska popisné statistiky.

Klíčová slova

Popisná statistika, R, Funkce, Argument, Graf

Abstract

The aim of this thesis is to introduce the statistical software R, its use in descriptive statistics and to explain the principle of entering commands and functions in R. In the theoretical part of the thesis the user will be given the basic information about descriptive statistics and the key principles of R software. An important part of the thesis is an illustration of functions and codes which are to be entered into a script or a command window. The practical part represents the way of applying descriptive statistics on real data by using functions and codes in R, including the corresponding types of graphs. There is always an explanation following the example.

This thesis contains all the important information which those who start with basic statistical analysis and R software should know. Despite the fact that working with R may be rather complicated at the beginning, this thesis is written in such a way so that it can be read by a beginner in R and statistical analysis.

Keywords

Descriptive statistics, R, Function, Argument, Graph

Obsah

Úvod **3**

1 Programovací jazyk R a RStudio **4**

1.1 Základní poznatky práce s R 4

1.2 Důležité funkce pro práci s daty 5

1.2.1 Použité příkazy a funkce 5

1.2.2 Tvorba vlastní funkce v R 7

1.2.3 Načtení dat do R 7

2 Popisná statistika **9**

2.1 Základní statistické pojmy 9

2.2 Odlehlé a extrémní hodnoty 10

2.3 Míry polohy a variability 11

2.3.1 Nominální proměnná 11

2.3.2 Ordinální proměnná 14

2.3.3 Kvantitativní proměnná 18

2.4 Rozdělení četností 22

2.4.1 Tabulky 22

2.4.2 Grafy 23

3 Aplikace vybraných charakteristik a grafů v R **29**

3.1 Příprava dat v R 30

3.2 Míry polohy v R 31

3.2.1 Analýza měr polohy kvantitativní spojité proměnné 31

3.2.2 Analýza měr polohy ordinální proměnné 32

3.2.3 Analýza měr polohy nominální proměnné 32

3.3 Míry variability v R 33

3.3.1 Analýza měr variability kvantitativní spojité proměnné 33

3.3.2 Analýza měr variability ordinální proměnné 34

3.3.3 Analýza měr variability nominální proměnné 35

3.4 Míry koncentrace v R 36

3.5 Rozdělení četností v R 36
3.6 Grafy v R 38
3.6.1 Histogram v R 38
3.6.2 Výsečový graf v R 41
3.6.3 Sloupcový graf v R 43
3.6.4 Spojnicový graf v R 44
3.6.5 Krabičkový graf v R 45
3.7 Výsledek analýzy 50
4 Závěr 51

Úvod

Pokud dostaneme data, je zapotřebí zjistit, jakého charakteru data jsou, dříve než se pustíme do hlubší práce s nimi. K tomu slouží popisná statistika. Je třeba zjistit základní informace, než začneme data analyzovat složitějšími metodami. A jelikož žijeme ve světě, který dynamicky roste, což se týká i světa technologií, je de facto nutností umět si práci ulehčit a umět pracovat s programy, které nám to umožní. Jedním takovým je právě software R. Dnes je velmi oblíbeným a stále a rychle se vyvíjejícím softwarem nejen v oblasti statistiky. Díky jeho dostupnosti roste i jeho popularita. Nevýhodou je, že v České republice zatím R není využíváno natolik hojně jako v zahraničí, což je důvodem nedostatku materiálů v českém jazyce pro práci s R. Motivací pro napsání této práce byl zmíněný nedostatek návodů pro různé oblasti statistiky.

Cílem této práce je přiblížit práci s R v oblasti popisné statistiky. Práce je rozdělena na tři části. První kapitola je teoretická a poukazuje na to, co to vlastně R je, jaké jsou jeho výhody, nevýhody a orientace v základních příkazech a chování R, bez kterých nelze práci s daty začít. Druhá kapitola je taktéž teoretická, vysvětluje, co je to popisná statistika, jaké jsou rozdíly mezi jednotlivými typy statistických proměnných a přístupu k nim. Uvádí funkce, popř. kód, pomocí kterého R provede požadovanou statistickou analýzu. Třetí kapitola je praktická, zde aplikuji příkazy v softwaru RStudio pro popisnou statistiku na reálná data. K dispozici je datový soubor, na kterém ilustruji vybrané a nejdůležitější příkazy popisné statistiky. Datový soubor se týká cen domů a bytů v San Franciscu v letech 2008-2009.

1 Programovací jazyk R a RStudio

R je programovací jazyk a prostředí specializované na statistiku, určené především pro statistickou analýzu dat a její grafické zobrazení. R je volně dostupný, šiřitelný, open source software. Zásadní rozdíl mezi R a některými jinými komerčními statistickými softwary, např. SPSS, je, že nedisponuje grafickým uživatelským rozhraním pro statistickou analýzu, nýbrž veškeré požadavky uživatel musí zadávat pomocí příkazů. Zde se projevuje jedna z obtíží, která může být překážkou, protože aby uživatel mohl příkazy zadávat, musí je i znát. Avšak existují softwary jako je RStudio, což je vývojové prostředí usnadňující práci s R. Obsahuje konzoli, editor podporující přímé spuštění kódu, nástroje pro kreslení grafů, historii a další.

R je dostupný pro širokou škálu operačních systémů, a to Microsoft Windows, macOS od Apple a Linux. R je součástí GNU projektu. R pracuje s balíčky, které jsou uloženy na tzv. Comprehensive R Archive Network (CRAN). V současné době (rok 2017) je na úložišti přes 10 000 balíčků volně dostupných ke stažení. Balíčky jsou sbírky funkcí, kódů a dat, které si uživatel navíc nainstaluje a tím rozšíří možnosti pro svou práci s R.

První kapitola popisuje základní práci s R a ovládání programu. Uvádím důležité příkazy a funkce, které jsou v práci použity i s vysvětlením jejich významu včetně návodu na tvorbu vlastní funkce. Celá práce s R je tvořena a popsána v prostředí RStudio.

1.1 Základní poznatky práce s R

Při spuštění programu RStudio má uživatel na výběr, zda chce příkazy psát přímo v konzoli či pracovat se skriptem, který se dá následně i uložit. Pokud se rozhodne pracovat v konzoli, každý řádek je uvozen znakem >, za kterým napíše příkaz a na dalším řádku (resp. řádcích) po stisknutí klávesy *Enter* R vypíše výsledek.

```
> 1+1
[1] 2
```

V případě práce se skriptem uživatel zadává příkazy na jednotlivé řádky a spouští je kombinací kláves *Ctrl + Enter*, případně ikonkou *Run*. Výsledek se ale opět zobrazí v konzoli hned za hranatými závorkami s číslem operace. Výhodami psaní do skriptu je možnost vkládat do něj komentáře, pokud uživatel chce práci různě členit na části či popsat, co daným kódem spouští, opakovaně spouštět kód znovu, ukládat a sdílet script a příslušnou analýzu s jinými

uživateli a další. Komentář se napíše za speciální znak mřížku (hash) a vše, co následuje za znakem, R ignoruje a je považováno za komentář, což i uživatel sám rozpozná odlišnou barvou textu.

```
# komentář
```

Jak bylo zmíněno výše, základní nabídku funkcí a dat je možné rozšířit prostřednictvím instalace balíčků pomocí příkazu ***install.packages()***, kde v závorkách jako argument figuruje název daného balíčku. Následně balíček načteme příkazem ***library()***, přičemž v závorkách opět uvedeme název nainstalovaného balíčku.

```
> install.packages("nazevbalicku")
> library(nazevbalicku)
```

Jakmile jsou balíčky s funkcemi načteny, uživateli již nic nebrání v práci začít. Z funkcí uvedených výše lze pochopit, že všechny funkce se uvádějí v podobě *název funkce* a za tím kulaté závorky s argumenty *(argument1, argument2, ...),* které jednotlivé funkce obsahují. Pokud si uživatel není jistý, jaké argumenty funkce obsahuje, R disponuje nápovědou. Uživateli stačí napsat jeden ze dvou následujících příkazů, pro vyhledání informací o dané funkci a v záložce *Help* se zobrazí popis funkce a nápověda s vysvětlením, co jednotlivé argumenty znamenají.

```
> ?názevfunkce
> help(názevfunkce)
```

1.2 Důležité funkce pro práci s daty

Při práci s daty v R je nezbytné znát základní příkazy a funkce, bez kterých se uživatel neobejde. Mnohdy se však stane, že nenalezneme vhodnou funkci, pomocí které bychom získali požadované výsledky, a proto si funkci musíme sami naprogramovat. Tuto problematiku řeší kapitola 1.2.2.

1.2.1 Použité příkazy a funkce

V práci je použito několik základních příkazů a funkcí, které se opakují a bez nichž se práce neobejde. Ilustraci příkazů uvádím v této kapitole. Detailnějším popisem práce s programem R se pak zabývá např. Norman Matloff ve své knize The art of R programming: tour of statistical software design, ze které jsem čerpala (Matloff, 2011).

Je důležité znát tvorbu vektorů, kterou provedeme např. pomocí příkazu níže, přičemž kombinací znaků „<-“ zajistíme uložení výstupu do proměnné s názvem vedoucím směrem od šipky.

```
> x <- c(1, 1, 2, 3, 4, 4, 5)
> x
[1] 1 1 2 3 4 4 5
```

Pokud chce uživatel vybrat prvek z nějaké pozice, zadá název proměnné a v hranatých závorkách uvede, který prvek chce vybrat, případně které prvky v řadě zapsané v podobě *od:do* chce vybrat.

```
> x[5]
[1] 4
> x[3:6]
[1] 2 3 4 4
```

Stejně tak se dají z vektoru hodnot vybrat konkrétní prvky podmíněné matematickými operátory jako >, >=, <, <=, ==. Například chce-li uživatel z vektoru *x* prvky větší než 2 včetně, zadá:

```
> x[x >= 2]
[1] 2 3
```

Pokud uživatel pracuje s datovým souborem (data framem), který má několik proměnných ve sloupcích, k příslušné proměnné může přistoupit například stylem `data$nazevpromenne`, kde znakem dolaru spojíme název datového souboru s názvem proměnné, aby R vědělo, odkud proměnná pochází.

Chce-li uživatel do datového souboru (data framu) přidat novou proměnnou, existuje postup, kterým toho docílit za předpokladu splnění podmínky, že nová proměnná bude stejného rozsahu jako proměnné ve stávajícím datovém souboru. V příkazovém řádku či skriptu uživatel uvede zápis v podobě `data$novapromenna = vypocet`, kde proměnná ‚vypocet‘ představuje novou proměnnou přidanou do datového souboru (data framu).

Následující výčet jsou funkce, které jsou v práci použité (detaily k těmto funkcím je možné nalézt v nápovědě v programu RStudio):

as.character() se používá ke kódování vektoru hodnot jakožto slovní (charakterové) proměnné,

as.factor() se používá ke kódování vektoru hodnot jakožto kategorií kategoriální proměnné,

c() vytvoří vektor hodnot, numerických i slovních,

cumsum() vrací vektor hodnot, které jsou kumulativním součtem,

data() načte datovou sadu nebo seznam z dostupných datových sad,

data.frame() vytváří datové rámce, soubor proměnných v podobě datové matice,

head() zobrazí prvních *n* řádků z datového souboru (data framu),

ifelse(), je známá funkce ‚když' (stejně jako v MS Excel), která má argumenty *test, yes, no.* Do argumentu *test* zapíšeme podmínku a podle toho, zda bude TRUE nebo FALSE se hodnoty přepíší podle hodnot z argumentů *yes* a *no*,

round() zaokrouhlí hodnoty na zadaný počet desetinných míst,

seq() vygeneruje pravidelnou sekvenci hodnot,

sort() seřadí vektor hodnot sestupně či vzestupně,

sum() vrací součet všech hodnot proměnné,

table() sestaví tabulku absolutních četností u každé kombinace úrovní jednotlivých faktorů,

tail() vrací poslední hodnotu/y vektoru, matice, datového souboru (data framu) aj.

1.2.2 Tvorba vlastní funkce v R

Běžně se v práci s R stane, že uživatel nenajde zabudovanou funkci pro výpočet libovolné veličiny. Tento problém lze vyřešit tvorbou vlastní funkce. Skládá se ze tří částí. První je pojmenování funkce a definice jejich argumentů, druhá je posloupnost příkazů, které jsou k vytvoření funkce potřeba, a třetí je návratová hodnota funkce (tj. hodnota, kterou funkce vrací).

```
> jménoFunkce = function(argument1, argument2, ...)
+ {posloupnost příkazů
+  ...
+  ...
+  return(výsledek)}
```

1.2.3 Načtení dat do R

Ještě před tím, než začneme práci s daty a máme k dispozici datový soubor různého formátu, je třeba jej načíst pomocí vhodné funkce. Pokud chceme importovat datový soubor z Excelu do R, použijeme funkci ***read.xlsx(),*** kterou obsahuje balíček **openxlsx.**

Důležitými argumenty jsou *xlsxFile* pro název datového souboru, *sheet* pro název listu, *startRow* pro číslo řádku, ve kterém data začínají a *colNames* jako logická hodnota, zda první řádek slouží jako pojmenování sloupců. Příkazy pro načtení dat do R uložím do proměnné s názvem ve tvaru *„datačíslo“*, přičemž každý z nich je tzv. data frame.

```
> data1 <- read.xlsx(file = "data1.xlsx", sheet = "data", startRow = 2, col
Names = T)
```

Je-li soubor s příponou *.csv*, postačí funkce ***read.table()***, kterou R již obsahuje v základním nastavení. Důležité argumenty jsou *file* pro název souboru, *sep* jako separátor jednotlivých sloupců, přičemž oddělovačem může být středník nebo čárka, *dec* pro oddělování desetinných míst (čárka nebo tečka), *header* pro logickou hodnotu indikující, zda soubor obsahuje názvy proměnných v prvním řádku a *row.names* slouží jako vektor názvů řádků.

Příkaz může mít následující podoby:

```
> data2 <- read.table(file = "data2.csv", sep = ",", dec = ".", header = T,
row.names = 1)
```

```
> data3 <- read.table(file = "data3.csv", sep = ";", dec = ",", header = T,
row.names = 1)
```

Do argumentu *file* byla v příkazech výše zadaná absolutní cesta, tedy název souboru z adresáře, který je nastaven jako výchozí. Druhou variantou je zadávání relativní cesty vedoucí k souboru, viz příkaz níže.

```
> data4 <- read.table(file = "C:/Users/UserName/Documents/data4.csv",
sep = ";", dec = ",", header = T, row.names = 1)
```

Je důležité mít správně nastaven pracovní adresář, odkud R načte požadovaná data. Ke zjištění, kde máme nastaven pracovní adresář, slouží funkce ***getwd()***, která zobrazí přesnou cestu k pracovnímu adresáři a funkce ***setwd()*** slouží ke změně pracovního adresáře, přičemž do argumentu se do uvozovek zadá přesná cesta vedoucí ke konečnému adresáři.

```
> getwd()
[1] "C:/Users/UserName/Documents"
> setwd("C:/Users/UserName/Desktop")
```

2 Popisná statistika

Popisná statistika, jak už z názvu vypovídá, se zabývá popisem dat. Snaží se stručně charakterizovat strukturu daného datového souboru. Data zpracovává ve formě tabulek a grafů a jako nástroj používá ukazatele, které se obecně pro popisnou statistiku užívají. Mezi tyto nástroje patří dvě základní kategorie měr: míry polohy a míry variability. Mezi ukazatele měr polohy patří např. průměr, modus a medián, mezi ukazatele měr variability rozptyl, směrodatná odchylka. Kromě měr polohy a variability se do popisné statistiky řadí i míry asymetrie a míry špičatosti.

2.1 Základní statistické pojmy

Mezi základní statistické pojmy se řadí pojem *statistická jednotka*, což jsou prvky, u kterých sledujeme hromadné jevy a procesy a jsou základními (elementárními) jednotkami statistického pozorování. Jsou to např. osoby, organizace, věci či události. Sledované vlastnosti statistických jednotek jsou nazývány *statistickými znaky*. U vlastností jednotek záleží na tom, co považujeme za statistickou jednotku a od toho se odvíjí, jaké vlastnosti ji charakterizují. Např. uvažujeme-li jako statistickou jednotku automobil, její charakteristické vlastnosti budou značka, stáří, výkon, cena, rok výroby, typ vozu, aj.

Jak je vidět z vlastností vyjmenovaných výše, každá je jiná. Cena se dá vyjádřit číslem, značka naopak slovem. Proto se statistické znaky třídí do různých skupin. Existuje několik způsobů klasifikace proměnných. Ve své práci uvádím členění, které používá prof. Ing. Hana Řezanková, CSc. ve své učebnici Úvod do statistiky (Řezanková a Löster, 2009). Hlavní členění je na kvantitativní a kvalitativní proměnné. Kvantitativní znaky lze vyjádřit číselně (cena, rok výroby, výkon), kvalitativní slovně (značka, typ vozu). Kvalitativní znaky se dále člení na ordinální a nominální. Ordinální nebo také pořadové proměnné nabývají hodnot, kdy jedna hodnota je větší než druhá a také je lze seřadit. Nelze ale o nich říci, o kolik ani kolikrát se jedna hodnota liší od druhé. Příkladem ordinální proměnné je stupeň dosaženého vzdělání, známkování, aj. Nominální proměnné jsou takové, o nichž lze pouze říci, zda jsou hodnoty stejné nebo jiné a nelze je uspořádat ani s nimi provádět aritmetické operace. Nominální proměnná je například barva vlasů, typ školy, aj.

Co se týče kvantitativních znaků, člení se na spojité a nespojité (také diskrétní) proměnné. Kvantitativní znaky jsou takové, se kterými lze provádět aritmetické operace. Nespojité proměnné nejčastěji nabývají celočíselných hodnot o malém počtu obměn jako je

počet dětí či členů v domácnosti. Spojité proměnné nabývají libovolných hodnot z oboru reálných čísel či z nějakého intervalu na reálné ose. Příkladem je výška, váha, příjem statistické jednotky.

Všechny tyto statistické znaky sledujeme u statistických jednotek, které souhrnně tvoří *statistický soubor*. Statistický soubor můžeme mít dvojího druhu – základní a výběrový. Základní soubor je statistický soubor všech jednotek, které sledujeme a o kterém provádíme závěry. Jeho rozsah může být příliš velký, např. celá populace země, proto se provádí výběrové šetření a tím získáváme výběrový soubor (Řezanková a Löster, 2009, Cyhelský a kol., 2009).

2.2 Odlehlé a extrémní hodnoty

Dříve než se kdokoli pustí do zkoumání rozdílů mezi jednotlivými typy proměnných a jejich charakteristiky, je třeba definovat, co jsou odlehlé a extrémní hodnoty. Ordinální a kvantitativní proměnné je třeba zkoumat nejen z hlediska měr polohy a variability, ale také z hlediska existence či neexistence velmi malých, resp. velmi velkých hodnot vzhledem ke tvaru rozdělení četností. Odlehlé a extrémní hodnoty mají špatný vliv na vypočtené statistiky. Mohou falešně snižovat, resp. zvyšovat hodnotu průměru jako míry polohy nebo hodnotu směrodatné odchylky jako míry rozptylu dat.

Existují dva přístupy stanovení odlehlých a extrémních hodnot. V případě symetrického rozdělení četností, zejména u normálního rozdělení[1] se v intervalu $\langle \bar{x} - s_x; \bar{x} + s_x \rangle$ nachází přibližně 68 % hodnot. V intervalu $\langle \bar{x} - 3s_x;\ \bar{x} + 3s_x \rangle$ se nachází 99,8 % hodnot a hodnoty nacházející se mimo tento interval jsou definovány jako extrémní. Na základě kvantilových měr se od sebe rozlišují odlehlé a extrémní hodnoty. Odlehlé hodnoty leží buď v intervalu $\langle \tilde{x}_{25} - 3(\tilde{x}_{75} - \tilde{x}_{25});\ \tilde{x}_{25} - 1{,}5(\tilde{x}_{75} - \tilde{x}_{25}) \rangle$ nebo v intervalu $\langle \tilde{x}_{75} + 1{,}5(\tilde{x}_{75} - \tilde{x}_{25});\ \tilde{x}_{75} + 3(\tilde{x}_{75} - \tilde{x}_{25}) \rangle$. Extrémní hodnoty jsou menší než $\tilde{x}_{25} - 3(\tilde{x}_{75} - \tilde{x}_{25})$ nebo větší než $\tilde{x}_{75} + 3(\tilde{x}_{75} - \tilde{x}_{25})$. (Řezanková a Löster, 2009)

[1] Normální (Gaussovo) rozdělení je jedno z nejdůležitějších rozdělení pravděpodobnosti spojité náhodné veličiny. Má zvonovitý tvar a parametry μ a σ^2. Za určitých podmínek dobře aproximuje řadu jiných pravděpodobnostních rozdělení.

2.3 Míry polohy a variability

Míry polohy charakterizují střed datového souboru, tedy hodnotu, kolem které se ostatní hodnoty nacházejí a kolísají. Jelikož data můžou být různého charakteru, různě proměnlivá, nestačí je popsat pouze střední hodnotou, tuto informaci musíme doplnit o její variabilitu. K tomu slouží míry variability, které hodnotí rozptýlenost dat kolem „středu“, tedy jak jsou si hodnoty navzájem blízké či vzdálené. Touto problematikou se ve své knize zabývají Cyhelský a Souček (2009), Řezanková a Löster (2009) a Souček (2006).

2.3.1 Nominální proměnná

U nominální proměnné jsme schopni o dvou hodnotách říci pouze to, zda jsou stejné či různé. Nelze s nimi provádět aritmetické operace. Je zapotřebí zjistit rozdělení četností (více o četnostech v kapitole 2.4), nejlépe ve formě tabulky a poté začít soubor charakterizovat pomocí vybraných měr.

Pokud bychom chtěli zavést nominální proměnnou v R, použitím funkce ***factor()*** toho docílíme. V případě, že chceme nějakou proměnnou převést na kategoriální, lze použít funkci ***as.factor()***, která se používá ke kódování vektoru hodnot jakožto kategorií kategoriální proměnné

a) Míry polohy

Pro nominální proměnnou je míra polohy charakterizována *modální kategorií* x_{Mo}. Je to kategorie s největší četností výskytu sledované vlastnosti. Sledovaná proměnná může mít jednu i více modálních kategorií. V případě, že modální kategorie bude mít relativní četnost větší než 0,5, označíme ji jako majoritní.

Modální kategorie se v R dá zjistit několika po sobě jdoucími příkazy. Pokud uživatele zajímá pouze relativní četnost modální kategorie, zadá následující příkazy:

```
> abs <- table(x)
> rel <- abs/sum(abs)
> pmo <- max(rel)
```

Pokud by chtěl vědět, jak četnost modální kategorie, tak název modální kategorie, následující dva příkazy to umožní:

```
> usporadani <- sort(rel)
> modalni_kat <- tail((usporadani), 1)
```

Nyní vysvětlení postupu v obou případech. Prvním krokem je vypočítat absolutní četnosti pomocí funkce ***table()***, které do argumentu *x* uživatel zadá název proměnné z datového souboru (data framu). Druhým krokem je zjištění relativních četností. V prvním případě již stačí zjistit maximální hodnotu relativních četností pomocí funkce ***max()***. Co se týče druhého případu, je třeba uspořádat relativní četnosti od nejmenší po největší pomocí funkce ***sort()*** a posledním krokem je zjištění výsledku pomocí funkce ***tail()***, která navrátí název modální kategorie i její počet pozorování.

V R není mezi základními funkcemi funkce k získání módu k dispozici, proto si ji uživatel musí sám vytvořit. Způsobů, jak spočítat modus v R je mnoho, uvádím však jeden z těch jednodušších.

```
> modus = function(x)
+ {mod <- table(as.factor(x))
+  modus <- names(mod)[mod == max(mod)]
+  return (modus)}
```

V prvním kroku je třeba funkci pojmenovat, zvolila jsem jednoduše název `modus`. Druhým krokem je sestavení „těla“ funkce, kde příkaz `table(as.factor(x))`, kterému je vhodné opět přiřadit název `mod`, spočítá, kolikrát se v souboru vyskytují jednotlivé kategorie proměnné *x*. Poté příkaz `names(mod)[mod == max(mod)]` zjistí „název“ (tj. hodnotu) z proměnné `mod`, kde `mod` bude maximální, což zajistí příkaz v hranatých závorkách. Opět příkazu přiřadím název `modus`, který uvedu do návratové hodnoty funkce, tj. do příkazu return.

b) Míry variability

Mezi ukazatele míry variability nominální proměnné patří *variační poměr, nominální rozptyl (nomvar)* a *entropie*. Variační poměr je rozdíl relativní četností modální kategorie od jedničky, neboli $v = 1 - p_{Mo}$. Nominální rozptyl vyjadřuje relativní počet dvojic jednotek, které se liší hodnotou u proměnné, kterou sledujeme. Počítá se dle vzorce

$$nomvar = 1 - \sum_{i=1}^{k} p_i^2 = \sum_{i=1}^{k} (p_i(1 - p_i)), \tag{2.1}$$

kde p_i pro $i = 1, 2, \ldots, k$ je relativní četnost sledované proměnné v *i*-té kategorii, *k* značí počet kategorií a $nomvar \in \langle 0; (k-1)/k \rangle$. Pro lepší interpretaci výsledné hodnoty je vhodné

použít *normalizovaný nominální rozptyl*, kde *nomvar* vynásobíme zlomkem *k/(k – 1)*, aby nabýval hodnoty z intervalu od 0 do 1. Poslední mírou variability je entropie *H*, kterou spočítáme podle vzorce

$$H = -\sum_{i=1}^{k} p_i \, ln \, p_i, \tag{2.2}$$

kde p_i pro $i = 1, 2, \ldots, k$ je relativní četnost sledované proměnné v *i*-té kategorii, *k* značí počet kategorií a platí, že $H \in \langle 0; \ln k \rangle$. Opět můžeme použít normalizovanou entropii, která vychází v intervalu $\langle 0, 1 \rangle$, kdy *H* vydělíme logaritmem z *k*, tedy $H^* = H/ln\ k$. V případě, že míra variability nabyde krajní hodnoty 0, hovoří se o nulovém rozptýlení, naopak čím vyšší hodnota, tím vyšší různorodost souboru.

Implementace měr variability nominální proměnné v R je popsána níže. Jelikož R nemá zabudovanou funkci pro výpočet variačního poměru, nominálního rozptylu a entropie, je třeba si funkce nadefinovat. Funkce jsou definovány následovně:

- Variační poměr

```
> variacni_pomer = function(x)
+ {abs <- table(x)
+  rel <- abs/sum(abs)
+  pmo <- max(rel)
+  return(1-pmo)}
```

Prvně je třeba vypočítat absolutní četnosti nominální proměnné *x* pomocí funkce ***table()*** a pojmenovat (abs). Poté tyto četnosti vypočítat v relativním vyjádření a pojmenovat (rel). Dále pomocí funkce ***max()*** zjistit modální (nejčetnější) hodnotu a pojmenovat (pmo) a v návratové hodnotě funkce uvést vzorec, podle kterého se variační poměr počítá.

- Nominální rozptyl

```
> nomvar = function(x)
+ {abs <- table(x)
+  rel <- abs/sum(abs)
+  return (1-sum(rel^2))}
```

Opět je třeba vypočítat absolutní četnosti nominální proměnné *x* pomocí funkce ***table()*** a pojmenovat, poté tyto četnosti vypočítat v relativním vyjádření, pojmenovat a v návratové hodnotě funkce uvést vzorec, podle kterého se nomvar počítá.

- Entropie

```
> entropie = function(x)
+ {abs <- table(x)
+  rel <- abs/sum(abs)
+  return(-sum(rel*log(rel, exp(1))))}
```

Také při definování funkce pro entropii je zapotřebí vypočítat absolutní četnosti, poté relativní četnosti a do návratové hodnoty funkce uvést vzorec, podle kterého se entropie počítá. V R se přirozený logaritmus *ln* vypočítá jako `log(x, base = exp(1))`, kde argument *base* je základ logaritmu.

Je třeba si dát pozor, jak je nominální proměnná uložena. Nominální proměnnou je případně potřeba kódovat jako charakterovou proměnnou pomocí funkce ***as.character()***, kde do argumentu *x* uživatel zadá název sloupce z datového souboru (data framu).

2.3.2 Ordinální proměnná

U ordinální proměnné jsme schopni určit pořadí hodnot. Nelze již však říci, o kolik se dvě hodnoty od sebe liší. Hodnotou ordinální proměnné může být číslo, text i datum. V R se ordinální proměnná zavede pomocí funkce ***ordered()***, která má argumenty *x* pro vektor hodnot, většinou malý počet rozdílných hodnot, *levels* pro volitelný vektor hodnot, kterých *x* může nabývat (výchozí nastavení je seřazení od nejmenší po největší hodnotu), *labels* buď pro volitelný vektor popisků pro hodnoty z argumentu *levels* nebo pro řetězec znaků o délce 1 a jiné.

a) Míry polohy

Stejně jako u nominální proměnné sledujeme *modální kategorii* x_{Mo}, která je nejčetnější kategorií sledované proměnné. Další mírou polohy je *mediánová kategorie* x_{Me}, neboli kategorie, pro kterou je kumulativní četnost 0,5 nebo vyšší, pokud kumulativní četnost předchozí kategorie byla menší než 0,5. Častěji se však používá *medián* $\tilde{x}$, který v případě ordinální proměnné vyžaduje očíslování kategorií. Medián je padesátiprocentní kvantil, který dle velikosti rozděluje uspořádaný soubor na dvě poloviny, tj. polovina hodnot je menší než medián a polovina hodnot větší. Pokud má soubor lichý počet hodnot, vezme se prostřední hodnota, pokud sudý, poté se zprůměrují dvě prostřední hodnoty.

Pro výpočet mediánu je v R zabudovaná funkce ***median()*** a jeho dva důležité argumenty jsou *x* pro hodnoty, z kterých chceme medián spočítat a logickou hodnotu *na.rm*, která indikuje, zda by z výpočtu měly být odstraněny chybějící pozorování.

```
> median(x, na.rm = T)
```

V případě ordinální proměnné je vhodnější vypočítat mediánovou kategorii, pomocí vlastní nadefinované funkce, neboť funkce ***median()*** v R funguje pouze pro kvantitativní proměnné, nikoliv pro kategoriální proměnné kódované slovně. Mediánová kategorie se v R zjistí následujícím způsobem:

```
> medianova_kat = function(x, prob = 0.5)
+ {abs <- table(x)
+  rel <- abs/sum(abs)
+  cumRel <- cumsum(rel)
+  return(cumRel[cumRel >= prob][1])}
```

Funkce má dva argumenty, *x* pro hodnoty, ze kterých chceme mediánovou kategorii zjistit a *prob = 0.5*, pro 50% kvantil neboli medián. První krok je vytvořit absolutní četnosti ordinální proměnné pomocí již známé funkce ***table()***, poté relativní četnosti, příkazem ***cumsum()*** se spočítají kumulativní četnosti proměnné, v tomto případě relativní kumulativní četnosti. V návratové hodnotě funkce je třeba vybrat pouze hodnoty kumulativní relativní četnosti větší rovny mediánu, přičemž příkazem `[1]` zajistíme navrácení pouze první hodnoty, která tuto podmínku splňuje. Funkce vrátí hodnotu mediánové kategorie i s jejím názvem.

Popsat polohu lze také pomocí *kvantilů*. Nejčastěji používanými jsou kvartily, které rozdělují soubor na 4 stejné části. *Dolní kvartil* $\tilde{x}_{25}$, který dělí soubor na 25 % hodnot menších a 75 % hodnot větších a *horní kvartil* $\tilde{x}_{75}$, který naopak dělí soubor na 75 % hodnot menších a 25 % hodnot větších. Padesátiprocentní kvantil $\tilde{x}_{50}$ neboli medián dělí soubor na 50 % hodnot nízkých a 50 % vysokých. Dále jsou to *decily*, které rozdělují uspořádaný soubor na 10 stejných částí a *percentily* na 100 stejných částí (Řezanková a Löster, 2009).

V R je pro výpočet kvantilů funkce ***quantile()***, která má jako argumenty v závorkách: *x* pro vektor hodnot, *probs* pro vektor kvantilů, které požadujeme spočítat, logickou hodnotu *na.rm*, zda chceme zahrnout chybějící pozorování. Argument *probs* můžeme zadat v podobě vektoru ***c()***, do kterého desetinným číslem zadáme, které kvantily požadujeme nebo v podobě sekvence pomocí funkce ***seq()***, která má argumenty *from* pro počáteční hodnotu, *to* pro konečnou hodnotu a *by* je přírůstek sekvence.

```
> quantile(x, probs, na.rm = T, ...)
```

V případě ordinální proměnné není vhodné tuto funkci použít, stejně jako funkci ***median()***. Při zjišťování kvantilů ordinální proměnné je lepší vytvořit vlastní funkci, která je definována podobně jako funkce pro mediánovou kategorii.

```
> quantileOrd = function(x, prob)
+ {abs <- table(x)
+  rel <- abs/sum(abs)
+  cumRel <- cumsum(rel)
+  return(cumRel[cumRel >= prob][1])}
```

Funkce má opět dva argumenty, *x* pro hodnoty, ze kterých chceme mediánovou kategorii zjistit a *prob* pro určení hodnoty kvantilu. První krok je vytvořit absolutní četnosti ordinální proměnné, z těch relativní četnosti a následně relativní kumulativní četnosti proměnné. V návratové hodnotě funkce je třeba vybrat pouze hodnoty kumulativní relativní četnosti větší rovny požadovanému kvantilu, přičemž příkaz `[1]` zajistí navrácení pouze první hodnoty, která tuto podmínku splňuje. Funkce vrátí hodnotu kategorie i s názvem této kategorie.

b) Míry variability

Jednoduchý způsob, jak hodnotit variabilitu ordinální proměnné je pomocí variačního rozpětí, tedy rozpětí mezi maximální a minimální hodnotou sledované proměnné neboli vzorec

$$R = x_{max} - x_{min}, \tag{2.3}$$

kde x_{max} je poslední hodnota zkoumané proměnné, která je určena podle pořadí a x_{min} je první hodnota zkoumané proměnné, která je určena podle pořadí.

Nevýhodou variačního rozpětí je v případě existence extrémních hodnot zkreslená představa o míře variability zkoumaného souboru. Druhý podobný způsob je mezikvartilové rozpětí neboli rozdíl mezi 75% kvantilem a 25% kvantilem, počítáno podle vzorce

$$R_Q = \tilde{x}_{75} - \tilde{x}_{25}. \tag{2.4}$$

Ordinální rozptyl dorvar, jinak také diskrétní ordinální variance vyjadřuje, o kolik míst v pořadí kategorií se liší všechny dvojice statistických hodnot. Počítá se podle vzorce

$$dorvar = 2\sum_{i=1}^{k-1}\left(P_i(1-P_i)\right), \tag{2.5}$$

kde P_i pro i = 1, 2, …, k-1 je kumulativní relativní četnost sledované proměnné a platí, že $dorvar \in \langle 0; \frac{k-1}{2} \rangle$. V případě vynásobení *dorvaru* zlomkem $\frac{2}{k-1}$ získáme *normalizovaný ordinální rozptyl*, který nabývá hodnot od 0 do 1.

V R není zabudovaná funkce pro variační rozpětí, mezikvartilové rozpětí ani ordinální rozptyl. Pro dorvar je funkci třeba nadefinovat, ale pro výpočet variačního a mezikvartilového rozpětí není zapotřebí složitého definování vlastních funkcí, stačí pár příkazů.

- Variační rozpětí

```
> tab <- sort(table(x))
> variacni_rozp = max(tab) - min(tab)
```

V případě variačního rozpětí stačí vytvořit tabulku absolutních četností a poté od sebe odečíst maximální a minimální hodnotu.

- Mezikvartilové rozpětí

```
> x75 <- quantileOrd(x, prob = 0.75)
> x25 <- quantileOrd(x, prob = 0.25)
> n <- sum(table(x))
> IQRord <- (x75 - x25)*n
```

V případě mezikvartilového rozpětí si stačí pomocí výše definované funkce `quartileOrd` vypočítat 75% kvantil a 25% kvantil. Hodnoty vychází v relativním vyjádření, proto je třeba zjistit rozsah proměnné v souboru (n) a poté vypočítat mezikvartilové rozpětí v absolutním vyjádření.

- Ordinální rozptyl

```
> dorvar = function(x)
+ {abs <- table(x)
+  rel <- abs/sum(abs)
+  Pi <- cumsum(rel)
+  return(2*sum(Pi*(1-Pi)))}
```

Nejdříve je třeba vypočítat absolutní a relativní četnosti, stejně jako u variačního poměru, nomvaru a entropie v kapitole 2.3.1. Dále pomocí funkce ***cumsum()*** pro kumulativní

součet vypočítat kumulativní relativní četnost P_i a do návratové hodnoty funkce zadat vzorec pro výpočet dorvaru.

2.3.3 Kvantitativní proměnná

Kvantitativní (numerická) proměnná je měřitelná, data lze seřadit a provádět s nimi aritmetické operace. Kvantitativní proměnné dělíme na diskrétní, které nabývají celočíselných hodnot a většinou malého počtu obměn hodnot a spojité, které nabývají hodnot z intervalu reálných čísel. Informace v této kapitole jsem čerpala z knih Hindls (2003) a Řezanková a Löster (2009).

a) Míry polohy

Stejně jako pro ordinální proměnnou i pro kvantitativní můžeme použít míry polohy, jako jsou kvantily. Další skupinou významných měr poloh jsou průměry. Prvním z nich je aritmetický průměr, který je nejznámější a nejvíce užívaný typ průměru. *Prostý aritmetický průměr* lze vypočítat dle vzorce

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i\,, \tag{2.6}$$

kde x_i pro i = 1, 2, …, n je i-tá napozorovaná hodnota. V případě velkých souborů, kde jsou hodnoty uspořádány do tabulky rozdělení četností, se aritmetický průměr upravuje do tvaru *váženého aritmetického průměru*, který je, jak už z názvu vypovídá, násoben váhami jednotlivých kategorií. Spočítá se podle vzorce

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{k} x_i\, n_i, \tag{2.7}$$

kde x_i pro i = 1, 2, …, k je hodnota i-té kategorie a n_i pro i = 1, 2, …, k je četnost (váha) i-té kategorie.

K výpočtu aritmetického průměru v R nám poslouží velmi jednoduchá funkce ***mean()***, kde jako argument uvedeme *x* pro data, ze kterých chceme průměr spočítat, případně argument *na.rm,* pokud chceme či nechceme započíst i chybějící hodnoty. Chceme-li průměr vypočítat bez zahrnutí krajních hodnot, tj. do výpočtu nezahrneme *p*-procent hodnot z každé strany, přidáme argument *trim*, do kterého zadáme hodnotu *p* od 0 do 0,5.

```
mean(x, na.rm = TRUE, trim, ...)
```

V případě, že je k dispozici tabulka rozdělení četností, je možné vypočítat i vážený aritmetický průměr pomocí funkce ***weighted.mean()***, kde argumenty funkce jsou stejné jako pro ***mean()***, ale navíc je přidán argument *w* pro váhy.

```
weighted.mean(x, w, na.rm = TRUE)
```

Velikou nevýhodou aritmetického průměru je, že může být značně zkreslen, pokud se v souboru vyskytují výrazně extrémní hodnoty. V takovém případě je lepší použít medián či modus, který bude více vypovídat o charakteristice sledované proměnné.

Existují případy, kdy použití aritmetického průměru není vhodné (např. při výpočtu průměrné rychlosti, koeficientů růstu a další). V takových případech se používá buď *harmonický*, nebo *geometrický průměr*. Harmonický průměr se spočítá jako celkový rozsah souboru vydělený sumou z převrácených hodnot všech pozorování. Prostý harmonický průměr se počítá podle vzorce

$$\bar{x}_H = \frac{n}{\sum_{i=1}^{n} \frac{1}{x_i}}, \tag{2.8}$$

kde n je rozsah souboru a x_i pro $i = 1, \ldots, n$ je i-tá napozorovaná hodnota. Nejčastější použití harmonického průměru je v případě, kdy jednotlivá pozorování jsou naměřena v jednotkách čas na výkon (např. rychlost auta).

Geometrický průměr se používá, pokud máme pozorování vyjádřeny jako řadu koeficientů růstu. Podmínkou použití geometrického průměru je, aby hodnoty byly nezáporné, neboť ze vzorce

$$\bar{x}_G = \sqrt[n]{x_1 \cdot x_2 \cdot \ldots \cdot x_n} = \sqrt[n]{\prod_{i=1}^{n} x_i}, \tag{2.9}$$

kde x_i pro $i = 1, 2, \ldots, n$ je napozorovaná hodnota i-tého pozorování, je patrné, že odmocninu ze záporného čísla nelze vypočítat. Analogicky stejně jako pro aritmetický průměr existuje i vážený harmonický a vážený geometrický průměr.

b) Míry variability

Pro kvantitativní proměnnou jsou nejvhodnější charakteristiky variability *variační rozpětí* (2.3), *kvartilové rozpětí* (2.4) a nejpoužívanější *rozptyl* a jemu blízká *směrodatná odchylka*. Rozptyl se spočítá jako

$$s_x^2 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n}, \tag{2.10}$$

kde x_i pro i = 1, …, n je označení pro i-tou napozorovanou hodnotu a n je celkový počet pozorování. Rozptyl je definován jako průměrná čtvercová odchylka hodnot od průměru. Je uváděn ve čtverečních jednotkách (Řezanková a Löster, 2009).

Pro lepší interpretaci se používají *směrodatná odchylka* a *variační koeficient.* Směrodatná odchylka se již uvádí ve stejných jednotkách jako sledovaná proměnná a spočítá se jako odmocnina z rozptylu neboli

$$s_x = \sqrt{s_x^2}. \tag{2.11}$$

Variační koeficient je relativní míra variability, je podílem směrodatné odchylky a průměru a je počítaný podle vzorce

$$V_x = \frac{s_x}{\bar{x}}, \tag{2.12}$$

přičemž V_x je bezrozměrné číslo a udává se v procentech po vynásobení stem. Variační koeficient má velikou výhodu, neboť díky němu lze porovnávat dva různorodé soubory. Což neplatí o rozptylu, pomocí něj totiž porovnávat variabilitu mezi dvěma datovými soubory nelze.

Kromě výše uvedených veličin existuje i ukazatel variability, který není příliš ovlivněn extrémními hodnotami, a to *mediánová absolutní odchylka*. Je definována vzorcem

$$MAD = medián(|x_i - \tilde{x}|), \tag{2.13}$$

kde x_i pro i = 1, 2, …, n je i-tá napozorovaná hodnota. Mediánová absolutní odchylka je medián absolutních odchylek hodnot od jejich mediánu. V R pro výpočet měr variability slouží následující funkce:

range() pro variační rozpětí, která zobrazí minimální a maximální hodnotu, ***IQR()*** pro mezikvartilové rozpětí, ***var()*** pro rozptyl, ***sd()*** pro směrodatnou odchylku, ***mad()*** pro mediánovou absolutní odchylku. Všechny výše zmíněné funkce mají argumenty *x* pro vektor dat a logickou hodnotu *na.rm* pro nezahrnutí chybějících pozorování. Variační koeficient nemá zabudovanou funkci, níže je uveden příkaz, pomocí kterého se nadefinuje vlastní funkce pro výpočet variačního koeficientu:

```
> varKoef = function(x, na.rm = T)
+ {if (na.rm)
+  x = na.omit(x)
+  return(sd(x)/mean(x))}
```

c) Míry koncentrace

Kromě měr poloh a variability sledujeme u kvantitativní proměnné též míry šikmosti a míry špičatosti, které slouží k hodnocení koncentrace. Šikmost je míra symetrie rozložení hodnot sledované proměnné, resp. míra asymetrie, neboť v případě šikmosti rovnající se nule platí vztah $\bar{x} = \tilde{x} = \hat{x}$. Pokud v souboru převažuje více vyšších hodnot než nižších, je zešikmení kladné (zprava zešikmené) a pravý konec je delší než levý. Hodnota šikmosti je větší než nula a průměr leží vpravo od mediánu, tzn. $\bar{x} > \tilde{x} > \hat{x}$. V případě většího počtu nižších hodnot, než vyšších je zešikmení záporné (zleva zešikmené) a levý konec je delší než pravý. Hodnota vychází záporná a platí, že průměr leží vlevo od mediánu, tedy vztah $\bar{x} < \tilde{x} < \hat{x}$.

Špičatost je míra, která indikuje, zda je rozložení hodnost ploché nebo špičaté. Jsou-li hodnoty koncentrované kolem středu, rozdělení je špičatější, naopak čím více hodnot vzdalujících se od středu, tím plošší je toto rozdělení (Řezanková a Löster, 2009, Mareš a kol., 2015).

Šikmost a špičatost se řadí do skupiny momentů, konkrétně se jedná o normované momenty. Šikmosti je definována jako následující normovaný moment třetího řádu

$$\alpha_x = \frac{\sum_{i=1}^{n} \left(\frac{x_i - \bar{x}}{s_x}\right)^3}{n}, \tag{2.14}$$

kde x_i při $i = 1, \ldots, n$ je sledovaná i-tá hodnota, s_x je směrodatná odchylka a n je celkový počet pozorování. Špičatost je definován jako následující normovaný moment čtvrtého řádu

$$\beta_x = \frac{\sum_{i=1}^{n} \left(\frac{x_i - \bar{x}}{s_x}\right)^4}{n}, \tag{2.15}$$

přičemž značení je totožné jako u šikmosti. Kromě špičatosti se často uvádí také její modifikovaná verze koeficient špičatosti, kdy se tvar rozdělení porovnává s tzv. normálním rozdělením. Od výsledku špičatosti se odečte hodnota 3, tj. $\Upsilon_x = \beta_x - 3$.

R v základní instalaci nemá funkce pro výpočet šikmosti a špičatosti. Obsahuje je ale balíček s názvem *moments* (Lukasz Komsta a Frederick Novomestky, 2015), kde pro šikmost je funkce ***skewness()*** a pro špičatost ***kurtosis()***. Obě funkce mají dva argumenty, a to *x* pro data a *na.rm* pro nezahrnutí chybějících pozorování.

2.4 Rozdělení četností

Četnost udává, kolik se vyskytuje hodnot sledovaného znaku ve statistickém souboru. Rozlišujeme absolutní a relativní četnosti. Absolutní četnosti vyjadřují počet hodnot spadajících do jednotlivých kategorií či intervalů, relativní četnosti vyjadřují procento hodnot ležících v dané kategorii či intervalu, neboli podíly na celkovém počtu hodnot.

V případě spojité proměnné nebo nespojité proměnné nabývající vysokého počtu obměn hodnot je vhodné tvořit intervalové rozdělení četností. Tím se rozumí shrnutí určitého počtu blízkých hodnot do jednoho intervalu a uvedení četností vždy pro celý interval. Hodnoty z jednoho intervalu se považují za totožné a zastupují se středem intervalu.

Nejlepší formou znázornění rozdělení četností je tabulka a graf. Zjištěné hodnoty pozorované proměnné poskytuje rozdělení četností jednotlivých kategorií (např. v případě kvalitativních proměnných) či intervalové rozdělení četností (např. v případě spojitých kvantitativních proměnných).

2.4.1 Tabulky

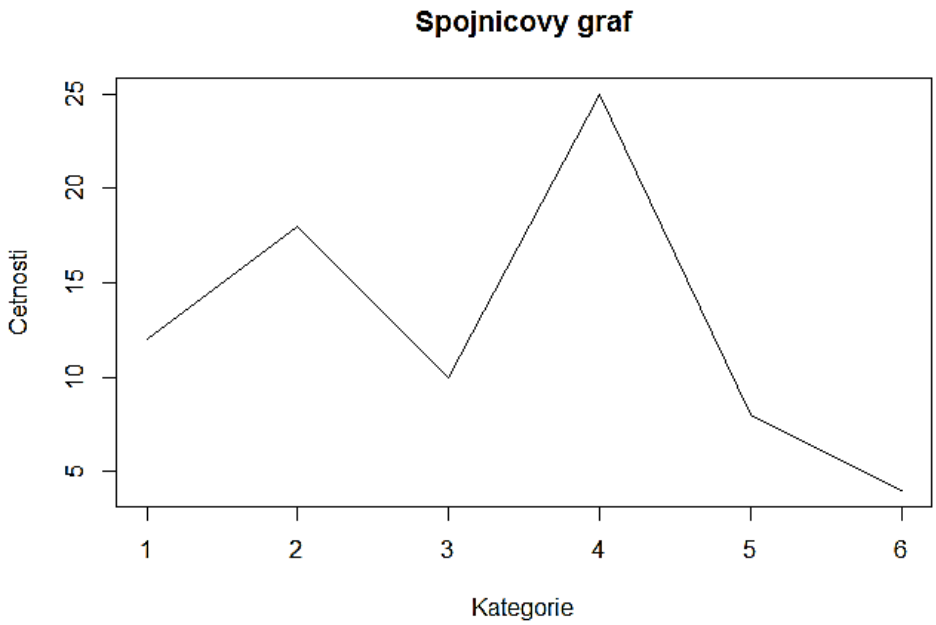
V tabulkách rozdělení četností nalezneme sloupec absolutních četností, které udávají počet naměřených hodnot dané kategorie nějaké kvantitativní proměnné a sloupec relativních četností vyjadřující podíl počtu výskytu sledované kategorie na celém souboru. Často se také v tabulkách rozdělení četností vyskytují sloupce s kumulativní absolutní a kumulativní relativní četností. Jsou to součty všech hodnot absolutních (resp. relativních) četností předcházejících a daného řádku, ve kterém nás tato hodnota zajímá.

2.4.2 Grafy

Podle typu proměnné rozlišujeme i různé typy grafického zobrazení. Grafy jsou výborným prostředkem pro zpřehlednění zjištěných dat. Níže je uvedeno pět nejznámějších typů, rozčleněných podle typu proměnné. Ke každému je uveden příkaz, kterým lze graf v R zkonstruovat.

1. Spojnicový graf

Graf prostého rozdělení četností kvantitativní spojité proměnné se nejčastěji znázorňuje pomocí polygonu četností. Vzniká spojením bodů úsečkami o souřadnicích (x_i, n_i) pro $i = 1, 2, \ldots, k$, kde x_i jsou hodnoty sledovaného znaku a n_i jejich odpovídající absolutní četnosti. Spojnicový graf je na obrázku 2.1.



Obrázek 2.1: Spojnicový graf

V R je pro spojnicový graf funkce ***plot()*** s argumenty *x* pro obměny hodnot sledovaného znaku, *y* pro absolutní četnosti sledované proměnné, *type* pro typ zakreslení (např. *p* pro body, *l* pro křivku, *b* pro obojí), *xlab* a *ylab* pro popisek osy x a y, *ylim* pro délku osy y a *main* pro název grafu. Pokud si uživatel předem vypočítá tabulku četností, kterou vhodně pojmenuje, stačí ji zadat místo argumentů x a y.

```
> plot(x, y, type, xlab, ylab, ylim, main, ...)
```

2. Sloupcový graf

Pro kategoriální proměnné se používá především graf sloupcový, kde výška sloupce představuje absolutní četnost sledovaného znaku pro danou kategorii. Používá se také v případě intervalového rozdělení četností pro nespojitou proměnnou, která má větší počet variant hodnot. V případě intervalového rozdělení spojité proměnné existuje speciální typ sloupcového grafu histogram, který bude popsán níže. Sloupcový graf je na obrázku 2.2.

Obrázek 2.2: Sloupcový graf

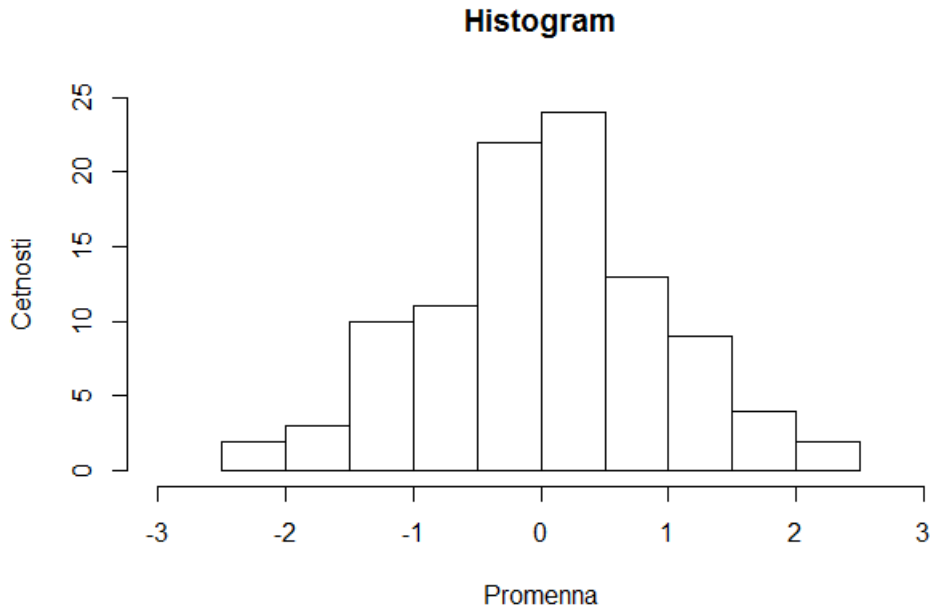
Pro zobrazení sloupcového grafu v R nám poslouží funkce ***barplot()***, jejíž nejpoužívanější argumenty jsou *height* pro vektor hodnot tvořících jednotlivé sloupce grafu (absolutní četnosti jednotlivých kategorií), *names.arg* pro vektor jmen, které mají být vyneseny pod každým sloupcem, *horiz* je logická hodnota, která určí, zda graf bude v horizontální či vertikální poloze, *col* pro barvu, *xlab* a *ylab* jako popisky osy x a y a *main* pro název grafu. Pokud si uživatel vytvoří tabulku absolutních četností a vhodně ji pojmenuje, stačí ji do funkce zadat místo argumentů *height* a *names.arg*, neboť oba argumenty jsou v tabulce již obsaženy.

```
> barplot(height, names.arg, horiz, col, xlab, ylab, main, ...)
```

3. Histogram četností

Speciální typ sloupcového grafu je histogram, který slouží k vizualizaci rozdělení kvantitativní proměnné. Používá se především pro intervalové rozdělení četností, přičemž jednotlivé sloupce se navzájem dotýkají a mají stejnou šířku.

Výška histogramu určuje četnost sledované veličiny. Z histogramu lze vyčíst typ rozdělení a jeho variabilitu. Také může sloužit k detekci odlehlých či extrémních hodnot a s nimi spojené asymetrie rozdělení. K tomu však slouží jiný typ grafu, a to krabičkový, který bude popsán v této kapitole později. Histogram je na obrázku 2.3.



Obrázek 2.3: Histogram

Volba pravidla pro volbu počtu intervalů (šířky intervalu) histogramu závisí na tvaru pravděpodobnostního rozdělení. Nejznámější pravidla jsou:

- Sturgesovo pravidlo, které počet intervalů určí jako

$$k \approx 1 + 3{,}3 * \log n, \tag{2.16}$$

kde k je počet intervalů a n je rozsah souboru neboli počet statistických jednotek v souboru. Sturgesovo pravidlo není vhodné, jestliže data nepochází přibližně z normálního rozdělení. Není také vhodné pro datové soubory o velkém rozsahu.

- Scottovo pravidlo, kde volba šíře intervalu se určí jako

$$h = \frac{3{,}5 * S_x}{\sqrt[3]{n}}, \tag{2.17}$$

kde h je šířka intervalu, S_x je výběrová směrodatná odchylka a n je rozsah souboru. Počet intervalů se následně vypočítá jako

$$k = \left\lceil \frac{max_x - min_x}{h} \right\rceil, \tag{2.18}$$

kde max_x je maximální napozorovaná hodnota, min_x je minimální napozorovaná hodnota a k je počet intervalů a výsledná hodnota se zaokrouhlí směrem nahoru. Stejně jako Sturgesovo pravidlo i Scottovo pravidlo je vhodnější pro výběry z přibližně normálního rozdělení.

- Freedman-Diaconisovo pravidlo stanovuje šíři intervalu podle vzorce

$$h = 2\frac{IQR(x)}{\sqrt[3]{n}}, \tag{2.19}$$

kde h je šíře intervalu, $IQR(x)$ je mezikvartilové rozpětí proměnné x a n je rozsah souboru. Je robustní vůči odlehlým či extrémním hodnotám, avšak také předpokládá normalitu proměnné.

Histogram se v R vytvoří pomocí funkce ***hist()***. Důležité argumenty jsou *x* pro vektor hodnot, *breaks* pro zvolené pravidlo šíře intervalů, *col* a *border* pro barvu sloupce a ohraničení, *xlab* a *ylab* pro popis názvů os a *main* jako hlavní název grafu.

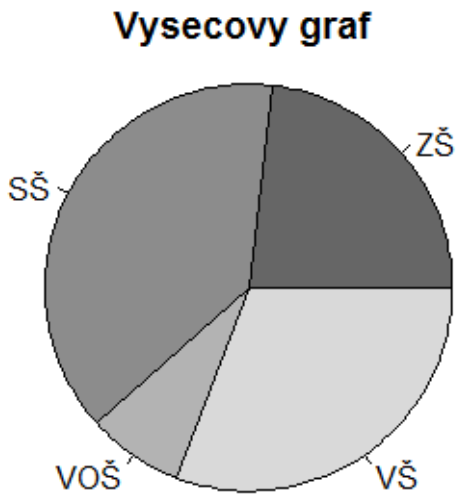
```
> hist(x, breaks, col, border, xlab, ylab, main, ...)
```

4. Výsečový graf

Nominální proměnnou lze graficky zobrazit jak sloupcovým grafem, tak i výsečovým. Jednotlivé výseče znázorňují relativní četnosti sledovaného znaku. Výsečový graf reprezentuje obrázek 2.4.

V R je pro výsečový graf zabudovaná funkce ***pie()*** s argumenty *x* pro nezáporná data, což mohou být i absolutní i relativní četnosti, z kterých chceme graf vytvořit, *labels* rovnající se funkci ***names(x)*** pro pojmenování jednotlivých výsečí a další.

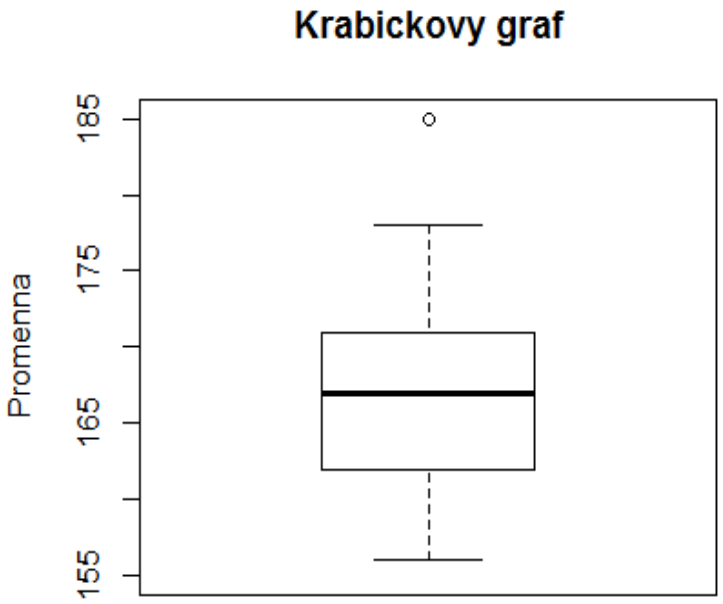
```
> pie(x, labels = names(x), ...)
```



Obrázek 2.4: Výsečový graf

5. Krabičkový graf

Krabičkový graf je velmi oblíbeným typem grafu pro kvantitativní proměnnou, neboť poskytuje široké spektrum informací. Skládá se z krabičky a „vousů“. Zjišťujeme z něj minimální a maximální napozorovanou hodnotu, což jsou v případě vertikální orientace a zároveň neexistence odlehlé či extrémní hodnoty, horní a dolní vousy. Krabička udává více informací. Spodní hrana krabičky prezentuje dolní kvartil a horní hrana horní kvartil, tudíž celá krabička představuje 50 % hodnot. Zbylých 50 % je rozděleno na 25 % hodnot mezi spodním vousem a dolní hranou krabičky a 25 % mezi horní hranou krabičky a horním vousem. Medián je reprezentován úsečkou uvnitř krabičky. Krabičkový graf je zkonstruován na obrázku 2.5.



Obrázek 2.5: Krabičkový graf

Použití krabičkového grafu slouží k detekci odlehlých či extrémních pozorování, posouzení variability a tvaru rozdělení, zda se vyskytuje asymetrie nebo zda rozdělení nemá tzv. tlusté konce[2] a k porovnání hodnot dané proměnné při třídění do skupin.

V R je pro krabičkový graf zabudována funkce ***boxplot()***. Existují dva způsoby, jakými graficky znázornit informace popisné statistiky pomocí krabičkového grafu.

I. Pokud máme spojitou proměnnou pouze pro jednu skupinu hodnot, zadáme argumenty následovně: *x* jsou hodnoty proměnné, ze kterých má být graf sestaven, *outline* slouží jako logická hodnota, zda mají být zahrnuty odlehlá pozorování, *border* pro barvu ohraničení krabičky a *col* pro barvu krabičky, *horizontal* určující, zda bude krabička v horizontální či vertikální poloze a další.

```
> boxplot(x, outline, border, col, horizontal, main, ...)
```

II. V případě, že proměnnou chceme ilustrovat do skupin podle jiné proměnné, je třeba využít opět funkce ***boxplot()*** s rozdílným zápisem argumentů a to:

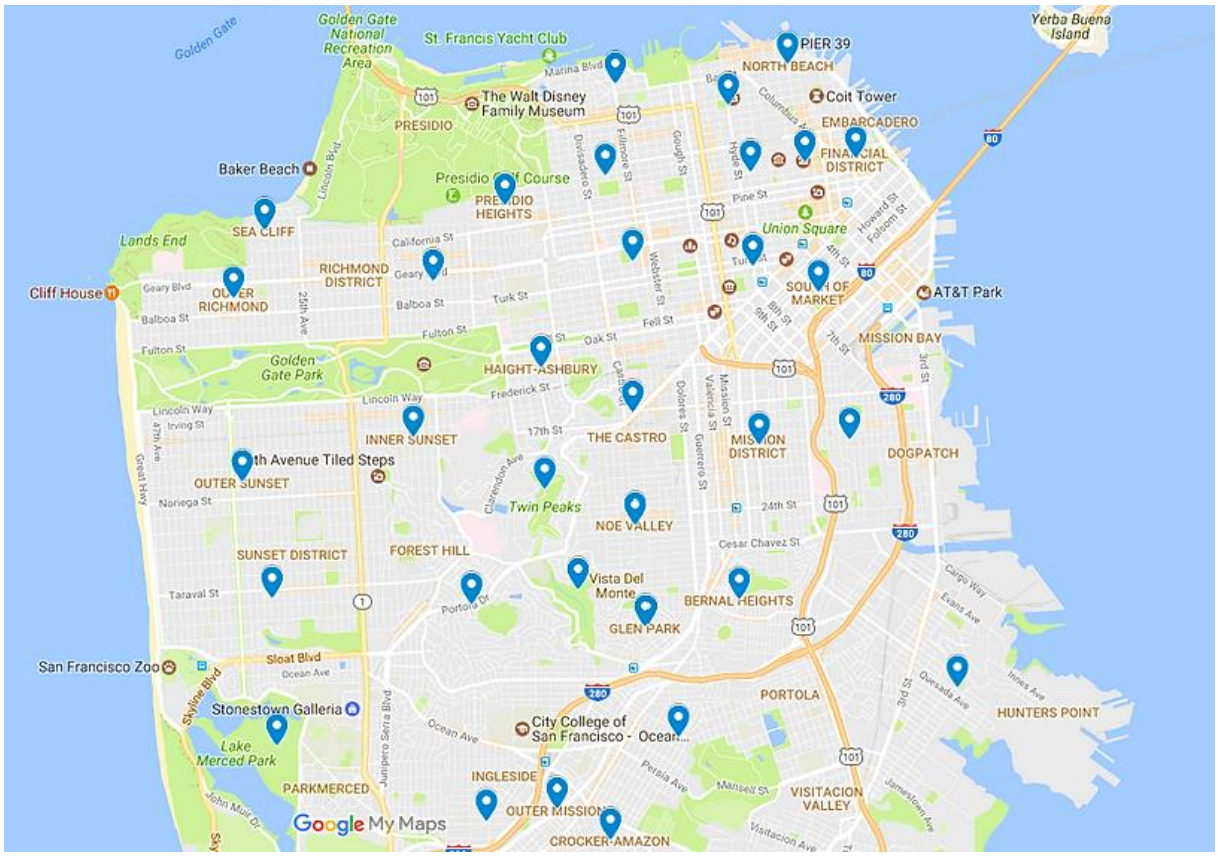
formula, což je vzorec x ~ y, kde x je číselný vektor datových hodnot, které mají být rozděleny do skupin podle kategoriální proměnné y. Dále argument *data*, do kterého uvedeme název datového souboru (data framu). Stejně jako v jiných typech grafu lze přidat argumenty *xlab*, *ylab* a *main* pro popisky os a název grafu, zadávané v uvozovkách. Argumenty *ylim* a *xlim* omezí délku os x a y a v důsledku toho „usekneme“ graf tam, kde chceme. Ostatní argumenty jsou stejné jako v předchozím případě.

```
> boxplot(x ~ y, data, border, col, xlab, ylab, main, ylim, xlim, outline)
```

[2] Rozdělení s tlustými konci (anglicky heavy-tailed distribution) je označení pro rozdělení, které má tendenci mít vysoký počet odlehlých či extrémních hodnot.

3 Aplikace vybraných charakteristik a grafů v R

Pro aplikaci popisné statistiky v R jsem zvolila datový soubor, který nese název San Francisco Home Sales. Data byla zjišťována v období od 13.2.2008 do 14.7.2009. Datový soubor obsahuje 3281 pozorování pro 15 proměnných a je ke stažení v R balíčku *nutshell* (Joseph Adler, 2012). Datový soubor nese informace o bytech a domech v San Franciscu, jejich cenách, velikosti rozlohy, počtu pokojů a další. Na obrázku 3.1 lze vidět mapu jednotlivých čtvrtí v San Franciscu.



Obrázek 3.1: Mapa nemovitostí datového souboru San Francisco Home Sales

Celá praktická část je rozdělena do sedmi částí. V první části je uvedeno, jak připravit data v R a jak provést jejich potřebné úpravy. Další části popisují, jak aplikovat míry polohy v R pro vybrané tři typy proměnných, míry variability v R a míry koncentrace v R s průběžně komentovanými výsledky. Nezanedbatelnou částí je rozdělení četností a tvorba grafů v R pro jednotlivé typy proměnných. V závěru je vyhodnocení výsledku analýzy popisné statistiky. Cílem této kapitoly je uvést možnosti a výsledky implementace popisné statistiky na datech v R a charakterizovat data týkající se nemovitostí v San Franciscu. Znalosti, co se týče popisné statistiky, byly částečně čerpány z knihy Petera Dalgaarda (2008) a částečně ze softwaru RStudio (R Core Team, 2016).

3.1 Příprava dat v R

Prvním krokem je instalace balíčku *nutshell*. Příkazem, který byl definován v kapitole 1.1 balíček nainstalujeme a poté načteme.

```
> install.packages("nutshell")
> library(nutshell)
```

Balíček *nutshell* disponuje více soubory, pro účely této práce jsem zvolila datový soubor San Francisco Home Sales, který načtu příkazem ***data()***.

```
> data(sanfrancisco.home.sales)
```

Zjistím, zda datový soubor obsahuje chybějící hodnoty. K tomu použiji funkci ***complete.cases()***, kde se do závorky zadá sekvence vektorů hodnot, matice hodnot, případně datové soubory (data framy). Výsledkem je logický vektor hodnot indikující, které pozorování nebo řádky neobsahují chybějící hodnoty v celé sekvenci. Ukázka příkazu a prvních dvou řádků výsledku pro datový soubor je uvedena níže:

```
> complete.cases(sanfrancisco.home.sales)
  [1] FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
  [8]  TRUE FALSE  TRUE  TRUE  TRUE FALSE FALSE
```

Pro práci s daty zvolím přístup odstranění chybějících hodnot a vytvoření nového datového souboru, kde již budou jen záznamy s kompletními údaji. Pomocí funkce ***na.omit()*** s argumentem *object*, což je typicky datový soubor (data frame), odstraním ze souboru sanfrancicso.home.sales chybějící hodnoty a vytvořím nový bez nich, který uložím pod názvem ‚datasf'.

```
> datasf <- na.omit(sanfrancisco.home.sales)
```

Pracovat budu pouze s několika proměnnými. Zvolila jsem tři různé typy proměnných: *price* (cenu) pro analýzu spojité kvantitativní proměnné, *bedrooms* (pokoje) pro analýzu ordinální proměnné a *neighborhood* (čtvrť) pro analýzu nominální proměnné. K náhledu výběru těchto proměnných použiji funkci ***data.frame()***, do které zadám jen ty sloupce, které mě zajímají a funkcí ***head()*** zobrazím prvních pět řádků pro představu. Příkaz si pojmenuju ‚vyber'.

```
> vyber <- data.frame(datasf$price, datasf$bedrooms, datasf$neighborhood)
```

```
> head(vyber, 5)
  datasf.price datasf.bedrooms datasf.neighborhood
1       385000               0   Visitacion Valley
2      2200000               3             Mission
3       565000               2      Bernal Heights
4       520000               5             Bayview
5       363000               3             Bayview
```

3.2 Míry polohy v R

Následující kapitola se zabývá aplikací měr poloh v R. Aplikaci provedu zvlášť pro tři výše zmíněné proměnné postupně, tj. na spojité proměnné *price* (cena), na ordinální proměnné *bedrooms* (pokoje) a na kvalitativní proměnné *neighborhood* (čtvrť).

3.2.1 Analýza měr polohy kvantitativní spojité proměnné

Průměr pro proměnnou, která nemá chybějící hodnoty, se spočítá pomocí funkce ***mean()*** pouze s argumentem pro data. Pokud by proměnná disponovala chybějícími hodnotami, přidal by se argument *na.rm*. Vše, co se týče funkce ***mean()*** je popsáno v kapitole 2.3.3.

```
> mean(datasf$price)
[1] 976741.4
```

Druhá velmi důležitá míra polohy je medián, která se v R spočítá pomocí funkce ***median()***. Pokud porovnáme výsledek průměru a mediánu, vidíme, že průměr je značně vyšší než medián. Důvodem je existence odlehlých či extrémních hodnot, které průměr značně zkreslují.

```
> median(datasf$price)
[1] 800000
```

Podíváme-li se na výsledné hodnoty blíže, zjistíme, že pro proměnnou *price* (cena) platí vztah $\bar{x} > \tilde{x}$, tedy 976 741, 4 > 800 000. O této proměnné lze říci, že je zprava zešikmená a pravý konec jejího rozdělení je delší než levý, tj. v souboru se vyskytuje větší počet vyšších hodnot, které jsou odlehlé.

Pro výpočet kvantilů lze uvést následující tři způsoby výpočtů, podle toho, zda nás zajímá jeden určitý kvartil, řada kvartilů nebo decily. Pro výpočet řady kvartilů využijeme příkazu ***c()*** a pro decily funkce ***seq()***, obojí v argumentu *probs*.

```
> quantile(datasf$price, probs = 0.25)
   25%
600000
```

```
> quantile(datasf$price, probs = c(0.25, 0.5, 0.75))
    25%     50%     75%
 600000  800000 1131250

> quantile(datasf$price, probs = seq(0.1, 0.9, 0.1))
    10%     20%     30%     40%     50%     60%     70%     80%     90%
 470000  550000  639500  715000  800000  885000 1041000 1250000 1620000
```

3.2.2 Analýza měr polohy ordinální proměnné

Pro charakteristiku měr poloh ordinální proměnné se určuje mediánová kategorie a modální kategorie. Mediánová kategorie se v R spočítá pomocí vlastní nadefinované funkce z kapitoly 2.3.1.

```
> medianova_kat(datasf$bedrooms)
        2
0.6690544
```

Mediánová kategorie v případě proměnné *bedrooms* (pokoje) je kategorie 2-pokojových nemovitostí s kumulativní četností 66,9 %. Tedy 0-, 1- a 2-pokojové nemovitosti soubor zastupují z 2/3.

Modální kategorie se v R zjistí pomocí několika po sobě jdoucích jednoduchých příkazů, které jsou definovány v kapitole 2.3.1

```
> abs <- table(datasf$bedrooms)
> rel <- abs/sum(abs)
> usporadani <- sort(rel)
> modalni_kat <- tail((usporadani), 1)
> modalni_kat
        0
0.4641834
```

Nejčetnější kategorie proměnné *bedrooms* (pokoje) je tedy kategorie 0-pokojových nemovitostí s 46,4% zastoupením, což je skoro polovina celého souboru.

3.2.3 Analýza měr polohy nominální proměnné

Modální kategorie nominální proměnné se zjišťuje stejně jako u ordinální proměnné:

```
> abs <- table(datasf$neighborhood)
> rel <- abs/sum(abs)
> usporadani <- sort(rel)
> modalni_kat <- tail((usporadani), 1)
> modalni_kat
Outer Sunset
  0.07449857
```

Nejčetnější kategorií proměnné *neighborhood* (čtvrť) je čtvrť se jménem Outer Sunset s relativní četností 7,4 % z celého souboru.

3.3 Míry variability v R

Kapitola pojednává o aplikaci měr variability v R. K charakteristice měr opět použiji proměnnou *price* (cena) pro analýzu kategoriální spojité proměnné, proměnnou *bedrooms* (pokoje) pro analýzu ordinální proměnné a *neighborhood* (čtvrť) pro nominální proměnnou. Jak je známo z kapitoly 2.3.3, charakterizovat soubor pouze pomocí měr poloh nemusí být dostačující, neboť může být zapotřebí znát i rozptýlenost napozorovaných hodnot v souboru.

3.3.1 Analýza měr variability kvantitativní spojité proměnné

Pokud nás zajímá minimální a maximální hodnota, lze použít následující funkce:

```
> min(datasf$price)
[1] 100000
> max(datasf$price)
[1] 9500000
```

Tyto dvě veličiny v sobě zahrnuje funkce ***range()***, která vypočítá rozpětí proměnné.

```
> range(datasf$price)
[1]  100000 9500000
```

Mezikvartilové rozpětí spočítá funkce ***IQR()***.

```
> IQR(datasf$price)
[1] 531250
```

Rozptyl a směrodatnou odchylku vypočítá funkce ***var()*** a ***sd()***.

```
> var(datasf$price)
[1] 5.0535e+11
> sd(datasf$price)
[1] 710879.7
```

Variační koeficient se spočítá pomocí vlastní funkce, která má v kapitole 2.3.3 uvedený příkaz pro její vytvoření:

```
> varCoef(datasf$price)
[1] 0.7278075
```

Variační koeficient proměnné *price* (cena) je 72,78 %, tzn. směrodatná odchylka se na aritmetickém průměru podílí přibližně ze 73 %. Variační koeficient stejně jako směrodatná odchylka indikují vysokou variabilitu proměnné *price* (cena) v datovém souboru.

Z důvodu vysoké variability a existence odlehlých až extrémních hodnot je vhodné použít mediánovou absolutní odchylku, což je robustní míra variability pro kvantitativní proměnnou:

```
> mad(datasf$price)
[1] 363978.3
```

Medián absolutních odchylek hodnot proměnné *price* (cena) od jejich mediánu, který vyšel 800 000, vychází 363 978,3, což je podstatně méně, neboť MAD je odolný vůči odlehlým hodnotám.

Chce-li uživatel rychlé shrnutí měr poloh, stačí zadat funkci ***summary()***, která vypočítá minimum, 25% kvartil, medián, průměr, 75% kvartil a maximum.

```
> summary(datasf$price)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 100000  600000  800000  976700 1131000 9500000
```

3.3.2 Analýza měr variability ordinální proměnné

Jedním ze způsobů hodnocení variability ordinální proměnné je pomocí variačního a mezikvartilového rozpětí, které se vypočítá následovně:

```
> tab <- table(datasf$bedrooms)
> variacni_rozp <- max(tab) - min(tab)
> variacni_rozp
[1] 647
> x75 <- quantileOrd(datasf$bedrooms, prob = 0.75)
> x25 <- quantileOrd(datasf$bedrooms, prob = 0.25)
> n <- sum(table(datasf$bedrooms))
> IQRord <- (x75 - x25)*n
> IQRord
  3
544
```

Hodnota variačního rozpětí vychází 647, mezikvartilového rozpětí vychází nižší, tj. 544, neboť mezikvartilové rozpětí není tolik ovlivněno odlehlými či extrémními hodnotami. Větší vypovídací schopnost má ovšem ordinální rozptyl, jehož funkce je definována v kapitole 2.3.2.

```
> dorvar <- dorvar(datasf$bedrooms)
[1] 1.852843
```

Pro lepší interpretaci rozptylu ordinální proměnné, který vyšel 1,853, je třeba vypočítat normovaný dorvar, který se spočítá vynásobením dorvaru zlomkem $2/(k–1)$. Počet kategorií zjistíme následujícím příkazem, přičemž funkce ***dim()*** vrací velikost rozměrů matice hodnot.

```
> k <- dim(table(datasf$bedrooms))
> k
[1] 14
```

Normovaný ordinální rozptyl poté spočítáme následovně:

```
> norm.dorvar <- 2*dorvar/(k-1)
> norm.dorvar
[1] 0.2850528
```

Variabilita ordinální proměnné *bedrooms* (pokoje) je 0,285, což značí nízkou variabilitu.

3.3.3 Analýza měr variability nominální proměnné

```
> variacni_pomer(datasf$neighborhood)
[1] 0.9255014
```

```
> nomvar <- nomvar(datasf$neighborhood)
[1] 0.9545437
```

Nominální rozptyl je třeba převést na normalizovaný nominální rozptyl, tj. nomvar vynásobit zlomkem $k/(k-1)$ *a* převést hodnotu od 0 do 1 a teprve poté jej interpretovat. Počet kategorií se zjistí stejným příkazem jako u dorvaru.

```
> k <- dim(table(datasf$neighborhood))
> k
[1] 34
```

```
> norm.nomvar <- nomvar*k/(k-1)
[1] 0.9834692
```

Normovaný nominální rozptyl proměnné *neighborhood* (čtvrť) vyšel 0,983. Nomvar říká, že čím více se hodnota blíží jedné, tím méně jsou data koncentrována a jsou rovnoměrněji rozložena do jednotlivých kategorií.

```
> entropie <- entropie(datasf$neighborhood)
[1] 3.223288
```

Stejně tak jako nomvar i entropii je třeba normalizovat, abychom mohli interpretovat její výsledek. Vydělením entropie přirozeným logaritmem z počtu kategorií získáme normalizovanou entropii:

```
> norm.entropie <- entropie/log(k, exp(1))
[1] 0.9140551
```

Normovaná entropie vyšla 0,914. Všechny tři míry variability vyšly sobě navzájem velmi blízké, tedy nad hodnotu 0,9, což značí vysokou variabilitu proměnné *neighborhood* (čtvrť) v souboru.

3.4 Míry koncentrace v R

Pro výpočet měr koncentrace v R je potřeba instalace balíčku *moments*. Balíček obsahuje funkci pro šikmost a špičatost. Míry koncentrace ilustruji na kvantitativní spojité proměnné *price* (cena).

```
> skewness(datasf$price)
[1] 4.557053

> kurtosis(datasf$price)
[1] 38.21369
```

Šikmost pro proměnnou *price* (cena) vyšla 4,557 což potvrzuje tvrzení z kapitoly 3.2 o zprava zešikmených datech a v souboru se vyskytuje větší počet odlehlých hodnot. Špičatost má hodnotu 38,21, což v případě kladné špičatosti indikuje rozložení blízké střední hodnotě, tedy většina pozorování je nahuštěna okolo střední hodnoty a jen málo pozorování je odlehlých. Koeficient špičatosti poté vychází 38,21 – 3 = 35,21.

3.5 Rozdělení četností v R

V R není zabudovaná funkce, pro výpočet rozdělení četností. Lze jej ovšem vytvořit pomocí použití několika po sobě následujících funkcí. Rozdělení četností ilustruji na proměnné *bedrooms* (pokoje), kvůli nedostatku nominálních proměnných volím právě tuto proměnnou.

Funkce ***table()*** využívá uspořádání počtů všech hladin faktorů sledované proměnné. Tabulku četností vytvoříme pro proměnnou *bedrooms* (pokoje). Jako argument funkce zadáme název proměnné v podobě *datasf$bedrooms* a přiřadíme jí název *tab*. Získáme řádek četností jednotlivých počtů pokojů.

```
> tab <- table(datasf$bedrooms)
> tab

  0   1   2   3   4   5   6   7   8   9  10  12  14  18
648  26 260 258 149  32   8   6   4   1   1   1   1   1
```

Výpočet absolutních četností převedeme do formy tabulky pomocí funkce ***as.data.frame()*** a přiřadíme název *tab_cet*. Funkcí ***head()*** zobrazíme první 4 záznamy.

```
> tab_cet <- as.data.frame(tab)
> head(tab_cet, 4)
  Var1 Freq
1    0  648
2    1   26
3    2  260
4    3  258
```

Do tabulky *tab_cet* přidáme třetí sloupec s relativními četnostmi. Jakým způsobem se přidává sloupec do již existujícího datového souboru je uvedeno v kapitole 1.2.1 věnující se příkazům a funkcím v R.

```
> tab_cet$RelFreq = round(tab_cet$Freq/sum(tab_cet$Freq)*100, 3)
```

Relativní četnost *RelFreq* vypočítáme jako absolutní četnost *Freq* podělenou součtem z absolutních četností vynásobenou stem. Výsledné relativní četnosti pomocí funkce ***round()*** s argumentem *x* pro vektor hodnot a *digits* pro počet desetinných míst, zaokrouhlím na 3 desetinná místa. Výslednou tabulku absolutních a relativních četností proměnné *bedrooms* (pokoje) ilustruje obrázek 3.2.

	Var1	Freq	RelFreq
1	0	648	46.418
2	1	26	1.862
3	2	260	18.625
4	3	258	18.481
5	4	149	10.673
6	5	32	2.292
7	6	8	0.573
8	7	6	0.430
9	8	4	0.287
10	9	1	0.072
11	10	1	0.072
12	12	1	0.072
13	14	1	0.072
14	18	1	0.072

Obrázek 3.2: Tabulka rozdělení četností

Největší četnost má kategorie pro 0-pokojové nemovitosti se 648 pozorováními a 46,4% zastoupením v celém souboru. Druhá nejčetnější kategorie jsou 2-pokojové nemovitosti s 260 pozorováními a 18,6% zastoupením. Poté 3- a 4-pokojové nemovitosti a dále u vícepokojových nemovitostí četnost postupně klesá.

3.6 Grafy v R

Z kapitoly 2.4.2 je již známo, které typy grafů používáme pro jednotlivé typy proměnných a známe jejich zápis kódu v R. V následujících podkapitolách bude představena aplikace grafického zobrazení dat v R. Nadpisy grafů a popisky os budou bez diakritiky.

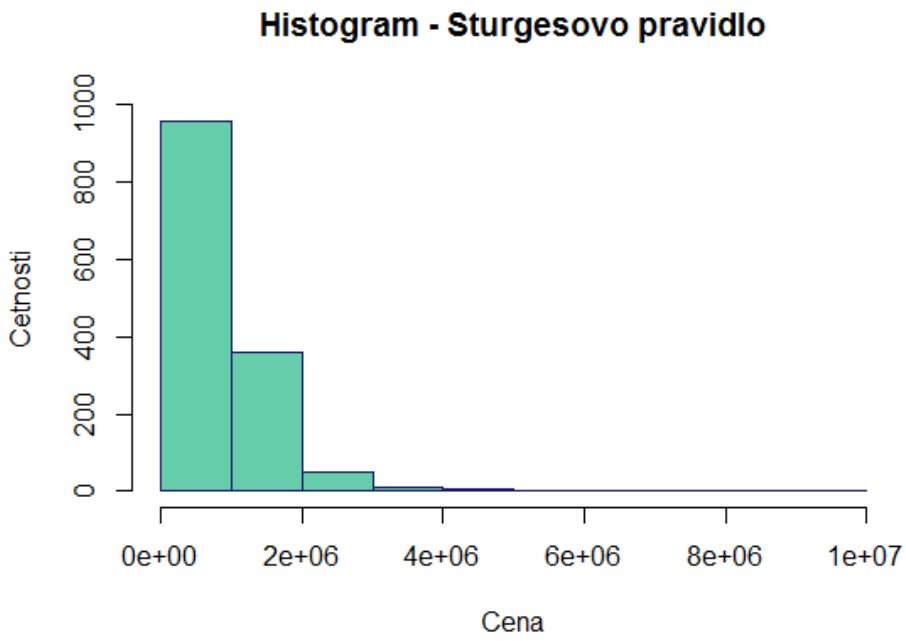
3.6.1 Histogram v R

Aplikace histogramu v R bude ilustrována na proměnné *price* (cena). Budou předvedena tři pravidla, která známe pro určení vhodné šíře jednotlivých sloupců, a to Sturgesovo, Scottovo a Freedman-Diaconisovo pravidlo. U nejvhodnějšího z nich ilustruji histogram s odlehlými hodnotami i bez odlehlých hodnot.

- Sturgesovo pravidlo

Funkce histogram má v R defaultně přednastaveno Sturgesovo pravidlo pro šíři intervalů. Zadáním prostého příkazu ***hist(datasf$price)*** sice získáme histogram, je ale třeba jej ještě upravit. Argumentem *xlim* a *ylim* protáhneme osy x a y tak, aby jednotlivé sloupce nebyly delší než stanovená délka os. Přidáme názvy os a název grafu a upravíme barvu sloupců. Výsledný graf popisuje kód níže a ilustruje obrázek 3.3.

```
> hist_Sturges <- hist(datasf$price, breaks = "Sturges", ylim = c(0, 1000),
xlim = c(0, 10000000), col = "aquamarine3", border = "midnightblue", main =
"Histogram - Sturgesovo pravidlo", xlab = "Cena", ylab = "Cetnosti")
```



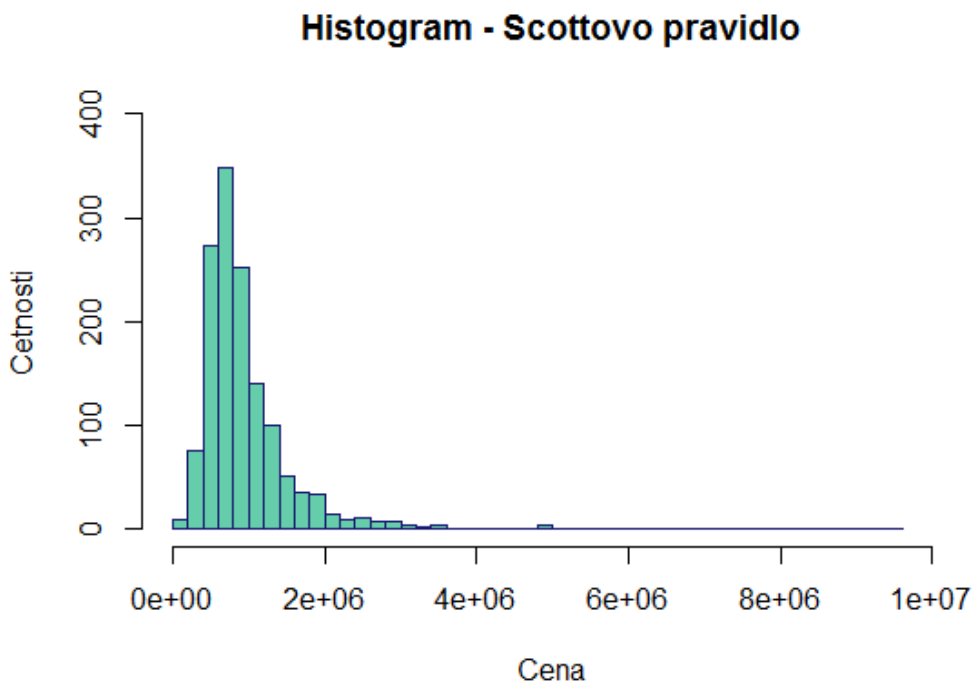
Obrázek 3.3: Histogram v R - Sturgesovo pravidlo

I přes veškeré úpravy ale vidíme, že histogram podle Sturgesova pravidla není to, co bychom očekávali. Nejsme schopni rozlišit, zda se v proměnné *price* (cena) vyskytují odlehlé hodnoty, neboť je v grafu nevidíme. Z kapitoly 2.4.2 již víme, že Sturgesovo pravidlo není vhodné pro soubory o velkém počtu pozorování.

- Scottovo pravidlo

Vhodnější pravidlo pro konstrukci histogramu je Scottovo, které ilustruji níže na obrázku 3.4. I přesto, že histogram vypadá značně lépe, Scottovo pravidlo není robustní vůči odlehlým hodnotám, kterých je v souboru mnoho.

```
> hist_Scott  <- hist(datasf$price, breaks = "Scott", ylim = c(0, 400), xlim
= c(0, 10000000), col = "aquamarine3", border = "midnightblue", main =
"Histogram - Scottovo pravidlo", xlab = "Cena", ylab = "Cetnosti")
```

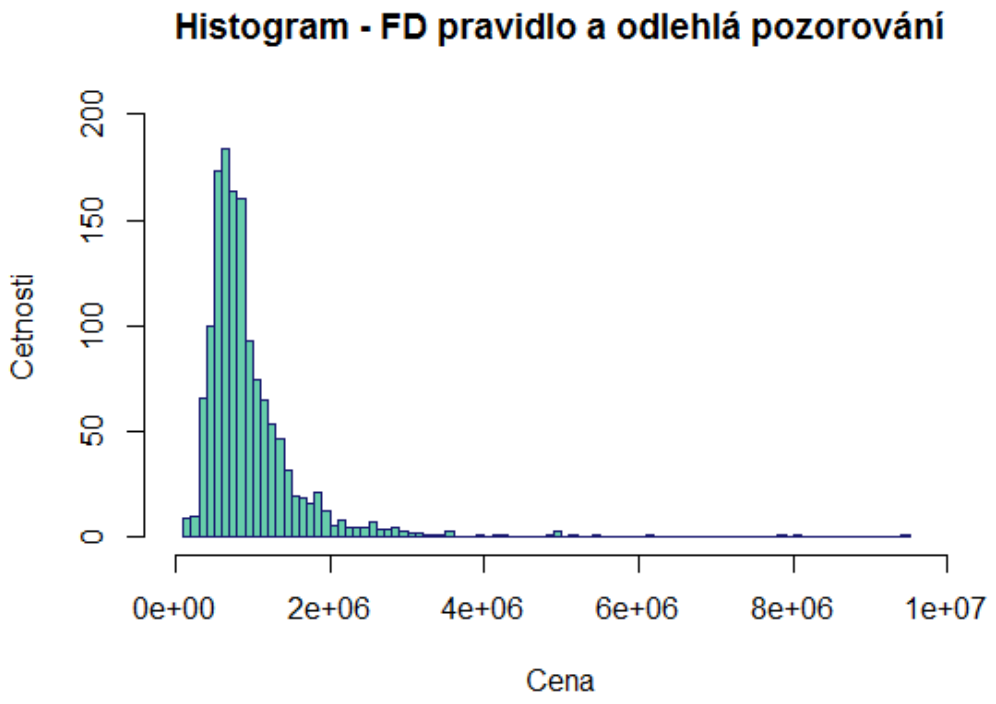


Obrázek 3.4: Histogram v R – Scottovo pravidlo

- Freedman-Diaconisovo pravidlo

Nejvhodnější je Freedman-Diaconisovo pravidlo, které ze tří uvedených je jako jediné robustní vůči odlehlým hodnotám a ze kterého lze nejlépe zkoumat rozdělení hodnot. Viz obrázek 3.5.

```
> hist_FD_s <- hist(datasf$price, breaks = "Freedman-Diaconis", ylim = c(0, 200), xlim = c(0, 10000000), col = "aquamarine3", border = "midnightblue", main = "Histogram - FD pravidlo a odlehlá pozorování", xlab = "Cena", ylab = "Cetnosti")
```



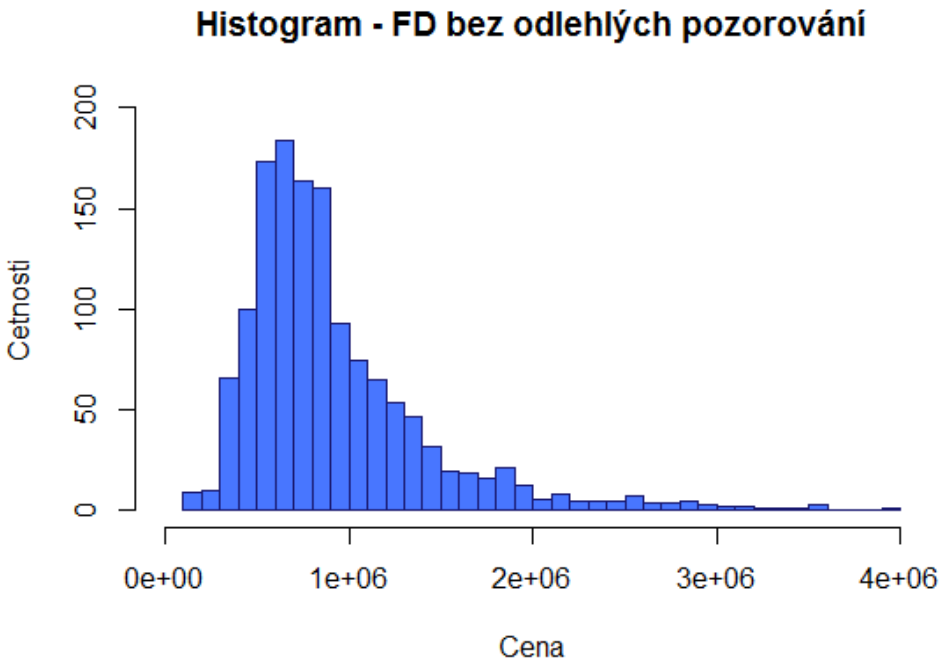
Obrázek 3.5: Histogram v R – FD pravidlo a odlehlá pozorování

Graf je ilustrován i s odlehlými hodnotami, kvůli kterým není tolik přehledný a je obtížnější z něj vyčíst konkrétnější hodnoty. Tyto odlehlé hodnoty je možné „useknout“ v určité délce a zobrazit graf bez nich. V kódu se tato podmínka zapíše v argumentu x, kde místo `datasf$price` zadám `datasf[datasf$price <= 4000000, „price“]`.

V překladu nová podmínka znamená, že z datového souboru *datasf* chci vybrat proměnné ze sloupce *datasf$price*, které jsou menší rovny 4 000 000, pro sloupec *„price“*. Kvůli takovému omezení je zapotřebí také upravit graf z vizuální stránky, tedy osu x zkrátit do 4 000 000 a přiřadit nový název grafu, případně změnit barvu.

```
> hist_FD_bez <- hist(datasf[datasf$price <= 4000000, "price"], breaks = "Freedman-Diaconis", ylim = c(0, 200), xlim = c(0, 4000000), col = "royalblue1", border = "midnightblue", main = "Histogram - FD bez odlehlych pozorovani", xlab = "Cena", ylab = "Cetnosti")
```

Graf s použitím Freedman-Diaconisova pravidla je ilustrován na obrázku 3.6.



Obrázek 3.6: Histogram v R – FD bez odlehlých hodnot

Nyní stručná interpretace histogramu pro proměnnou *price* (cena): Histogram vyjadřuje četnost jednotlivých hodnot pro zvolenou proměnnou. Nejčetnější je cena okolo 600 000, přibližně pro 180 pozorování. Vyšší ceny se postupně snižují, jak lze vidět ze všech čtyř typů histogramů uvedených výše.

3.6.2 Výsečový graf v R

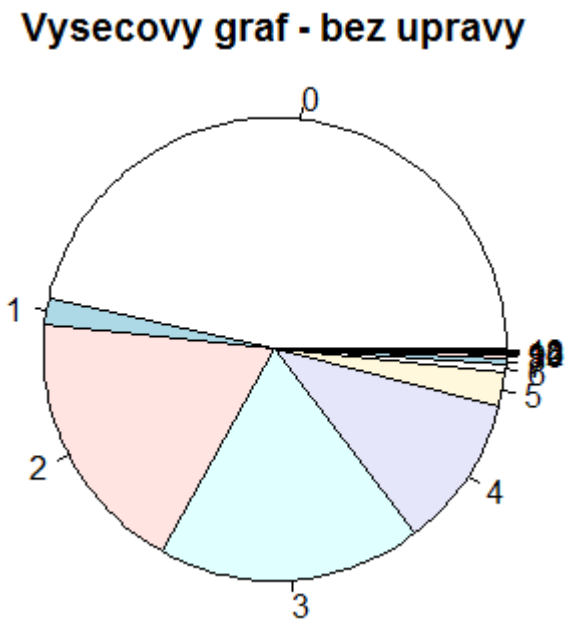
Pokud uživatel zkoumá kategoriální proměnnou a chce pro ni zkonstruovat výsečový graf v R, je nejprve třeba vytvořit absolutní četnosti proměnné, pro kterou graf požadujeme. Pro analýzu kategoriální proměnné a možnosti ukázky konstrukce výsečového grafu v R jsem zvolila proměnnou *bedrooms* (pokoje). Tato proměnná v praktické části slouží jako ordinální proměnná, ale kvůli nedostatku kvalitativních proměnných v souboru jsem pro potřeby konstrukce výsečového grafu použila tuto proměnnou. Pomocí funkce ***table()*** se vytvoří absolutní četnosti jednotlivých obměn.

```
> tab <- table(datasf$bedrooms)

  0   1   2   3   4   5   6   7   8   9  10  12  14  18
648  26 260 258 149  32   8   6   4   1   1   1   1   1
```

Poté se výpočet aplikuje na výsečový graf, kde data budou výše vytvořené absolutní četnosti. Graf vytvořený pomocí kódu níže je na obrázku 3.7.

```
> pie(tab, main = "Vysecovy graf - bez upravy")
```



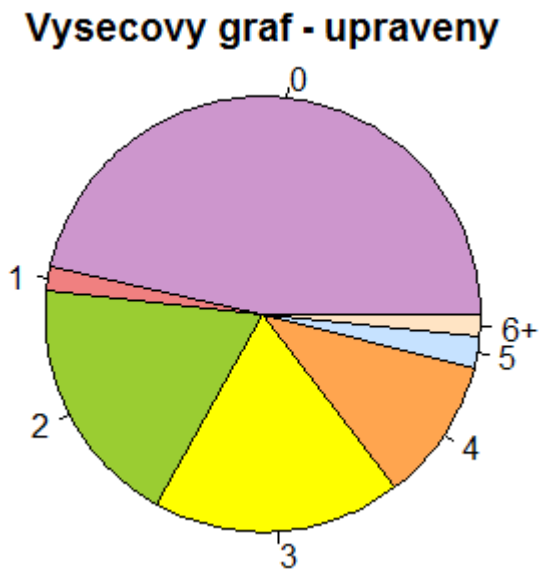
Obrázek 3.7: Výsečový graf v R - bez úpravy

Podíváme-li se pozorně na absolutní četnosti jednotlivých pozorování, zjistíme, že vyšší hodnoty mají nižší četnosti, a proto je graf nepřehledný. Vyřešit tento problém můžeme, pokud sloučíme dohromady četnosti pro počet pokojů 6 a výše do jedné skupiny „6+“. Dále je nespočetně mnoho možností na změnu barvy, stačí si vyhledat, jaké barvy jsou v R možné, např. na internetu a podle toho barvy jednotlivých výsečí změnit.

```
> bedrooms2 <- ifelse(datasf$bedrooms >= 6, "6+", datasf$bedrooms)
> pokoje_upr <- table(bedrooms2)
> pokoje_upr
bedrooms2
  0   1   2   3   4   5  6+
648  26 260 258 149  32  23
```

Nyní již můžeme vytvořit výsečový graf, kde nová proměnná *pokoje_upr*, kterou jsme pro potřeby výsečového grafu vytvořili, figuruje jako argument *x*. Graf ilustruje obrázek 3.8.

```
> pie(x = pokoje_upr, col = c("plum3", "lightcoral", "olivedrab3", "yellow",
"tan1", "slategray1", "bisque"), main = "Vysecovy graf - upraveny")
```

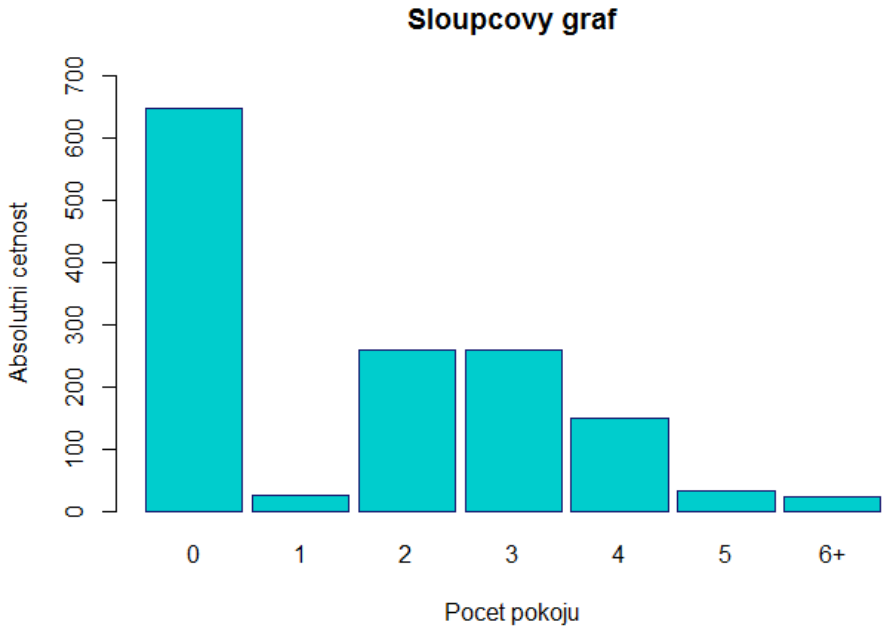


Obrázek 3.8: Výsečový graf v R - upravený

3.6.3 Sloupcový graf v R

U sloupcového grafu se postupuje obdobně jako u výsečového, tedy je třeba znát absolutní četnosti proměnné *bedrooms* (pokoje), které jsme v R zjistili pomocí příkazů v kapitole 3.6.2. Většina argumentů v kódu níže je uživateli známá již z předchozích typů grafů. Navíc je argument *space*, který určí šířku mezer mezi jednotlivými sloupci. Viz obrázek 3.9.

```
> barplot(pokoje_upr,  xlab = "Pocet pokoju", ylab = "Absolutni cetnost",
main = "Sloupcovy graf", border = "midnightblue", col = "cyan3", space = 0.1,
ylim = c(0, 700))
```



Obrázek 3.9: Sloupcový graf v R

Ze sloupcového grafu lze opět vidět, že 0-pokojové nemovitosti jsou zastoupeny nejčetněji, s více než 600 pozorováními. Následují je 2- a 3-pokojové nemovitosti, které mají podobnou četnost.

3.6.4 Spojnicový graf v R

Spojnicový graf se v R tvoří pomocí funkce ***plot()***. Pro zobrazení spojnicového grafu jsem zvolila proměnnou *bedrooms* (pokoje) a obdobný způsob zobrazení dat, kdy nižší četnosti sjednotím do jedné skupiny. Následujícím příkazem, který jsem použila již pro výsečový a sloupcový graf, sjednotím pokoje pro počet 9 a vyšší do jedné skupiny.

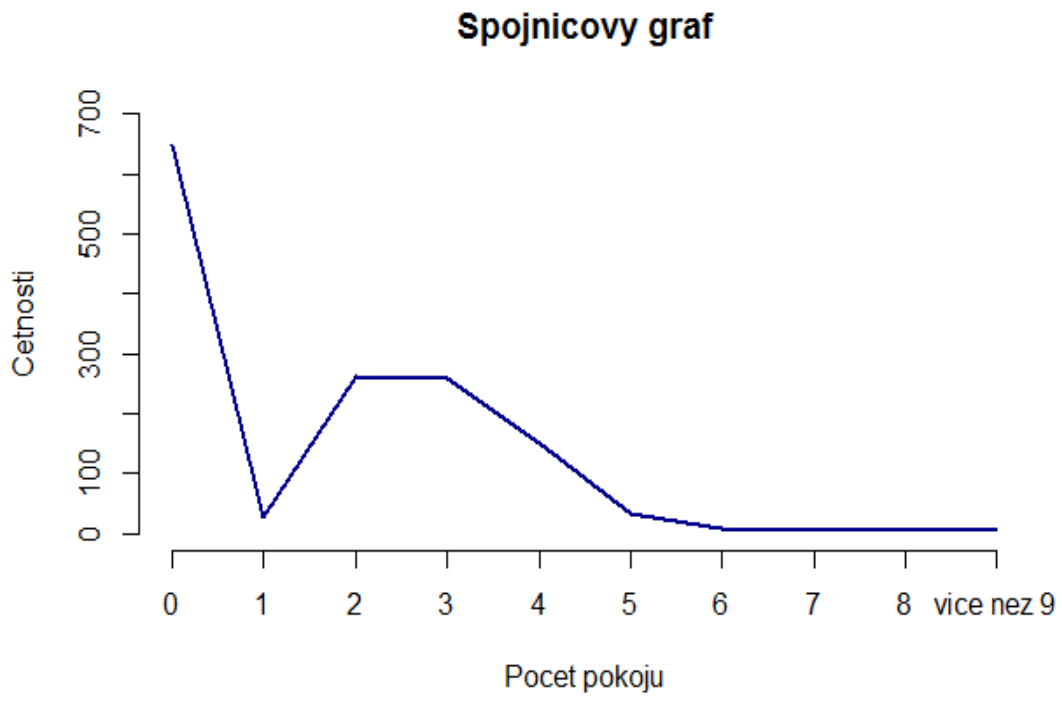
```
> pokoje_sluc <- ifelse(datasf$bedrooms >= 9, "vice nez 9", datasf$bedrooms)
```

Poté pro nové hodnoty vytvořím tabulku absolutních četností již známým příkazem ***table().***

```
> pokoje_tab <- table(upr_pokoje)
```

Nyní do funkce pro graf zadám nově vytvořenou tabulku četností a do argumentu *type* zadám písmeno „l“ pro vykreslení spojnicové čáry. Graf je zobrazen na obrázku 3.10.

```
> plot(pokoje_tab, type = "l", xlab = "Pocet pokoju", ylab = "Cetnosti", ylim
= c(0, 700), main = "Spojnicovy graf", col = "darkblue")
```



Obrázek 3.10: Spojnicový graf v R

Z grafu lze vyčíst, že největší četnost mají 0-pokojové nemovitosti, skoro 700 pozorování, poté 2- a 3-pokojové s necelými 300 pozorování a od 4-pokojových výše četnost klesá.

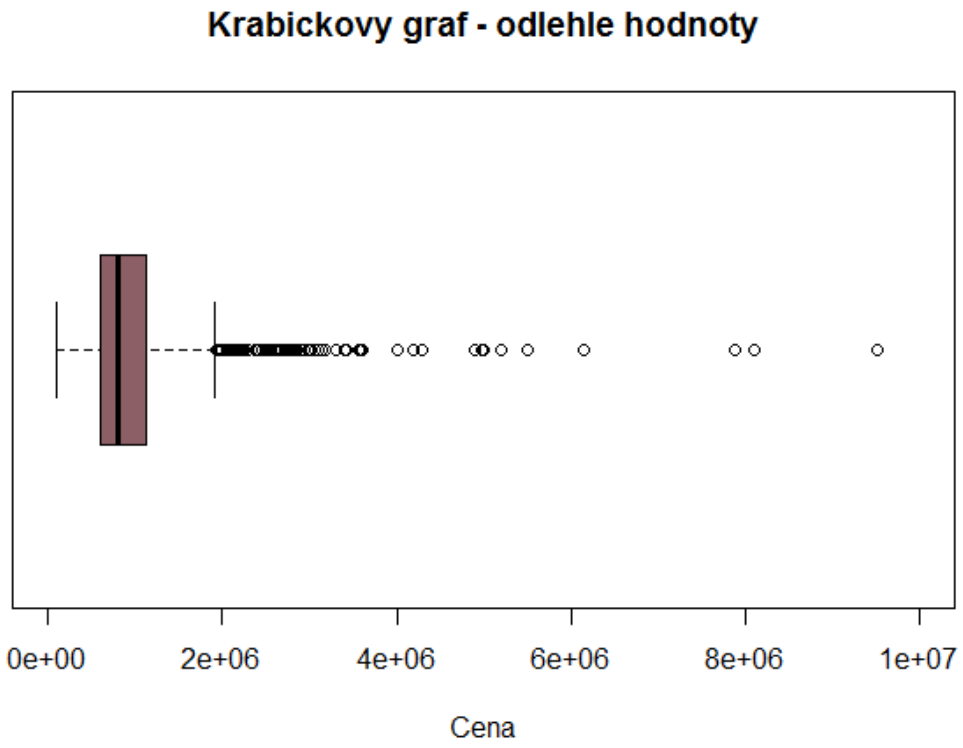
3.6.5 Krabičkový graf v R

Krabičkový graf, blíže popsaný v kapitole 2.4.2, má mnoho stran využití. Lze z něj vyčíst mnoho informací, jako jsou míry polohy, variability, ale také míry asymetrie. V R existuje několik možností, jak zkonstruovat krabičkový graf. Budou představeny v bodech níže.

a) Krabičkový graf s odlehlými hodnotami

Tvorba krabičkového grafu v R je podobná jako u histogramu. Základním příkazem ***boxplot(datasf$price)*** bychom získali graf, který je ale třeba z několika hledisek upravit. Z hlediska délky řady odlehlých hodnot je lepší graf otočit do horizontální polohy. Poté známými argumenty přidat nadpis a popis osy x a délku osy y, což může uživatele zmást, neboť v defaultním nastavení je graf vertikální a s ním i osa y je osa cen, což po otočení grafu zůstává. Krabičkový graf s odlehlými hodnotami ilustruje obrázek 3.11.

```
> boxplot(x = datasf$price, outline = T, col = "lightpink4", horizontal = T,
xlab = "Cena", main = "Krabickovy graf - odlehle hodnoty", ylim = c(0,
10000000))
```



Obrázek 3.11: Krabičkový graf - odlehlé hodnoty

b) Krabičkový graf bez odlehlých hodnot

Krabičkový graf v plné podobě je značně nepřehledný, kvůli existenci odlehlých a extrémních hodnot. Funkce ***boxplot()*** pro takové případy obsahuje argument *outline*, což je logická hodnota indikující, zda chceme do grafu zahrnout odlehlá pozorování či ne. V případě *FALSE* je R nevykreslí a lépe se z grafu vyčtou hodnoty. Oproti předchozímu kódu, který zahrnuje i odlehlé hodnoty je třeba upravit délku osy y z 10 000 000 na 2 000 000. Graf bez odlehlých hodnot ilustruje obrázek 3.12.

```
> boxplot(x = datasf$price, outline = F, col = "lightpink4", horizontal = T,
xlab = "Cena", main = "Krabickovy graf - bez odlehlych hodnot", ylim = c(0,
2000000))
```

Obrázek 3.12: Krabičkový graf - bez odlehlých hodnot

Z krabičkového grafu bez odlehlých hodnot lze lépe vyčíst hodnoty o ceně nemovitostí. Minimální cena bude přibližně 100 000, maximální necelé 2 000 000 (za předpokladu neuvažování odlehlých hodnot). Dolní kvartil bude lehce nad 500 000 a horní kvartil přibližně 1 100 000. Medián přibližně 800 000.

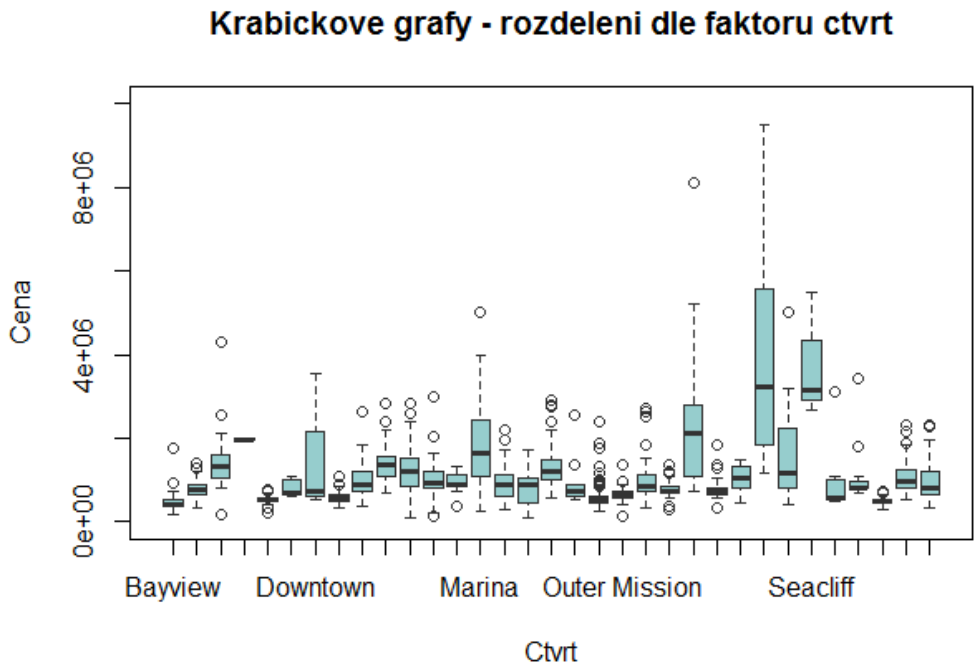
I přesto, že graf, kde *outline* = *F*, tedy kde nejsou zobrazeny odlehlé hodnoty, je lépe čitelný, není bez znalosti úplných informací charakteru dat pravdivý. Skutečnost odlehlých hodnot s jejím odstraněním nemizí. Analogicky jako v kapitole 3.6.1 věnující se konstrukci histogramu v R, lze hodnoty proměnné „useknout“ a poté graf sestrojit.

c) Krabičkové grafy rozdělené podle faktoru

Předchozí případy ilustrovaly krabičkový graf pro jednu proměnnou. V R existuje možnost zkonstruovat více než jeden krabičkový graf v jednom výstupu, přičemž grafy jsou stejné proměnné rozděleny podle faktoru jiné proměnné. Zápis je ve tvaru *proměnná~faktor*.

V případě, který je ilustrován níže na obrázku 3.13, kdy faktor zahrnuje více obměn, bude výsledný graf velice nepřehledný. Stejně jako každý problém, i tento se dá v R vyřešit.

```
> boxplot(price ~ neighborhood, data = datasf, col = "paleturquoise3", border
= "gray20", xlab = "Ctvrt", ylab = "Cena", main = "Krabickove grafy -
rozdeleni dle faktoru ctvrt", ylim = c(0, 10000000))
```



Obrázek 3.13: Krabičkové grafy - rozdělení dle faktoru čtvrť

d) Krabičkové grafy rozdělené podle faktoru po úpravě

Následující postup je řešením, jak vybrat pouze některé obměny faktoru do jednoho grafu o několika krabičkách. První krok je vypsat absolutní četnosti proměnné *neighborhood* (čtvrť) pomocí již známé funkce ***table()***. Příkazem ***head()*** zobrazit prvních osm pro představu.

```
> helper <- table(datasf$neighborhood)
> head(helper, 8)

         Bayview    Bernal Heights   Castro-Upper Market   Chinatown
              72                80                    51           2
  Crocker Amazon   Diamond Heights              Downtown   Excelsior
              33                 5                     4          81
```

V druhém kroku vyberu, které čtvrti chci zobrazit v grafu o větším počtu krabiček, konkrétně o třech krabičkách, každá pro jednu čtvrť. Do vektoru do uvozovek uvedu tři vybrané čtvrti, každá pro jinou část San Francisca.

```
> helper2 <- c("Pacific Heights", "Western Addition", "Presidio Heights")
[1] "Pacific Heights"  "Western Addition"  "Presidio Heights"
```

Následující příkaz vybere z celého datového souboru jen ty řádky, které ve sloupci *neighborhood* mají názvy čtvrtí z pomocného vektoru *helper2* a vytvoří nový soubor.

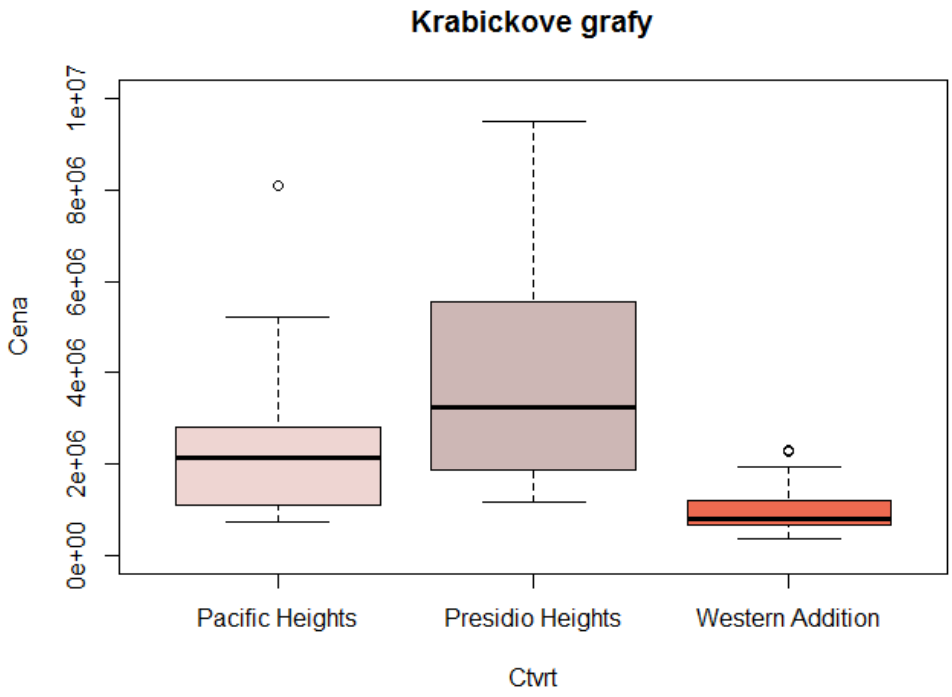
```
> data_help <- datasf[datasf$neighborhood %in% helper2, ]
```

Je nezbytností v novém souboru *data_help* proměnnou *neighborhood* překódovat jako charakterovou (slovní) proměnnou.

```
> data_help$neighborhood <- as.character(data_help$neighborhood)
```

Nyní již můžeme aplikovat krabičkový graf pro proměnnou *price* (cena) rozdělené podle proměnné *neighborhood* (čtvrť) z nově vytvořeného datového souboru *data_help*. Přidáme nadpis, popisky os a změníme barvy krabiček. Výstup je ilustrován na obrázku 3.14.

```
> boxp <- boxplot(price ~ neighborhood, data = data_help, xlab = "Ctvrt",
ylab = "Cena", main = "Krabickove grafy", col = c("mistyrose2", "mistyrose3",
"coral2"), ylim = c(0, 10000000))
```



Obrázek 3.14: Krabičkové grafy pro tři vybrané čtvrti

Nyní si uživatel může klást otázku, jak zjistí přesné hodnoty vykresleného krabičkového grafu. K tomu v R slouží funkce ***str()***, která vypíše strukturu krabičkového grafu, případně příkazy ***graf$stats***, ***graf$n***, ***graf$out***, kde za ‚graf' dosadíme jméno grafu, které jsme mu přiřadili.

```
> str(boxp)
List of 6
 $ stats: integer [1:5, 1:3] 740000 1079000 2125000 2787500 5200000 ...
 $ n    : num [1:3] 23 12 53
 $ conf : num [1:2, 1:3] 1562130 2687870 1530564 4919436 682068 ...
 $ out  : num [1:3] 8100000 2300000 2260000
 $ group: num [1:3] 1 3 3
 $ names: chr [1:3] "Pacific Heights" "Presidio Heights" "Western Addition"
```

Funkce ***str()*** pro krabičkový graf z obrázku 3.14 vyhodnotila 6 skupin informací: *stats, n, conf, out, group* a *names*. Skupina *$stats* informuje o hodnotách krabičkového grafu počínaje dolním vousem, dolní hrana krabičky, medián, horní hrana krabičky a horní vous. Výstup však ukazuje informace pouze pro první krabičku. V případě, jako je tento, pokud máme více krabičkových výstupů v jednom, použijeme příkaz *boxp$stats*, která vypíše informace pro všechny krabičky. V řádcích jsou informace týkající se pěti výše zmíněných veličin a ve sloupcích skupiny. Pro Pacific Heights je to skupina [,1], pro Presidio Heights skupina [,2] a pro Western Addition [,3].

```
> boxp$stats
        [,1]    [,2]    [,3]
[1,]  740000 1175000  340000
[2,] 1079000 1850000  642000
[3,] 2125000 3225000  801000
[4,] 2787500 5565000 1190000
[5,] 5200000 9500000 1945000
attr(,"class")
Pacific Heights
      "integer"
```

Takto zvlášť příkazem v R zjistíme i ostatní skupiny hodnot. Příkazem *boxp$n*, zjistíme počet pozorování pro každou skupinu, tj. pro každou čtvrť. Opět první počet z řady je pro Pacific Heights, druhý pro Presidio Heights a třetí pro Western Addition.

```
> boxp$n
[1] 23 12 53
```

Příkaz *boxp$out* slouží ke zjištění počtu hodnot, které R považuje za odlehlé či extrémní. Z funkce ***str(boxp)*** a jejího výstupu R vyhodnotilo 3 odlehlá pozorování. Všechny vypíšeme příkazem *boxp$out*.

```
> boxp$out
[1] 8100000 2300000 2260000
```

Aby uživatel věděl, která z odlehlých pozorování patří ke které skupině (ke které čtvrti), příkaze *boxp$group* zjistí skupinu, ve které příkaz *boxp$out* vyhodnotil odlehlá pozorování.

```
> boxp$group
[1] 1 3 3
```

Například pro skupinu 1, tedy Pacigic Heights bylo identifikováno jedno odlehlé pozorování, a to 8 100 000. Pro třetí skupinu, tj. Western Addition byly zjištěny 2 odlehlé hodnoty. Tyto údaje lze ověřit podle grafu na obrázku 3.14.

3.7 Výsledek analýzy

Z kapitoly 3.2, která zkoumá míry polohy jsem usoudila zešikmení hodnot proměnné *price* (cena) doprava, kde platí vztah $\bar{x} > \tilde{x}$, tedy 976 741,4 > 800 000. Takové rozdělení indikuje výskyt odlehlých hodnot. Z toho důvodu průměr inklinuje k vyšším hodnotám, neboť je ovlivněn právě extrémními hodnotami. Pokud uvážím, že se data týkají cen nemovitostí, je toto rozložení logické, neboť většina bytů a domů se bude pohybovat v nějakém cenovém intervalu a vždy bude existence luxusnějších nemovitostí, které jsou však extrémně drahé. Aby analýza proměnné *price* (cena) byla úplná, kapitola 3.3 pojednává o mírách variability. Směrodatná odchylka cen je 710 879, 7 a v kombinaci s průměrem vyšel variační koeficient přibližně 0,73. Lze tedy říci, že odlehlé hodnoty hodně ovlivňují jak míry polohy, tak míry variability. Pro ucelení analýzy cen nemovitostí je v kapitole 3.4 informace o šikmosti a špičatosti. I tyto dvě charakteristiky (míry) potvrdí, že soubor je kladně zešikmen a nejvíce hodnot je koncentrováno kolem průměru. Ceny nemovitostí se liší podle čtvrtí, což potvrzují i krabičkové grafy na obrázku 3.13 a 3.14.

Z kapitoly 3.6.2 lze říci, že největší četnost nemovitostí, co se týče počtu pokojů mají 0-pokojové nemovitosti, hned za nimi 2- a 3-pokojové. Nejméně se vyskytují 6- a více pokojové nemovitosti. V porovnání s cenou jsou tyto vícepokojové nemovitosti opět důvodem odlehlých hodnot, neboť jsou to luxusní byty a vily, kterých se v San Franciscu nevyskytuje tolik, jako běžných 0- až 5-pokojových bytů a domů. Co se týče rozmístění nemovitostí, variabilita proměnné *neighborhood* (čtvrť) je velice vysoká, tudíž rozdělení četností do jednotlivých kategorií čtvrtí je rovnoměrné.

4 Závěr

Cílem práce bylo přiblížit práci s R v oblasti popisné statistiky. Vytvořit jakýsi návod na popisnou statistiku a její implementaci v softwaru R, který se v dnešní době častěji stává populárnějším a vyhledávanějším programem pro práci s daty. Především díky jeho nemalým výhodám, jako je volná dostupnost a neomezenost licencí. V důsledku toho se na internetu rozšiřují diskuzní fóra, přibývají balíčky funkcí a celková obsluha programu se zjednodušuje.

První kapitola je teoretická vysvětlující, co je software R a jeho práce v programu RStudio, základní syntaxe a příkazy, které jsou nevyhnutelné, pokud chce s R kdokoli začít. Na první část navazuje druhá kapitola, která je rovněž teoretická, vysvětluje, co je to popisná statistika, jaké nástroje popisné statistiky známe, především tabulky a grafy. Dále rozlišení mezi různými typy proměnných. U každé charakteristiky je uvedena odpovídající funkce či příkaz z R s vysvětlením, co znamenají jednotlivé argumenty. Třetí kapitola je praktická, zde jsem realizovala příkazy v R na popisnou statistiku na reálném datovém souboru. Celá praktická část je postavena na datovém souboru, dostupném z balíčku v R, týkajícího se cen domů a bytů v San Franciscu v letech 2008-2009.

Praktická část je rozdělena do sedmi podkapitol. První se vztahuje k úvodu, který předchází samotné aplikaci. Instalace potřebného balíčku a příprava k analýze. Druhá popisuje vybrané míry polohy, třetí míry variability a čtvrtá míry koncentrace. I přes to, že některé příkazy a funkce jsou v R již zabudované a tím i jednodušší pro práci, nevyhnula jsem se tvorbě vlastních funkcí např. pro variační koeficient a několik dalších. Pátá podkapitola je popis tvorby tabulky rozdělení četností, obsahující příkaz v R i výstup v podobě obrázku. V šesté podkapitole je realizace grafických výstupů na pět typů grafů. Zde jsem představila jednu
ze silných stránek programu R, a to v tvorbě grafů a jejich přizpůsobení do dokonalé podoby. Poslední podkapitolou je shrnutí závěrů z analýzy z praktické části.

Troufám si tvrdit, že i méně zkušený uživatel zvládne po přečtení této práce analýzu dat v pojetí popisné statistiky v softwaru R. Bude schopen závěrů, co se týče charakteru dat a bude také umět předložená data prezentovat ve vhodném grafu a potenciálně jej upraví dle vlastních požadavků. Bakalářská práce v tomto pojetí plní svůj účel, neboť může sloužit jako pomocný manuál základní analýzy popisné statistiky v R. Cíle bakalářské práce byly splněny

Zdroje

CYHELSKÝ, Lubomír, Jana KAHOUNOVÁ a Richard HINDLS. *Elementární statistická analýza*. Vyd. 2., dopl. Praha: Management Press, 1999. ISBN 80-7261-003-1.

CYHELSKÝ, Lubomír a Eduard SOUČEK. *Základy statistiky*. Praha: Vysoká škola finanční a správní, 2009. ISBN 978-80-7408-013-5.

DALGAARD, Peter. *Introductory statistics with R*. 2nd ed. New York, N.Y.: Springer, c2008. Statistics and computing. ISBN 978-0-387-79053-4.

HINDLS, Richard. *Statistika pro ekonomy*. 4. vyd. Praha: Professional Publishing, c2003. ISBN 80-86419-52-5.

MAREŠ, Petr, Ladislav RABUŠIC a Petr SOUKUP. Analýza sociálněvědních dat (nejen) v SPSS. Brno: Masarykova univerzita, 2015. ISBN 978-80-210-6362-4.

MATLOFF, Norman S. *The art of R programming: tour of statistical software design*. San Francisco: No Starch Press, c2011. ISBN 978-1-59327-384-2.

SOUČEK, Eduard. *Statistika pro ekonomy*. Praha: Vysoká škola ekonomie a managementu, 2006. ISBN 978-80-86730-06-6.

ŘEZANKOVÁ, Hana a Tomáš LÖSTER. *Úvod do statistiky*. Praha: Oeconomica, 2009. ISBN 978-80-245-1514-4.

Internetové zdroje

R Core Team (2016). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/

Joseph Adler (2012). nutshell: Data for "R in a Nutshell". R package version 2.0. URL https://CRAN.R-project.org/package=nutshell

Lukasz Komsta and Frederick Novomestky (2015). moments: Moments, cumulants, skewness, kurtosis and related tests. R package version 0.14. URL https://CRAN.R-project.org/package=moments

SFGate: San Francisco Bay Area - News, Bay Area news, Sports, Business, Entertainment, Classifieds - SFGate. SFGate: San Francisco Bay Area - News, Bay Area news, Sports, Business, Entertainment, Classifieds - SFGate [online]. Copyright © Copyright [cit. 24.04.2017]. Dostupné z: http://www.sfgate.com

The Mercury News - Bay Area news, sports, business, entertainment, lifestyle and commentary. The Mercury News - Bay Area news, sports, business, entertainment, lifestyle and commentary [online]. Copyright © 2017 Digital First Media [cit. 24.04.2017]. Dostupné z: http://www.mercurynews.com

[online]. Dostupné z: http://www.stat.columbia.edu/~tzheng/files/Rcolor.pdf

Seznam obrázků

Obrázek 2.1: Spojnicový graf 23
Obrázek 2.2: Sloupcový graf 24
Obrázek 2.3: Histogram 25
Obrázek 2.4: Výsečový graf 27
Obrázek 2.5: Krabičkový graf 27
Obrázek 3.1: Mapa nemovitostí datového souboru San Francisco Home Sales 29
Obrázek 3.2: Tabulka rozdělení četností 37
Obrázek 3.3: Histogram v R - Sturgesovo pravidlo 38
Obrázek 3.4: Histogram v R – Scottovo pravidlo 39
Obrázek 3.5: Histogram v R – FD pravidlo a odlehlá pozorování 40
Obrázek 3.6: Histogram v R – FD bez odlehlých hodnot 41
Obrázek 3.7: Výsečový graf v R - bez úpravy 42
Obrázek 3.8: Výsečový graf v R - upravený 43
Obrázek 3.9: Sloupcový graf v R 43
Obrázek 3.10: Spojnicový graf v R 44
Obrázek 3.11: Krabičkový graf - odlehlé hodnoty 45
Obrázek 3.12: Krabičkový graf - bez odlehlých hodnot 46
Obrázek 3.13: Krabičkové grafy - rozdělení dle faktoru čtvrť 47
Obrázek 3.14: Krabičkové grafy pro tři vybrané čtvrti 48