Efektivita marketingových akcí

Bc. Petr Boháč

2025-03-02

Abstrakt

Tato semestrální práce si klade za cíl úspěšně vypracovat dataminingový projekt na téma **efektivita marketingových akcí**. Projekt je zaměřen na analýzu historických dat z marketingových kampaní a jejich vyhodnocení. Cílem projektu je zjistit, jak různé typy produktů reagují na marketing, jaký vliv má výše investice do marketingu na úspěšnost kampaně a zda je možné predikovat úspěšnost kampaně na základě historických dat. Projekt bude realizován v programovacím jazyce R a výsledky budou prezentovány v podobě R Markdown dokumentu. Struktura práce se drží jednotnou DM metodologií CRSIP-DM používanou při řešení data miningových projektů.

I. Business Understanding

Základem každého úspěšného dataminingového projektu je pochopení jeho pointy, respektive proč se do projektu pouštím a co od něj očekávám.

Náš pomyslný zákazník je marketingová agentura, která se pravděpodobně zabývá marketingovými kampaněmi pro širokou škálu produktů. Po několika úspěšných kampaních se rozhodla, že by bylo dobré analyzovat data z těchto kampaní, což by mohlo přinést nové poznatky, které by potenciálně mohly celý proces zefektivnit. Pro zákazníka by mohly být nejzajímavější například následující informace:

- jak různé typy produktů reagují na marketing
- jak velký vliv má výše investice do marketingu na úspěšnost kampaně
- možná predikce úspěšnosti kampaně na základě historických dat

Definice úspešnosti

V této fázi je také dobré si ujasnit, co je vlastně úspěšnost tohoto projektu z technické data miningové stránky. Hlavím cílem projeku je jednoznačně dozvědět se z historických data něco nového, o čem jsme doposud nevěděli, ale bez stanovení nějakého cíle se těžko určuje úspěšnost.

Jelikož agentura dělá marketing k různým typům produktů, bylo by dobré si udělat představu o tom, které kategorie reagují na marketing dobře a které ne, což by mohlo vést k vyřazení produktů, které jednoduše zákazníky tak nepřitahují. Tudíž nějaký způsob **seřazení produktů podle úspěšnosti kampaně** by byl ideální.

Další důležitou metrikou je výše investice do marketingu. Zde bychom se měli zaměřit na to, jaký vliv má výše investice na úspěšnost kampaně. Je jasné, že čím více peněz do marketingu investujeme, tím větší úspěch můžeme očekávat. **Ale jak moc?** A je to lineární závislost? Nebo je to spíše exponenciální? Nebo je to od určité částky kontraproduktivní? To jsou otázky, které bychom si měli položit a pokusit se na ně v rámci tohoto projektu najít odpovědi.

Zlatým grálem celého projektu je predikce úspěšnosti kampaně na základě historických dat. Zde bychom se měli zaměřit na to, jaké faktory ovlivňují úspěšnost kampaně a jak je možné tyto faktory využít k predikci úspěšnosti kampaně. Úspěšným výstupem by tedy mohl být nějaký **regresní model**, který by měl být schopen predikovat úspěšnost kampaně na základě historických dat. Tento model by mohl být použit k tomu,

abychom byli schopni předpovědět úspěšnost kampaně ještě před jejím spuštěním, což by mohlo vést k úspoře peněz a času.

Projektový plán

Kvalitně dopředu naplánovaný projekt je základním kamenem úspěšného projektu. V rámci technického plánu bychom měli mít jasno v tom, jaké kroky nás čekají a jakým způsobem budeme projekt realizovat. Co se týče výběru programovacího jazyka pro proces analýzy, nabízí se dva jasní kandidáti - Python a R. Oba jazyky jsou extensivně používané v oblasti datové analytiky a oba mají své výhody a nevýhody. Python je jazyk, který je velmi populární a má širokou škálu knihoven, které nám mohou pomoci při realizaci projektu. Na druhou stranu R je jazyk, který je více zaměřen na statistiku a analýzu dat. Jeho specializace nabízí uživatelům intuitivnější přístup k řešení úloh statistické analýzy, což by mohlo být pro náš projekt výhodou.

Programovací jazyk pro tento projekt byl hned ze začátku zvolen zadavatelem projektu, panem doktorem Lamrem, a to **R**. Jak již bylo zmíněno výše, R je specificky zaměřen na tento typ projektů, což by mělo usnadnit a urychlit celý proces.

Co se týče formy výstupu, nabízí se několik možností podle toho, komu budou výsledky projektu předkládány. V případě, že by se měl výsledek projektu prezentovat v místnosti plné technologicky nedotčených lidí (primárně management), bylo by dobré mít výstup v podobě prezentace, která by byla zaměřena na business a na to, co projekt přinesl. Ovšem vzhledem k tomu, že výsledky budou prezentovány před skupinou lidí, kteří se v oblasti datové analytiky pohybují, byl zvolen formát **R Markdown**, který je dobrým kompromisem mezi prezentací a technickým reportem. Tento formát umožňuje kombinovat text, kód a grafy do jednoho dokumentu, což je ideální pro prezentaci výsledků projektu.

II. Data Understanding

Data mining jako proces je postavený na předpokladu, že máme k dispozici dostatečné množství dat - obecně platí že čím více, tím lépe. S rostoucím množstvím dat je obtížnější si udělat představu o tom, co vlastně data obsahují a jaké informace nám mohou poskytnout. Proto je důležité si data nejprve důkladně prozkoumat a zjistit, s čím vlastně pracujeme.

Pro účely tohoto projektu nám byl poskytnut dataset, který obsahuje historická data z marketingových kampaní. Tento dataset nám obecně říká, na jaké typy produktů byly kampaně zaměřeny, kolik peněz bylo do jednotlivých produktů investováno a jak se kampaně promítly do zisků.

Co máme k dispozici za data?

Dataset nám byl poskytnut ve formátu **CSV** (z angl. "Comma-Separated Values"), což je standardní formát pro ukládání dat v tabulkové podobě. Tento formát je velmi populární a je podporován většinou programovacích jazyků, což usnadňuje práci s daty. Hodnoty jsou odděleny čárkami, desetinná místa jsou oddělena tečkami a celý soubor je očistěn od nadbytečných whitespace znaků - prakticky perfekně zformátované data.

Ke čtení dat z CSV souboru máme ve standardní knihovně R (dále jen *stdlib*) k dispozici několik funkcí, které nám umožňují načíst data do paměti a začít s nimi pracovat. Hlavní funkcí, která provádí vlastní parsování je funkce `read.table()`, která slouží pro čtení obecných *tabular* dat. Ostatní níže uvedené funkce jsou pouze šikovné wrapper funkce.

- `read.csv()`
  - data oddělena čárkami
  - desetinná místa oddělena tečkami
- `read.csv2()`
  - data oddělena středníky
  - desetinná místa oddělena čárkami
- `read.delim()`
  - data oddělena tabulátory (`\t`)

    - desetinná místa oddělena tečkami
- `read.delim2()`
    - data oddělena tabulátory
    - desetinná místa oddělena čárkami

```
# Načtení dat z CSV souboru do proměnné df
df <- read.csv("data/GOODS1n.csv")
```

V dalším kroku by bylo dobré si udělat high-level overview našich dat, která jsme načetli do proměnné `df`. Pro tento účel nám poslouží funkce `str()`, která je pro náš účel naprosto ideální. Zavoláním této jedné funkce získáme základní informace o struktuře dat, jako jsou názvy sloupců, datové typy a počet řádků.

```
# Základní informace o struktuře dat
str(df)
```

```
## 'data.frame':    200 obs. of  5 variables:
##  $ Class    : chr  "Confection" "Drink" "Luxury" "Confection" ...
##  $ Cost     : num  24 79.3 82 74.2 90.1 ...
##  $ Promotion: int  1467 1745 1426 1098 1968 1486 1248 1364 1585 1835 ...
##  $ Before   : int  114957 123378 135246 231389 235648 148885 123760 251072 287043 240805 ...
##  $ After    : int  122762 137097 141172 244456 261940 156232 128441 268134 310857 272863 ...
```

```
# Všechny možné kategorie produktů
unique(df$Class)
```

```
## [1] "Confection" "Drink"      "Luxury"     "Meat"
```

Jak z výstupu vidíme, načtený dataframe má rozměry 200 x 5, což znamená, že obsahuje **200 řádků**, ve kterých jsou data rozdělena do 5 sloupců. Sloupce, respektive proměnné, které máme k dispozici, jsou tedy následující:

- `Class`
    - Kategorie produktu (`Confection`, `Drink`, `Luxury`, `Meat`)
- `Cost`
    - Cena produktu
- `Promotion`
    - Výše investice do marketingu pro daný produkt
- `Before`
    - Zisk **před** marketingovou kampaní
- `After`
    - Zisk **po** marketingové kampani

Kvalitativní ověření dat

Na první pohled se zdá, že data jsou v pořádku a že neobsahují žádné chybějící hodnoty. Nicméně je dobré si udělat základní statistiku pro jednotlivé sloupce, abychom měli jistotu, že data jsou v pořádku a že neobsahují žádné extrémní hodnoty, které by mohly ovlivnit výsledky analýzy. Pro tento účel nám poslouží funkce `summary()` (informační hodnotou velice podobná `Data Audit` uzlu v SPSS Modeleru), která nám poskytne základní deskriptivní statistické informace o jednotlivých sloupcích, jako jsou průměr, medián, minimum a maximum.

```
# Základní statistika pro jednotlivé sloupce
summary(df)
```

```
##     Class                Cost          Promotion        Before
##  Length:200         Min.   :  5.08   Min.   :1004   Min.   :100751
##  Class :character   1st Qu.: 30.95   1st Qu.:1208   1st Qu.:149175
##  Mode  :character   Median : 53.68   Median :1470   Median :203421
##                     Mean   : 54.91   Mean   :1485   Mean   :201183
```

```
##                    3rd Qu.: 79.36   3rd Qu.:1745   3rd Qu.:251121
##                    Max.   :104.98   Max.   :1986   Max.   :299340
##      After
##  Min.   :104393
##  1st Qu.:159918
##  Median :215303
##  Mean   :214671
##  3rd Qu.:270884
##  Max.   :346375
```

I když náš dataset doposud vypadá v pořádku, ničemu neublíží, když si uděláme ještě pár dalších kontrol. Zmínili jsme například možnou existenci chybějících hodnot, které by mohly ovlivnit výsledky analýzy. Pro tento účel nám poslouží funkce `is.na()`, která nám vrátí TRUE pro každou chybějící hodnotu a FALSE pro každou hodnotu, která není chybějící. Funkci `colSums()` pak použijeme k tomu, abychom zjistili, kolik chybějících hodnot máme v jednotlivých sloupcích.

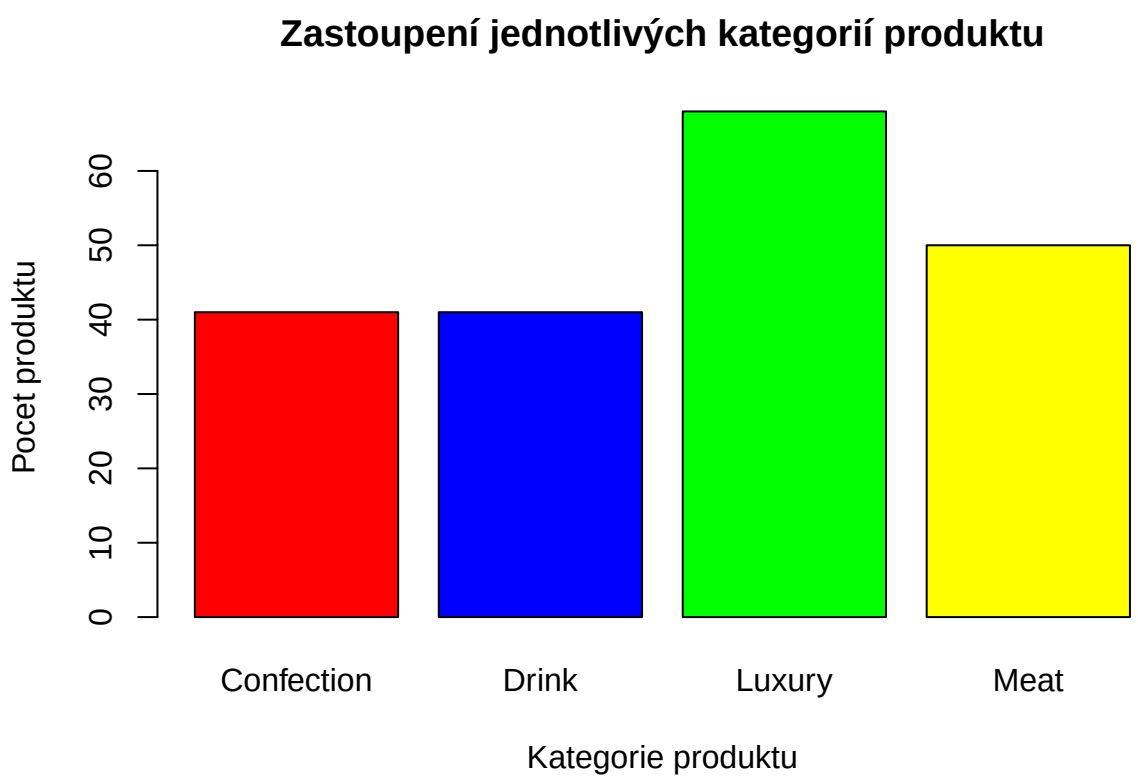
```r
# Kontrola chybějících hodnot
colSums(is.na(df))
```

```
##     Class      Cost Promotion    Before     After
##         0         0         0         0         0
```

Z výsupu je očividné, že data neobsahují žádné chybějící hodnoty, což nám o to více usnadní třetí fázi projektu (Data Preparation).

Proměnné, které nabývají pouze několika hodnot, jsou kvalitativní (kategorické) a proměnné, které nabývají libovolných hodnot, jsou kvantitativní (numerické). V našem případě máme k dispozici jednu kvalitativní proměnnou `Class`, která obsahuje 4 různé kategorie produktů. Kategorické proměnné můžou trpět tzv. *nevyvážeností*, což znamená, že některé kategorie mohou být zastoupeny více než jiné, což by mohlo ovlivnit výsledky analýzy.

```r
barplot(
  table(df$Class),
  main = "Zastoupení jednotlivých kategorií produktů",
  xlab = "Kategorie produktu",
  ylab = "Počet produktů",
  col = c("red", "blue", "green", "yellow"),
  names.arg = c("Confection", "Drink", "Luxury", "Meat")
)
```



Ze sloupcového grafu je vidět, že kategorie `Luxury` je zastoupena nejvíce s celkovým počtem produktů 70. Tata distribuce by mohla být problémová v případě trénování klasifikačního modelu a pravděpodobně by v další fázi projektu vyžadovala umělé vyvážení. Jelikož ale náš cílový model bude regresního charakteru, tak by to neměl být problém.

Co ale je pro regresní modely relevantní, je rozložení numerických proměnných. Pro tento účel nám poslouží funkce `hist()`, která nám zobrazí histogram pro jednotlivé sloupce. Histogram je grafické znázornění rozložení dat, které nám umožňuje vidět, jak jsou data rozložena a zda obsahují nějaké extrémní hodnoty.

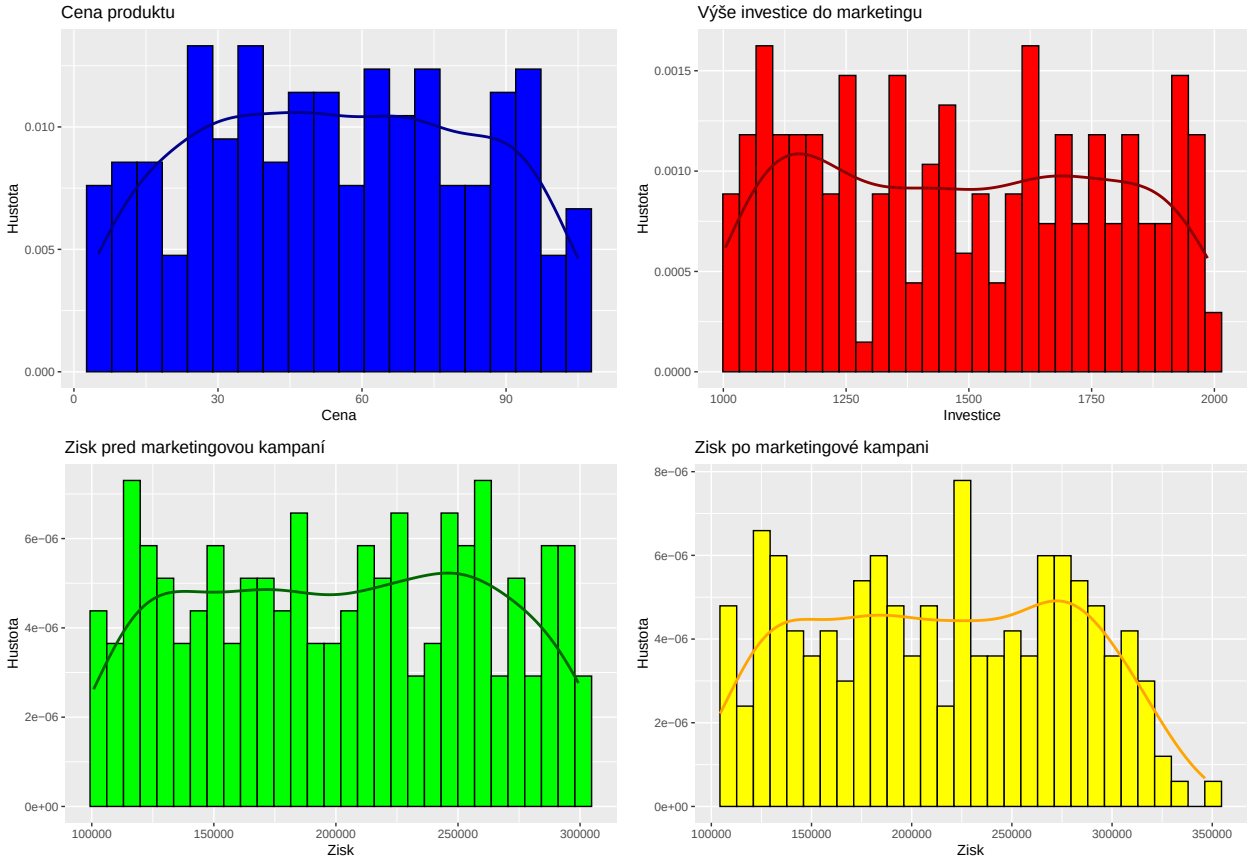
```
# Cena produktu
ggplot(df, aes(x = Cost)) +
  geom_histogram(
    aes(y = after_stat(density)),
    fill = "blue",
    color = "black",
    bins = 20
  ) +
  geom_density(color = "darkblue", linewidth = 1) +
  labs(title = "Cena produktu", x = "Cena", y = "Hustota")

# Výše investice do marketingu
ggplot(df, aes(x = Promotion)) +
  geom_histogram(
    aes(y = after_stat(density)),
    fill = "red",
    color = "black",
    bins = 30
```

```
  ) +
  geom_density(color = "darkred", linewidth = 1) +
  labs(title = "Výše investice do marketingu", x = "Investice", y = "Hustota")

# Zisk před marketingovou kampaní
ggplot(df, aes(x = Before)) +
  geom_histogram(
    aes(y = after_stat(density)),
    fill = "green",
    color = "black",
    bins = 30
  ) +
  geom_density(color = "darkgreen", linewidth = 1) +
  labs(title = "Zisk před marketingovou kampaní", x = "Zisk", y = "Hustota")

# Zisk po marketingové kampani
ggplot(df, aes(x = After)) +
  geom_histogram(
    aes(y = after_stat(density)),
    fill = "yellow",
    color = "black",
    bins = 30
  ) +
  geom_density(color = "orange", linewidth = 1) +
  labs(title = "Zisk po marketingové kampani", x = "Zisk", y = "Hustota")
```



Z výstupu je vidět, že rozložení jednotlivých proměnných je v pořádku a **neobsahuje žádné extrémní hodnoty** (tzv. *outliers*).

Hlubší průzkum dat

Ke konci této fáze, která je zaměřena na porozumění datům, by bylo také dobré prozkoumat, jesli data mezi sebou nemají nějaké zřejmé vztahy. Je dost možné, že žádné převratné vztahny mezi daty neodhalíme, ale jak již bylo řečeno, v této fázi máme porozumět datům a **pokusit** se odhalit potenciální vzory, které by mohly být užitečné pro další fáze projektu.

Jednou věcí, kterou můžeme zkusit, je vytvořit nad daty tzv. *korelační matici*. Korelační matice je tabulka, která nám ukazuje, jak jsou jednotlivé proměnné mezi sebou korelovány. Korelace je míra toho, jak jsou dvě proměnné mezi sebou spojeny a může nabývat hodnot od -1 do 1. Pokud je korelace blízká 1, znamená to, že obě proměnné jsou silně pozitivně korelovány. Pokud je korelace blízká -1, znamená to, že obě proměnné jsou silně negativně korelovány. Pokud je korelace blízká 0, znamená to, že obě proměnné nejsou mezi sebou korelovány. Pro výpočet korelační matice použijeme funkci `cor()`, která nám vrátí korelační matici pro všechny numerické sloupce v datovém rámci. Jelikož máme k dispozici pouze 4 numerické sloupce, tak matice bude mít rozměry 4 x 4.

```
# Korelační matice
cor(df[, c("Cost", "Promotion", "Before", "After")])
```

```
##                  Cost    Promotion      Before        After
## Cost       1.00000000 -0.032464896 -0.02322929 -0.020879598
## Promotion -0.03246490  1.000000000 -0.06568177 -0.008242122
## Before    -0.02322929 -0.065681769  1.00000000  0.993937445
## After     -0.02087960 -0.008242122  0.99393744  1.000000000
```

Většina hodnot v korelační matici je blízká 0, což znamená, že mezi jednotlivými proměnnými není žádná silná korelace. Nicméně je zde jedna zajímavá věc - a to je silná pozitivní korelace mezi proměnnými `Before` a `After`, což je logické, jelikož zisk po marketingové kampani by měl být vyšší než zisk před marketingovou kampaní a naopak.

Korelační matice nám toho tedy moc neřekla, ale stálo to za pokus. Mohli bychom zkusit ještě nějaké pokročilejší metody, ale náš originální dataset stejně neobsahuje moc zajímavých dat, které bychom mohli prozkoumat. V další fázi projektu se zaměříme na přípravu dat pro další fáze projektu, což by nám mohlo otevřít nové možnosti pro hlubší analýzu.

III. Data Preparation

Třetí fáze CRISP-DM projektů obecně zabírá nejvíce času z celého projektu. Obecně se říká, že **80% času strávíme přípravou dat a 20% času analýzou dat**. Tento poměr je samozřejmě orientační a může se lišit projekt od projektu, ale je dobré mít na paměti, že příprava dat je velmi důležitá a že by se jí mělo věnovat dostatek času.

V reálných projektech se většinou setkáváme s daty, která nejsou v ideálním stavu a potřebují nejdříve trochu lásky. Může jít o různé problémy, jako jsou chybějící hodnoty, duplicitní hodnoty, extrémní hodnoty, špatné formátování dat a podobně. V našem případě jsme ale měli štěstí a dataset byl prakticky v perfektním stavu, což tuto fázi značně urychlí.

Derivace nových proměnných

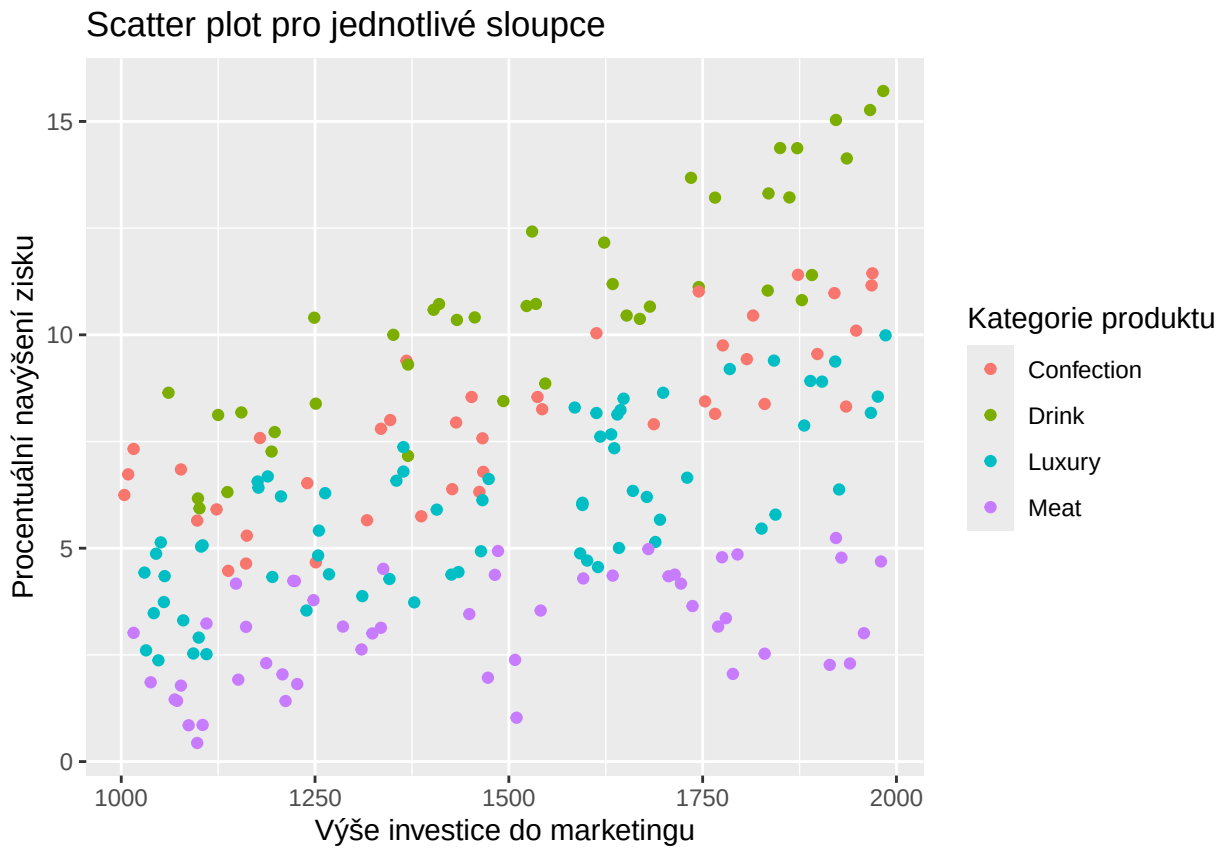
I tak bude ale potřeba si s daty trochu pohrát, než nad nimi budeme moci začít trénovat modely. Prvním krokem bude přidat do datového rámce nový sloupec, který bude zachycovat navýšení zisku po marketingové kampani. Nazveme tento sloupec `RevenueIncrease` a jeho hodnota bude vypočtena jako rozdíl mezi ziskem po marketingové kampani a ziskem před marketingovou kampaní v procentech, respektive následující vzorec:

$$\text{SalesIncrease} = \frac{\text{After} - \text{Before}}{\text{Before}} \times 100\%$$

```r
# Přidání nového sloupce do datového rámce
df$RevenueIncrease <- (df$After - df$Before) / df$Before * 100
```

Vybavme si poslední část předchozí fáze, kde jsme se snažili odhalit nějaké vzory v datech. V tu dobu jsme měli k dispozici pouze 4 numerické sloupce, které jsme prozkoumali. Nyní máme k dispozici 5 sloupců, což nám dává více možností pro analýzu dat. Například bychom mohli zkusit zjistit, jaký vliv má výše investice do marketingu na procentuální navýšení zisku. Pro tento účel použijeme funkci `plot()`, která nám zobrazí scatter plot pro sloupce `Promotion` a `RevenueIncrease`. Scatter plot je grafické znázornění dat, které nám umožňuje vidět, jak jsou data rozložena a zda obsahují nějaké extrémní hodnoty.

```r
# Scatter plot pro jednotlivé sloupce
ggplot(df) +
  geom_point(aes(
    x = Promotion,
    y = RevenueIncrease,
    color = Class
  )) +
  labs(
    title = "Scatter plot pro jednotlivé sloupce",
    x = "Výše investice do marketingu",
    y = "Procentuální navýšení zisku",
    color = "Kategorie produktu"
  )
```



Z předešlého grafu můžeme vyčíst hned několik věcí ohledně toho, jak různé kategorie produktů reagují na různé výše investice do marketingu. Je například patrné, že bez ohledu na to, kolik jsme investovali do marketingu pro produkty typu `Meat`, zisky se drží pod hranicí 5%. Ostatní kategorie se obecně drží jednoho trendu - čím více investujeme do marketingu, tím větší navýšení zisku můžeme očekávat. Nicméně nejzajímavější informací, kterou z grafu můžeme vidět, je to, že kategorie `Drink` reaguje na marketing **lépe, než všechny ostatní kategorie**.

Příznakování dat

Kategorie produktů jsou pro naše modely zásadní informací. Problémem ale je, že modely neumí pracovat s textovými hodnotami, které jsou v našem datasetu. Proto je potřeba převést tyto hodnoty na numerické hodnoty, které modely budou schopny zpracovat. Tento proces se nazývá *příznakování* (z angl. "feature engineering") a je velmi důležitý pro další fáze projektu.

Konkrétní proces příznakování se bude skládat z následujících kroků:

1. Derivace nových binárních proměnných pro každou z kategorií produktů
   - Tímto způsobem vytvoříme 4 nové proměnné, které budou mít hodnotu 1, pokud je produkt v dané kategorii a 0, pokud není
2. Vynásobení každé z nových proměnných hodnotou `Promotion` pro daný produkt
   - tohle fr nevim proc delame
3. Odstranění nyní redundantních sloupců `Class` a `Promotion`
   - sloupec `Class` je redundantní, jelikož jsme ho převedli na 4 nové proměnné
   - sloupec `Promotion` je redundantní, jelikož jsme ho vynásobili každou z nových proměnných

```r
# Derivace nových proměnných pro každou z kategorií produktů
df$Class_Confection <- ifelse(df$Class == "Confection", 1, 0)
df$Class_Drink <- ifelse(df$Class == "Drink", 1, 0)
df$Class_Luxury <- ifelse(df$Class == "Luxury", 1, 0)
df$Class_Meat <- ifelse(df$Class == "Meat", 1, 0)

# Vynásobení každé z nových proměnných hodnotou Promotion pro daný produkt
df$Class_Confection <- df$Class_Confection * df$Promotion
df$Class_Drink <- df$Class_Drink * df$Promotion
df$Class_Luxury <- df$Class_Luxury * df$Promotion
df$Class_Meat <- df$Class_Meat * df$Promotion

# Odstranění redundantních sloupců
df$Class <- NULL
df$Promotion <- NULL

# Výsledek příznakování
head(df)
```

```
##    Cost Before  After RevenueIncrease Class_Confection Class_Drink Class_Luxury
## 1 23.99 114957 122762        6.789495              1467           0            0
## 2 79.29 123378 137097       11.119486                 0        1745            0
## 3 81.99 135246 141172        4.381645                 0           0         1426
## 4 74.18 231389 244456        5.647200              1098           0            0
## 5 90.09 235648 261940       11.157319              1968           0            0
## 6 69.85 148885 156232        4.934681                 0           0            0
##   Class_Meat
## 1          0
## 2          0
## 3          0
## 4          0
```

```
## 5          0
## 6       1486
```

Rozdělení dat na různé sady

V posledním kroku této fáze je potřeba rozdělit data na různé sady, které budeme používat pro trénování a testování modelů. Dělení datasetů tímto způsobem je velmi důležité, jelikož nám umožňuje testovat modely na datech, která nebyla použita pro trénování. Tímto způsobem můžeme zjistit, jak dobře modely fungují na nových datech a zda jsou schopny generalizovat na nová data. Pokud bychom modely testovali na stejných datech, na kterých byly trénovány, mohli bychom získat zkreslené výsledky, které by nám neřekly nic o tom, jak dobře modely fungují na nových datech. Obecně se doporučuje rozdělit data na 3 sady:

- **Trénovací sada** (z angl. "training set") - tato sada se používá pro trénování modelů
- **Testovací sada** (z angl. "test set") - tato sada se používá pro testování modelů
- **Validační sada** (z angl. "validation set") - tato sada se používá pro validaci modelů

V našem případě, kdy celý náš dataset obsahuje žalostných 200 řádků, se omezíme pouze na první dva typy sad.

Obecně se doporučuje rozdělit data na 70% pro trénovací sadu a 30% pro testovací sadu. V našem případě ale radši z pochopitelných důvodů použijeme poměr 9:1, což bude znamenat, že trénovací sada bude obsahovat 180 řádků a testovací sada bude obsahovat 20 řádků. Pro rozdělení dat použijeme funkci `sample()`, která nám vrátí náhodný vzorek z datového rámce.

```r
# Rozdělení dat na trénovací a testovací sadu
set.seed(123) # pro reprodukovatelnost

train_index <- sample(seq_len(nrow(df)), size = 0.9 * nrow(df))
train_data <- df[train_index, ]
test_data <- df[-train_index, ]

# Kontrola rozměrů datových rámců
dim(train_data)
```

```
## [1] 180   8
```

```r
dim(test_data)
```

```
## [1] 20  8
```

IV. Modeling

Nejzajímavější, ale často i nejkratší fáze celého projektu je samotná tvorba modelů. Modely mohou být různého typu podle toho, jaký problém se snažíme vyřešit. V našem případě se jedná o regresní modely, které se snaží predikovat hodnotu na základě historických dat. Pro tento účel máme k dispozici několik různých typů modelů. Pro naše účely použijeme tzv. **lineární regresi**, která je jedním z nejjednodušších a nejčastěji používaných modelů pro regresní úlohy. Lineární regrese se snaží najít nejlepší přímku, která prochází daty a která nám umožňuje predikovat hodnotu na základě historických dat. Tento model je velmi jednoduchý a snadno interpretovatelný, což je pro naše účely ideální.

V R máme k dispozici funkci `lm()`, která nám umožňuje vytvořit lineární regresní model. Tato funkce nám umožňuje specifikovat, jaké proměnné chceme použít pro trénování modelu a jaký typ modelu chceme vytvořit.

V našem případě chceme vytvořit lineární regresní model, který bude predikovat hodnotu `RevenueIncrease` na základě hodnot proměnných `Cost`, `Class_Confection`, `Class_Drink`, `Class_Luxury` a `Class_Meat`. Tento model nám umožní zjistit, jaký vliv mají jednotlivé proměnné na hodnotu `RevenueIncrease` a jak jsou mezi sebou korelovány.

```r
# Vytvoření lineárního regresního modelu
model <- lm(
  RevenueIncrease ~
    Cost +
      Class_Confection +
      Class_Drink +
      Class_Luxury +
      Class_Meat,
  data = train_data
)
```

V ideálním případě bychom v této fázi měl natrénovat více kandidátů a iterovat je dokud nenajdeme ten nejlepší. Jelikož ale máme k dispozici tak málo dat a navíc se jedná o školní projekt, tak se spokojíme pouze s jedním modelem. V reálných projektech by bylo dobré natrénovat více modelů a porovnat jejich výsledky, abychom zjistili, který model funguje nejlépe. Pro tento účel máme k dispozici několik různých metrik, které nám umožňují porovnat jednotlivé modely a zjistit, který model funguje nejlépe.

V. Evaluation

V této fázi projektu se zaměříme na vyhodnocení modelu, který jsme vytvořili v předchozí fázi. Pro tento účel použijeme testovací sadu, kterou jsme vytvořili v předchozí fázi. Tímto způsobem můžeme zjistit, jak dobře model funguje na nových datech a zda je schopen generalizovat na nová data.

Vyhodnocení modelu

Vyhodnocování modelů je velmi důležité, jelikož nám umožňuje zjistit, jak dobře model funguje na nových datech a zda je schopen generalizovat na nová data. Pro tento účel máme k dispozici několik různých metrik, které nám umožňují porovnat jednotlivé modely a zjistit, který model funguje nejlépe. R nám nabízí velice užitečnou funkci `summary()`, která nám poskytne základní informace o modelu, jako jsou koeficienty, standardní chyby, t-hodnoty a p-hodnoty. Tyto informace nám umožňují zjistit, jak dobře model funguje a zda jsou jednotlivé proměnné mezi sebou korelovány.

```r
# Základní informace o modelu
summary(model)
```

```
##
## Call:
## lm(formula = RevenueIncrease ~ Cost + Class_Confection + Class_Drink +
##     Class_Luxury + Class_Meat, data = train_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.28884 -1.16234  0.09772  0.95959  2.32775
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -0.9974228  0.5066146  -1.969   0.0506 .
## Cost              0.0037140  0.0032326   1.149   0.2522
## Class_Confection  0.0058105  0.0003323  17.488  < 2e-16 ***
## Class_Drink       0.0074395  0.0003184  23.367  < 2e-16 ***
## Class_Luxury      0.0045391  0.0003240  14.011  < 2e-16 ***
## Class_Meat        0.0027198  0.0003373   8.062 1.16e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 1.234 on 174 degrees of freedom
## Multiple R-squared:  0.8651, Adjusted R-squared:  0.8612
## F-statistic: 223.2 on 5 and 174 DF,  p-value: < 2.2e-16
```

I když funkce `summary()` nabízí pouhé shrnutí modelu, výstup je pro laika velice těžko čitelný. Proto si celý výstup projdeme krok po kroku a vysvětlíme si, co jednotlivé části znamenají.

Residuals První část výstupu nám poskytuje základní informace o reziduálech (z angl. "residuals"), což jsou rozdíly mezi skutečnými hodnotami a hodnotami, které model predikuje. Tato část výstupu nám poskytuje minimum, 1. kvartil, medián, 3. kvartil a maximum. Lidově řečeno, reziduály nám říkají, jak moc se model mýlí. Čím menší reziduály, tím lépe model funguje.

Na základě výstupu můžeme říct, že model se mýlí v průměru o 1.16% a že maximální chyba je 2.32%. To je pro náš účel přijatelný výsledek, jelikož model je schopen predikovat hodnoty s relativně malou chybou.

Coefficients Druhá část výstupu nám poskytuje informace o koeficientech (z angl. "coefficients"), což jsou hodnoty, které model používá pro predikci. Tato část výstupu nám poskytuje odhadované hodnoty koeficientů, standardní chyby, t-hodnoty a p-hodnoty. Koeficienty nám říkají, jaký vliv má jednotlivá proměnná na hodnotu `RevenueIncrease`. Čím větší koeficient, tím větší vliv má daná proměnná na hodnotu `RevenueIncrease`.

Koeficienty nám říkají, jaký vliv má jednotlivá proměnná na hodnotu `RevenueIncrease`. Například koeficient pro proměnnou `Cost` je 0.0037, což znamená, že pokud zvýšíme cenu produktu o 1 Kč, očekáváme, že se zisk po marketingové kampani zvýší o 0.0037%. Koeficient pro proměnnou `Class_Confection` je 0.0058, což znamená, že pokud je produkt v kategorii `Confection`, očekáváme, že se zisk po marketingové kampani zvýší o 0.0058%. Koeficienty pro ostatní kategorie produktů jsou podobné a ukazují, že všechny kategorie produktů mají pozitivní vliv na hodnotu `RevenueIncrease`.

Z předchozího tvrzení to vypadá, že všechny prediktory mají silný vliv na hodnotu `RevenueIncrease`, ale to není pravda. Pouhá hodnota `Estimate` nám neříká celý příběh. Každý koeficient má také svou standardní chybu, která nám říka, jak jistý si model je. V případě prediktoru `Cost` je standardní chyba 0.0032, což znamená, že model si není jistý tím, jaký vliv má tato proměnná na hodnotu `RevenueIncrease`. Naopak koeficienty pro ostatní kategorie produktů mají velmi malou standardní chybu, což znamená, že model si je jistý (užitečně znázorněno `***` na konci řádků) tím, jaký vliv mají tyto proměnné na hodnotu `RevenueIncrease`. Kombinací koeficientu a standardní chyby dle následujícího vzorec

$$t = \frac{\text{Estimate}}{\text{Std. Error}}$$

můžeme vypočítat tzv. *t-hodnotu*, která nám říká, jak moc je daný koeficient významný. Čím větší t-hodnota, tím větší vliv má daná proměnná na hodnotu `RevenueIncrease`. T-hodnota se počítá jako podíl koeficientu a standardní chyby. V našem případě jsou t-hodnoty pro všechny kategorie produktů velmi vysoké, což znamená, že všechny kategorie produktů mají pozitivní vliv na hodnotu `RevenueIncrease`.

Ostatní metriky Poslední část výstupu nám poskytuje informace o ostatních metrikách, které nám říkají, jak dobře model funguje. Tato část výstupu nám poskytuje reziduální standardní chybu, R-kvadrát a F-statistiku. Do detailů se pouštět nebudeme, ale obecně platí, že čím menší reziduální standardní chyba, tím lépe model funguje. R-kvadrát nám říká, kolik variability v datech je vysvětleno modelem. Čím větší R-kvadrát, tím lépe model funguje. F-statistika nám říká, jak dobře model funguje v porovnání s průměrným modelem. Čím větší F-statistika, tím lépe model funguje.

Jednoduše řečeno, vzhledem k množství dat, které jsme měli k dispozici, náš model **funguje velmi dobře**. R-kvadrát je 0.8651, což znamená, že model vysvětluje 86.51% variability v datech. To je velmi dobrý výsledek a znamená to, že model je schopen predikovat hodnoty s relativně malou chybou.

Aplikace modelu na testovací sadu

V posledním kroku této fáze by bylo dobré aplikovat model na testovací sadu, kterou jsme vytvořili v předchozí fázi, abysme si ověřili jeho kvalitu. Tímto způsobem můžeme zjistit, jak dobře model funguje na nových datech a zda je schopen generalizovat na nová data. Pro tento účel použijeme funkci `predict()`, která nám umožňuje aplikovat model na nová data a získat predikce pro jednotlivé řádky.

```
# Použití modelu na testovací sadu
predictions <- predict(model, newdata = test_data)

# Porovnání predikcí s reálnými hodnotami
comparison <- data.frame(
  Actual = test_data$RevenueIncrease,
  Predicted = predictions
)

comparison
```

```
##        Actual Predicted
## 15   4.376382  3.196880
## 19   9.395873  7.576431
## 28   2.265785  4.486175
## 47   6.561927  4.519214
## 61   8.379899  9.758254
## 65   7.667841  6.796882
## 73   4.928820  5.908074
## 95  10.675622 10.645119
## 104  5.067882  4.188989
## 106 11.405809 10.253265
## 113 10.723665 10.532837
## 115  1.418863  2.527387
## 120 10.348791  9.937047
## 136  6.581501  5.377954
## 148  4.381812  3.910020
## 151  3.008190  4.406554
## 152  8.122059  7.536738
## 160  6.681992  4.745759
## 161  1.423121  2.306950
## 175  4.389444  4.848160
```

Jak jsme se už zmínili, model funguje velmi dobře a je schopen predikovat hodnoty s relativně malou chybou (maximálně 2.32%). V průměru se model mýlí o 1.16%, čemuž poslední výstup přibližně odpovídá.

VI. Deployment

Jelikož se jedná o školní projekt, tak poslední fázi projektu nebudeme nijak rozebírat. V reálných projektech by bylo dobré model nasadit do produkce a začít ho používat pro predikci. Tato fáze bude tedy spíše sloužit jako závěr projektu a shrnutí toho, co jsme udělali.

Cílem tohoto projektu bylo vytvořit lineární regresní model, který bude schopen predikovat nárůst zisku po marketingové kampani na základě historických dat. Mimo toho jsme se také snažili odhalit nějaké vzory v datech a zjistit, jaký vliv mají jednotlivé proměnné na ostatní proměnné.

Dozvěděli jsme se, že produkty kategorie `Drink` reagují na marketing lépe, než všechny ostatní kategorie. Na druhou stranu produkty kategorie `Meat` reagují na marketing nejhůře a zisky se drží pod hranicí 5% bez ohledu na výši investice do marketingu. Co se týče modelu, tak funguje velmi dobře a je schopen predikovat

hodnoty s relativně malou chybou (průměrně 1.16% a maximálně 2.32%). R-kvadrát je 0.8651, což znamená, že model vysvětluje 86.51% variability v datech.