UNIVERZITA HRADEC KRÁLOVÉ

PŘÍRODOVĚDECKÁ FAKULTA
KATEDRA MATEMATIKY

APLIKACE LINEÁRNÍ A LOGISTICKÉ REGRESE VE FINANČNICTVÍ

APPLICATIONS OF LINEAR AND LOGISTIC REGRESSION IN FINANCE

BAKALÁŘSKÁ PRÁCE

AUTOR PRÁCE **MONIKA SOCHOROVÁ**

VEDOUCÍ PRÁCE **doc. Mgr. DUŠAN BEDNAŘÍK, Ph.D.**

Hradec Králové 2021

Univerzita Hradec Králové
Přírodovědecká fakulta

Zadání bakalářské práce

Autor: **Monika Sochorová**

Studium: S18AM029BP

Studijní program: B1103 Aplikovaná matematika

Studijní obor: Finanční a pojistná matematika

Název bakalářské práce: **Aplikace lineární a logistické regrese ve finančnictví**

Název bakalářské práce AJ: Applications of linear and stochastic regression in finance

Cíl, metody, literatura, předpoklady:

Cílem je použití modelů lineární, logistické regrese a jim podobných pro predikování a klasifikaci dat z finanční oblasti s využitím
knihoven jazyka Python. Práce by měla ukázat, jak lze efektivně pracovat se soubory dat s použitím knihoven numpy, pandas resp. scikit-learn a vyladit vhodný model strojového učení.

1. Wes McKinney: Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython, O'Reilly Media; 2 edition (2017) .

2. G. James, D. Witten, T. Hastie, R. Tibshirani: An introduction to statistical learning (with applications in R), Springer (2014).

2.

Zadávající pracoviště: Katedra matematiky,
Přírodovědecká fakulta

Vedoucí práce: doc. Mgr. Dušan Bednařík, Ph.D.

Oponent: Mgr. Tomáš Zuščák, Ph.D.

Datum zadání závěrečné práce: 11.9.2020

Abstrakt

Tato práce obsahuje základní matematický popis lineárního a logistického modelu. Obsahuje dále představení programovacího jazyka Python, který je v závěru práce použit pro vyladění vhodného modelu lineární i logistické regrese na reálných datech.

Abstract

This thesis contains a basic mathematical describtion of linear and logistic regression models. It also introduces a programming language Python, which is later used to optimize an adequate model for both linear and logistic regression based on real data.

Klíčová slova

lineární regrese, logistická regrese, Python, regresní analýza

Keywords

linear regression, logistic regression, Python, regression analysis

Citace

SOCHOROVÁ, Monika. *Aplikace lineární a logistické regrese ve finančnictví*. Hradec Králové, 2021. Bakalářská práce. Univerzita Hradec Králové, . Vedoucí práce doc. Mgr. Dušan Bednařík, Ph.D.

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně a veškerou použitou literaturu a jiné zdroje řádně uvedla.

. . . . . .
Monika Sochorová
9. července 2022

Obsah

Úvod **3**

1 Regresní modely **5**
1.1 Lineární regresní model 6
1.1.1 Odhady parametrů 7
1.1.2 Kvalita modelu 10
1.2 Logistická regrese 12
1.2.1 Zobecněné lineární modely 12
1.2.2 Logistický regresní model 13
1.2.3 Odhad parametrů 16
1.2.4 Kvalita modelu 17

2 Regrese v prostředí Pythonu **20**
2.1 Seznámení s Pythonem 20
2.1.1 Knihovny 21
2.2 Postup tvorby regresního modelu 23
2.2.1 Příprava dat 23
2.2.2 Tvorba a hodnocení modelu 25

3 Aplikace lineární a logistické regrese **28**
3.1 Model odhadu cen nemovitostí 28
3.1.1 Data 28
3.1.2 Úprava dat 29
3.1.3 Tvorba modelu 34
3.2 Model splácení úvěru 38
3.2.1 Data 38
3.2.2 Úprava dat 38
3.2.3 Tvorba modelu 40

Závěr **45**

Literatura **47**

A **49**

B **51**

C **52**

D **54**

Úvod

Tato práce je zaměřena na představení lineárního a logistického regresního modelu a jejich využití v peněžní a ekonomické oblasti. Regresní analýza je matematické odvětví, které se zabývá zkoumáním vztahů mezi závislou a nezávislými proměnnými. Dvě základní funkce této analýzy jsou schopnost predikce a objasnění souvislostí.

V první části práce se budeme věnovat matematickému pojetí dvou vybraných regresních modelů. Nejprve se zaměříme na model lineární, který je základním stavebním kamenem pro další, složitější modely. Jeho pochopení je tedy pro tuto oblast zásadní. Lineární model představuje vztah, který co nejlépe vyjádří jistá pozorována data. Ve dvourozměrném prostoru se tedy jedná o přímku či křivku, kterou data proložíme. Úkolem této křivky je potom data co nejlépe aproximovat. Naším úkolem je najít koeficienty pro co nejlepší možný model. Ty se obvykle hledají pomocí metody nejmenších čtverců, která bude také popsána. Dále si popíšeme základy tzv. zobecněných lineárních modelů, které vznikly ze snahy rozšířit aplikovatelnost regresní analýzy na větší množství problémů. Nás bude nejvíce zajímat model logistický, který na rozdíl od lineárního pracuje s diskrétní závislou proměnnou. V našem případě budeme popisovat binární logistický model, který má pouze dvě hodnoty výstupu, a to existence a neexistence určité pozorované vlastnosti. Cílem binární logistické regrese je tedy předpovědět pravděpodobnost, že jev nastane, případně nenastane.

Jelikož se regresní analýza v současnosti přesunula výhradně do počítačového prostředí, je pro její aplikace nezbytná znalost nějakého programovacího jazyka. V našem případě budeme pracovat s jazykem Python, a to kvůli jeho přehlednosti a relativní jednoduchosti. Python je navíc v současnosti jedním z nejpopulárnějších jazyků pro vědecké účely, a má tedy i bohatou online komunitu. Ukážeme si zde základní přehled rozšiřujících knihoven a jejich možností pro aplikaci regresních modelů. Dále popíšeme základní postup tvorby regresního modelu v tomto prostředí.

Ve třetí části využijeme získaných znalostí o regresních modelech a zkombinujeme je s reálnou analýzou za pomoci Pythonu. Pro oba ze dvou dříve představených modelů si ukážeme implementaci na reálných datech. V případě lineární regrese budeme pracovat s datovým souborem s údaji z oblasti trhu nemovitostí obsahující informace o prodejích domů. Aplikaci logistické regrese si ukážeme na datech o splácení půjček v závislosti na parametrech jako úroková míra nebo vypůjčená částka. Naším úkolem bude v obou případech zjistit, které proměnné jsou pro odhad závislé proměnné nejdůležitější a také se pokusit o předpověď do budoucích období.

Cílem celé práce je popsat základní rysy lineárního a logistického modelu a ukázat jejich aplikovatelnost za pomoci jazyka Python. Dále je cílem představit jazyk Python a ukázat jeho možnosti pro účely regresní analýzy.

Kapitola 1

Regresní modely

V úvodu zde uvedeme krátké shrnutí historických původů pojmu regrese. Tento pojem byl poprvé použit v díle anglického vědce Francise Galtona v 19. století. Galton se mimo jiné zabýval tělesnou výškou dětí ve vztahu k rodičům. Jedním z výsledků jeho výzkumů bylo zjištění, že jakákoli výchylka od průměrné výšky, ať už kladná či záporná, která je pozorována u rodiče, se na dítě nepřenese celá. Ve výsledku lze tedy u potomků pozorovat jistá tendence k návratu do průměrné výšky. Tyto poznatky Galton shrnul v článku Regression towards mediocrity in hereditary stature (Regrese směrem k průměru v dědičném vzrůstu), odtud tedy pochází pojem regrese. [1].

Ze současného hlediska pohlížíme na regresní analýzu jako na statistickou techniku používanou k zobrazení vztahu mezi proměnnými. V regresní analýze máme dva hlavní druhy proměnných. První je závislá proměnná, což je veličina, kterou zkoumáme. Nezávislé proměnné jsou ty veličiny, u kterých chceme zjistit, jak závislou proměnnou ovlivňují. Hlavním úkolem regrese je tedy určit nejlepší možný model, který vystihne vazby mezi zkoumanými veličinami. Podle množství nezávislých proměnných lze regresní modely rozdělit na jednoduché (s 1 nezávislou proměnnou) a vícenásobné (2 a více proměnných). Je zřejmé, že určitou závislou proměnnou pouze velice zřídka ovlivňuje jen jediný faktor. Z tohoto důvodu se budeme dále zaměřovat na regresi vícenásobnou.

Její využití je velice široké napříč mnoha obory. Obecně lze říci, že dvě hlavní aplikace regrese jsou porozumění vztahů mezi veličinami a predikce do budoucích období. V oblasti financí a obchodu je její využití následující [10]:

- Předpovědi - běžné využití regrese spočívá v predikci budoucích rizik a příležitostí. Regrese umožní předpovědět změnu zkoumaného ukazatele na základě změny vstupních faktorů. Jedná se například o analýzu poptávky, příjmů, výdajů, počtu prodaného zboží apod.
- Model oceňování kapitálových aktiv (CAPM - *Capital asset pricing model*) - CAPM je model z oblasti teorie portfolia, díky kterému lze ohodnotit očekávanou výnosnost aktiva vzhledem k rizikům.

- Porovnání s konkurencí - další využití regrese spočívá v možnosti porovnání hlavních ekonomických ukazatelů mezi konkurenčními subjekty. Lze tak zjistit například vliv různých faktorů na množství prodaného zboží.
- Součást rozhodovacího procesu - na základě regresní analýzy dat v podniku lze činit rozhodnutí o prodejích, investicích či výdajích.

Dvěma nejčastěji využívanými druhy regrese, se kterými se nejen v ekonomické oblasti setkáme, jsou modely lineární (se spojitou závislou proměnnou) a logistický (s diskrétní závislou proměnnou). Na následujících stránkách se budeme věnovat popisu těchto dvou modelů z matematického hlediska.

1.1 Lineární regresní model

V následující části se vychází zejména ze zdroje [11] a [7].

Modelem lineární regrese rozumíme následující vztah pro náhodnou veličinu y:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k + \epsilon, \tag{1.1}$$

kde y je závislá proměnná (regresand), x_i, pro $i = 1, ..., k$, jsou nezávislé proměnné (regresory), β_i, pro $i = 1, ..., k$, jsou neznámé parametry (regresní koeficienty). Parametry β_i reprezentují očekávanou změnu proměnné y při změně x_j o jednotku, pokud jsou všechny ostatní regresory $x_i (i \neq j)$ konstantní. Náhodná veličina ϵ zde vyjadřuje složku, která není modelem vysvětlena a je nazývána náhodnou chybou, popř. odchylkou. Abychom mohli model sestavit, budeme dále pracovat s náhodným výběrem n vzájemně nezávislých pozorování y a s nimi spojených proměnných x. Zavedeme tedy model pro i-té pozorování jako:

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_k x_{ik} + \epsilon_i, \quad i = 1, 2, \cdots, n. \tag{1.2}$$

Pro jednotlivá pozorování lze rozepsat model jako soustava rovnic:

$$\begin{aligned}
y_1 &= \beta_0 + \beta_1 x_{11} + \beta_2 x_{12} + \cdots + \beta_k x_{1k} + \epsilon_1 \\
y_2 &= \beta_0 + \beta_1 x_{21} + \beta_2 x_{22} + \cdots + \beta_k x_{2k} + \epsilon_2 \\
&\vdots \\
y_n &= \beta_0 + \beta_1 x_{n1} + \beta_2 x_{n2} + \cdots + \beta_k x_{nk} + \epsilon_n.
\end{aligned}$$

Pokud dále využijeme maticového zápisu, lze soustava vyjádřit jako:

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1k} \\ 1 & x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{nk} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{pmatrix} + \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{pmatrix}.$$

Pro přehlednost dále zavedeme vektory $\mathbf{y} = (y_1, y_2, \cdots, y_n)^T$, $\boldsymbol{\beta} = (\beta_1, \beta_2, \cdots, \beta_k)^T$ a $\boldsymbol{\epsilon} = (\epsilon_1, \epsilon_2, \cdots, \epsilon_n)^T$ a matici $\mathbf{X}_{n\times(k+1)}$, nazývanou regresní matice, popř. matice plánu. Potom můžeme předchozí vztah vyjádřit ve formě:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}. \tag{1.3}$$

Aby bylo zavedení lineárního modelu kompletní, je třeba uvést následující podmínky:

1. Vektor proměnných $\mathbf{y}$ pochází z normálního rozdělení.
2. Model je lineární ve svých parametrech β_i.
3. Matice $\mathbf{X}$ má plnou hodnost, regresory jsou tedy vzájemně lineárně nezávislé.
4. Náhodné chyby ϵ_i jsou normálně rozdělené a platí pro ně:
   - $E(\epsilon_i) = 0$,
   - $D(\epsilon_i) = \sigma^2$ pro $i = 1, \cdots, n$,
   - $cov(\epsilon_i, \epsilon_j) = 0$ pro $i \neq j$.

Podmínka $D(\epsilon_i) = \sigma^2$ bývá označována jako podmínka homoskedasticity. Všimněme si dále, že z podmínky 4. lze odvodit následující vztahy pro vektor náhodných chyb $\boldsymbol{\epsilon}$:

$$E(\boldsymbol{\epsilon}) = \mathbf{0}, \tag{1.4}$$
$$cov(\boldsymbol{\epsilon}) = \sigma^2\mathbf{I}. \tag{1.5}$$

A ze vztahu 1.4 lze dále vyjádřit střední hodnotu vektoru $\mathbf{y}$ jako

$$E(\mathbf{y}) = \mathbf{X}\boldsymbol{\beta}. \tag{1.6}$$

1.1.1 Odhady parametrů

Odhad β

Pro stanovení regresního modelu, který bude vhodně popisovat pozorovaná data je třeba odhadnout neznámé parametry $\boldsymbol{\beta}$. V literatuře se nejčastěji setkáme s odhadem pomocí **metody nejmenších čtverců** (ang. *Least squares method* - LSM).

Hledáme tedy takové parametry $\beta_0, \beta_1, \cdots, \beta_k$, díky kterým bude součet čtverců odchylek pozorovaných y_i od odhadnutých hodnot $\widehat{y_i}$ minimální. Hledané parametry budeme nadále označovat $\widehat{\beta_0}, \widehat{\beta_1}, \cdots, \widehat{\beta_k}$. Jako $\widehat{y_i}$ budeme rozumět odhad střední hodnoty $\widehat{E(y_i)}$, tedy

$$\widehat{y_i} = \widehat{E(y_i)} = \widehat{\beta_0} + \widehat{\beta_1}x_{i1} + \widehat{\beta_2}x_{i2} + \cdots + \widehat{\beta_k}x_{ik}.$$

Funkci nejmenších čtverců, kterou budeme minimalizovat, označíme $S(\widehat{\beta}_i)$. Platí pro ni

$$\begin{aligned} S(\widehat{\beta}_i) &= \sum_{i=1}^{n}(y_i - \widehat{y}_i)^2 \\ &= \sum_{i=1}^{n}(y_i - \widehat{\beta}_0 - \widehat{\beta}_1 x_{i1} - \widehat{\beta}_2 x_{i2} - \cdots - \widehat{\beta}_k x_{ik})^2 \\ &= \sum_{i=1}^{n}(y_i - \widehat{\beta}_0 - \sum_{j=1}^{k}\widehat{\beta}_j x_{ij})^2. \end{aligned} \tag{1.7}$$

Pokud předchozí výraz derivujeme podle $\widehat{\beta}_0$ a $\widehat{\beta}_j$ a položíme nule, dostáváme

$$\begin{aligned} -2\sum_{i=1}^{n}(y_i - \widehat{\beta}_0 - \sum_{j=1}^{k}\widehat{\beta}_j x_{ij}) &= 0 \\ -2\sum_{i=1}^{n}(y_i - \widehat{\beta}_0 - \sum_{j=1}^{k}\widehat{\beta}_j x_{ij})x_{ij} &= 0. \end{aligned}$$

Tímto způsobem bychom získali $k+1$ rovnic s řešením $\widehat{\beta}_0, \widehat{\beta}_1, \cdots, \widehat{\beta}_k$. Využijeme-li maticového zápisu modelu $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$, budeme hledat vektor parametrů $\widehat{\boldsymbol{\beta}} = (\widehat{\beta}_0, \widehat{\beta}_1, \cdots, \widehat{\beta}_k)^T$, který minimalizuje

$$S(\widehat{\boldsymbol{\beta}}) = (\mathbf{y} - \boldsymbol{X}\widehat{\boldsymbol{\beta}})^T(\mathbf{y} - \boldsymbol{X}\widehat{\boldsymbol{\beta}}).$$

Tento výraz dále upravíme

$$\begin{aligned} S(\widehat{\boldsymbol{\beta}}) &= \mathbf{y}^T\mathbf{y} - \widehat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{y} - \mathbf{y}^T\mathbf{X}\widehat{\boldsymbol{\beta}} + \widehat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{X}\widehat{\boldsymbol{\beta}} \\ &= \mathbf{y}^T\mathbf{y} - 2\widehat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{y} + \widehat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{X}\widehat{\boldsymbol{\beta}}. \end{aligned}$$

Provedeme-li parciální derivaci podle $\widehat{\boldsymbol{\beta}}$ a položíme-li ji nule, získáme

$$-2\mathbf{X}^T\mathbf{y} + 2\mathbf{X}^T\mathbf{X}\widehat{\boldsymbol{\beta}} = \mathbf{0}.$$

Odkud po úpravě získáme

$$\mathbf{X}^T\mathbf{X}\widehat{\boldsymbol{\beta}} = \mathbf{X}^T\mathbf{y}. \tag{1.8}$$

Tato soustava rovnic se nazývá *normální rovnice*. Její řešení nalezneme jako

$$\widehat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}. \tag{1.9}$$

Nutná je existence inverzní matice $(\mathbf{X}^T\mathbf{X})^{-1}$. Tato matice bude existovat, pokud sloupce matice $(\mathbf{X}^T\mathbf{X})$, tedy regresory, budou lineárně nezávislé. Tato podmínka je splněna díky předpokladu 3. Abychom ověřili, že nalezený bod je lokálním minimem, určíme Hessovu matici[1] druhých derivací jako $\mathbf{H} = 2\mathbf{X}^T\mathbf{X}$. Lze ukázat (viz. str 21, [4]), že tato matice je pozitivně definitní, nalezené řešení je tedy ostrým lokálním minimem [3].

Vektor odhadnutých hodnot $\widehat{\mathbf{y}}$ je potom

$$\widehat{\mathbf{y}} = \mathbf{X}\widehat{\boldsymbol{\beta}} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}. \tag{1.10}$$

Odhad σ^2

Dalším parametrem, jehož odhad budeme hledat, je rozptyl σ^2 náhodných veličin $\boldsymbol{\epsilon}$. Rozdíl mezi pozorovanými hodnotami závislé proměnné a jejich odhadem můžeme vyjádřit pomocí $\boldsymbol{\epsilon} = (\epsilon_1, \epsilon_2, \cdots, \epsilon_n)^T$, kde $\epsilon_i = y_i - \widehat{y}_i$. Vyjádříme tedy

$$\boldsymbol{\epsilon} = \mathbf{y} - \hat{\mathbf{y}} = \mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}}. \tag{1.11}$$

Vektor $\boldsymbol{\epsilon}$ označíme jako *vektor reziduí* a v souvislosti s ním zavedeme dále **reziduální součet čtverců**, který označíme jako SS_{RES} a bude platit

$$SS_{RES} = \sum_{i=1}^{n}(y_i - \widehat{y}_i)^2 = \sum_{i=1}^{n}\epsilon_i^2 = \boldsymbol{\epsilon}^{\boldsymbol{T}}\boldsymbol{\epsilon}. \tag{1.12}$$

Po dosazení $\boldsymbol{\epsilon} = \mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}}$ dostáváme

$$\begin{aligned} SS_{RES} &= (\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}})^T(\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}}) \\ &= \mathbf{y}^T\mathbf{y} - 2\widehat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{y} + \widehat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{X}\widehat{\boldsymbol{\beta}}. \end{aligned}$$

Výraz můžeme dále díky 1.8 upravit na

$$\begin{aligned} SS_{RES} &= \mathbf{y}^T\mathbf{y} - 2\widehat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{y} + \widehat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{y} \\ &= \mathbf{y}^T\mathbf{y} - \widehat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{y}. \end{aligned} \tag{1.13}$$

Dále zavedeme pojem **reziduální rozptyl**, který vyjadřuje rozptýlenost pozorovaných dat od zvoleného modelu. Označíme ho MS_{RES} a pomocí něj budeme odhadovat σ^2. Připomeňme si také, že odhadujeme $k+1$ parametrů, a máme tedy $n-k-1$ stupňů volnosti. Vzorec pro MS_{RES} zavedeme jako

$$MS_{RES} = \frac{SS_{RES}}{n-k-1}. \tag{1.14}$$

Označíme-li tedy odhad parametru σ^2 jako $\widehat{\sigma}^2$, získáváme

$$\widehat{\sigma}^2 = \frac{\mathbf{y}^T\mathbf{y} - \widehat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{y}}{n-k-1}. \tag{1.15}$$

[1] čtvercová matice jejíž prvky jsou druhé parciální derivace funkce

1.1.2 Kvalita modelu

Kvalita zvoleného modelu je často vyjadřována pomocí **Koeficientu determinace $\mathbf{R^2}$**. Tento koeficient vyjadřuje, jaká část rozptylu proměnné $\mathbf{y}$ je vysvětlena modelem. Základní myšlenkou je rozklad celkového rozptylu na dvě části. První čast je rozptyl vysvětlený modelem a druhá rozptyl nevysvětlený. Dále budeme označovat

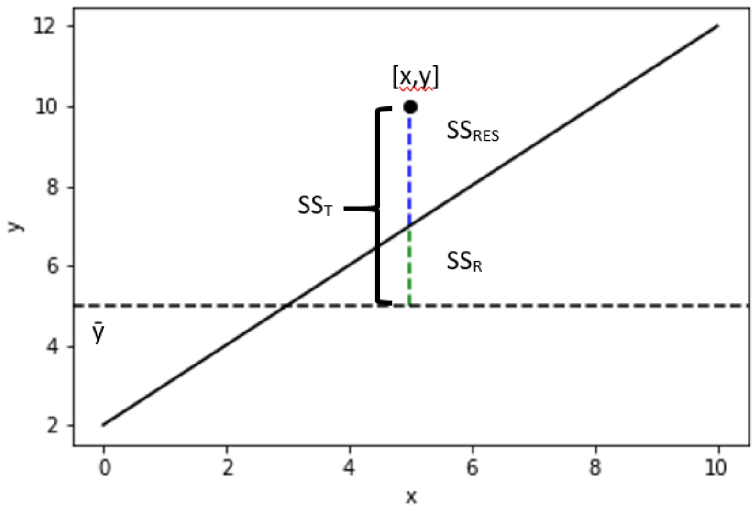
- $SS_T = \sum_{i=1}^{n}(y_i - \bar{y})^2$ (*Total sum of squares*) je celkový součet čtverců odchylek y_i od průměru $\bar{y}$,
- $SS_{RES} = \sum_{i=1}^{n}(y_i - \widehat{y}_i)^2$ (*Residual sum of squares*) je součet čtverců reziduí (část nevysvětlená modelem),
- $SS_R = \sum_{i=1}^{n}(\widehat{y}_i - \bar{y})^2$ (*Regression sum of squares*) je součet čtverců odchylek odhadnutých hodnot $\widehat{y}_i$ od průměru $\bar{y}$ (část vysvětlená modelem).

Přičemž platí

$$SS_T = SS_{RES} + SS_R.$$

Z 1.12 už známe vztah pro SS_{RES} jako

$$SS_{RES} = \mathbf{y}^T\mathbf{y} - \widehat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{y}.$$



Obrázek 1.1: Znázornění rozkladu rozptylu

Dále upravíme vztah pro SS_T

$$\begin{aligned} SS_T &= \sum_{i=1}^{n}(y_i - \bar{y})^2 = \sum_{i=1}^{n} y_i^2 - n\bar{y}^2 = \mathbf{y}^T\mathbf{y} - \frac{\left(\sum_{i=1}^{n} y_i\right)^2}{n} \\ &= \left(\mathbf{y}^T\mathbf{y} - \widehat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{y}\right) + \left(\widehat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{y} - \frac{\left(\sum_{i=1}^{n} y_i\right)^2}{n}\right) \\ &= SS_{RES} + SS_R. \end{aligned}$$

A získáváme tedy

$$SS_{RES} = \mathbf{y}^T\mathbf{y} - \widehat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{y}, \tag{1.16}$$

$$SS_R = \widehat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{y} - \frac{\left(\sum_{i=1}^{n} y_i\right)^2}{n}, \tag{1.17}$$

$$SS_T = \mathbf{y}^T\mathbf{y} - \frac{\left(\sum_{i=1}^{n} y_i\right)^2}{n}. \tag{1.18}$$

A pro samotný koeficient determinace R^2 získáváme

$$\begin{aligned} R^2 &= \frac{SS_R}{SS_T} = 1 - \frac{SS_{RES}}{SS_T} \\ &= \frac{\widehat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{y} - \frac{\left(\sum_{i=1}^{n} y_i\right)^2}{n}}{\mathbf{y}^T\mathbf{y} - \frac{\left(\sum_{i=1}^{n} y_i\right)^2}{n}}. \end{aligned} \tag{1.19}$$

Je zřejmé, že R^2 bude nabývat hodnot mezi 0 a 1. Pokud bude R^2 rovno 0, model nevysvětluje žádnou část rozptylu závislé proměnné, a je tedy neužitečný. Naopak hodnota 1 vyjadřuje stoprocentní vysvětlení rozptylu, a jedná se tedy o dokonalou modelaci. Obecně lze říci, že při vytváření regresního modelu se snažíme o co nejvyšší hodnotu R^2. Navíc platí, že přidáme-li do modelu ke stávajícím proměnným další vysvětlovací proměnou x_i, hodnota R^2 se vždy buď zvýší nebo zůstane stejná. Chceme-li tedy mezi sebou porovnat modely s různým počtem proměnných, je třeba využít tzv. **Adjustovaného koeficientu determinace $\mathbf{R^2_{adj}}$**, který ve výpočtu zahrnuje i počty stupňů volnosti. Jeho vzorec je

$$R^2_{adj} = \frac{(n-1)R^2 - k}{n - k - 1}. \tag{1.20}$$

1.2 Logistická regrese

1.2.1 Zobecněné lineární modely

Tato část vychází zejména ze zdrojů [7] a [13].

Logistickou regresi řádíme mezi takzvané zobecněné lineární modely (GLM - Generalized Linear Model). GLM představuje rozšíření regrese na data, která nesplňují podmínky pro obyčejný lineární regresní model. I v případě zobecněného lineárního modelu máme závislou proměnnou $\mathbf{y} = (y_1, y_2, \cdots, y_n)^T$, nezávislé proměnné $\mathbf{x_i} = (1, x_{i1}, x_{i2}, \cdots, x_{ik})^T$, pocházející z matice plánu $\mathbf{X}_{n\times(k+1)}$, a vektor neznámých parametrů $\boldsymbol{\beta} = (\beta_1, \beta_2, \cdots, \beta_k)^T$. Zobecněný lineární model sestává z následujících částí:

Závislá proměnná $\mathbf{y} = (y_1, y_2, \cdots, y_n)^T$,

která pochází z rozdělení exponenciálního typu, kam patří například normální, Poissonovo, binomické, exponenciální, Gamma rozdělení a další. Předpokládáme, že každé pozorování y_i je nezávislé a pro jeho hustotu f (popř. pravděpodobnostní funkci v případě diskrétní veličiny) platí

$$f(y_i, \theta_i, \phi) = exp\left(\frac{y_i\theta i - b(\theta_i)}{a(\phi)} + h(y_i, \phi)\right), \tag{1.21}$$

kde θ_i nazýváme přirozený parametr, ϕ rozptylový parametr a a, b, h jsou neznámé funkce. Funkce b navíc musí mít kladnou spojitou druhou derivaci. Pro exponenciální typy rozdělení dále platí

$$\begin{aligned} E(y_i) &= \mu_i = \frac{db(\theta_i)}{d\theta_i} \\ D(y_i) &= \frac{d^2b(\theta_i)}{d\theta_i^2}a(\phi) = \frac{d\mu_i}{d\theta_i}a(\phi) \end{aligned} \tag{1.22}$$

Zde si můžeme všimnout, že rozptyl $D(y_i)$ je funkcí střední hodnoty $E(y_i)$, na rozdíl od klasické lineární regrese, kde byl rozptyl konstantní.

Lineární prediktor η_i,

který je lineární kombinací proměnných $\mathbf{x_i}$ a parametrů $\boldsymbol{\beta}$ a je definován vztahem

$$\eta_i = \mathbf{x_i}^T\boldsymbol{\beta}. \tag{1.23}$$

Linkovací funkce g,

která vyjadřuje závislost mezi střední hodnotou $E(y_i) = \mu_i$ a lineárním prediktorem η_i. Platí pro ni vztah

$$g[E(y_i)] = g(\mu_i) = \eta_i = \mathbf{x_i}^T\boldsymbol{\beta}. \tag{1.24}$$

Po linkovací funkci požadujeme, aby byla ryze monotónní se spojitou druhou derivací. Můžeme také dále odvodit

$$E(y_i) = g^{-1}(\mu_i) = g^{-1}(\mathbf{x_i}^T\boldsymbol{\beta}).$$

Všimněme si, že v případě klasické lineární regrese byla střední hodnota $E(y_i)$ přímo rovna lineární kombinaci regresorů $\mathbf{x_i}$. V případě GLM nám stačí, že se bude $E(y_i)$ rovnat nějaké lineární kombinaci této funkce.

Pokud dále zvolíme

$$g[E(y_i)] = \theta_i. \tag{1.25}$$

nazýváme g kanonickou linkovací funkcí. Jak později uvidíme, kanonická linkovací funkce značně zjednodušuje vztahy při odhadech parametrů.

1.2.2 Logistický regresní model

V případě logistické regrese budeme zkoumat situaci, kdy závislá proměnná y není spojitá, jako u lineární regrese, ale kategorická, tedy diskrétní. Logistickou regresi dále můžeme rozdělit dle typu proměnné y na

1. binární - proměnná y je binární, tedy existují pouze 2 možné hodnoty - 0 a 1, tedy existence či neexistence zkoumaného jevu,
2. multinomickou - proměnná nabývá 3 nebo více různých hodnot, dále ji dělíme na
   (a) Ordinální - proměnná y je ordinální. Hodnoty, kterých nabývá, je možné určitým způsobem seřadit.
   (b) Nominální - proměnná y je nominální. Hodnoty, kterých nabývá, nelze seřadit.

Model

Dále se budeme zabývat pouze logistickým modelem s binární proměnnou. V logistickém regresním modelu opět figuruje závislá proměnná $\mathbf{y} = (y_1, y_2, \cdots, y_n)^T$, nezávislá proměnná $\mathbf{x_i} = (1, x_{i1}, x_{i2}, \cdots, x_{ik})^T$, parametry $\boldsymbol{\beta} = (\beta_1, \beta_2, \cdots, \beta_k)^T$. Dále předpokládejme, že závislá proměnná $\mathbf{y}$ pochází z alternativního rozdělení s parametrem π. Pro její hustotu tedy platí

$$f(y = y_i) = \begin{cases} \pi, & y_i = 1 \\ 1 - \pi, & y_i = 0 \\ 0, & \text{jinak.} \end{cases}$$

Toto rozdělení můžeme přepsat do kompaktnější formy jako

$$f(y = y_i) = \pi^{y_i}(1-\pi)^{1-y_i}, \quad \text{pro } y_i = 0, 1.$$

Pro veličinu y lze zjistit střední hodnotu a rozptyl následujícím způsobem

$$E(y) = 1\pi + 0(1-\pi)) = \pi, \tag{1.26}$$
$$D(y) = \pi(1-\pi). \tag{1.27}$$

Jelikož se hodnoty $E(y) = \pi$ pohybují v rozmezí $0 \leq \pi \leq 1$, je zřejmé, že se zde klasický lineární model tvaru $E(y) = \mathbf{x_i}^T\boldsymbol{\beta}$ k interpretaci nehodí. S výjimkou případu, kdy $\boldsymbol{\beta} = \mathbf{0}$ totiž nelze zaručit, že hodnoty π_i budou ležet v daném intervalu. Na rozdíl od klasické lineární regrese nebudeme navíc modelovat hodnoty y_i, ale pravděpodobnost, že y bude určité hodnoty nabývat. Nejčastěji se v této souvislosti setkáme s volbou podmíněné pravděpodobnosti

$$P(y_i = 1 \mid \mathbf{x_i}) \overset{\text{ozn.}}{=} \pi(\mathbf{x_i}). \tag{1.28}$$

V případě binární proměnné je logistický model nejčastěji popisován pomocí **logistické funkce**, která má následující tvar

$$F(t) = \frac{1}{1+e^{-t}}.$$

Jedná se o křivku esovitého tvaru, jejíž hodnoty leží mezi 0 a 1. Budeme-li dále uvažovat jako parametr t lineární kombinaci proměnných $\mathbf{x_i}$ a parametrů $\boldsymbol{\beta}$, můžeme zapsat vztah $\pi(\mathbf{x_i})$ jako

$$\pi(\mathbf{x_i}) = \frac{1}{1+exp(-\mathbf{x_i}^T\boldsymbol{\beta})} = \frac{exp(\mathbf{x_i}^T)\boldsymbol{\beta}}{1+exp(\mathbf{x_i}^T\boldsymbol{\beta})}. \tag{1.29}$$

Pokusíme se nyní najít linkovací funkci, pro kterou bude platit

$$g(\pi(\mathbf{x_i})) = \mathbf{x_i}^T\boldsymbol{\beta}. \tag{1.30}$$

Opět zde vystupuje lineární prediktor η_i ve tvaru 1.23. Dále si určíme podobu hustoty alternativního rozdělení ve dříve uvedeném tvaru 1.21

$$f(y_i, \pi(\mathbf{x_i})) = \pi(\mathbf{x_i})^{y_i}(1-\pi(\mathbf{x_i}))^{1-y_i} = exp\left[y_i \ln\frac{\pi(\mathbf{x_i})}{1-\pi(\mathbf{x_i})} + \ln(1-\pi(\mathbf{x_i}))\right]. \tag{1.31}$$

Odtud můžeme určit přirozený parametr θ_i ve tvaru $\ln\frac{\pi(\mathbf{x_i})}{1-\pi(\mathbf{x_i})}$. Kanonickou linkovací funkci g (1.25) můžeme tedy vyjádřit jako

$$g[\pi(\mathbf{x_i})] = \theta_i = \ln \frac{\pi(\mathbf{x_i})}{1 - \pi(\mathbf{x_i})}.$$

Tuto funkci budeme nadále označovat logit a platí pro ni vztah

$$logit(\pi(\mathbf{x_i})) = \ln \frac{\pi(\mathbf{x_i})}{1 - \pi(\mathbf{x_i})} = \mathbf{x_i}^T \boldsymbol{\beta}. \tag{1.32}$$

Předchozí úpravu nazýváme **logitová transformace** pravděpodobnostní funkce $\pi(\mathbf{x_i})$. Výraz v logaritmu nazýváme **poměr šancí** (ang. Odds Ratio - OR), který vyjadřuje podíl pravděpodobnosti výskytu jevu ($y_i = 1$) ku nevýskytu ($y_i = 0$).Tento poměr má podobu

$$OR = \frac{\pi(\mathbf{x_i})}{1 - \pi(\mathbf{x_i})}.$$

Z tohoto důvodu je funkce logit někdy označována jako log-odds.

Podobně jako jsme zavedli nutné podmínky pro lineární regresní model, stanovíme si je také pro model logistický. Jelikož se jedná o zobecněný lineární model, budou tyto podmínky oproti dříve uvedeným zjednodušeny.

1. Proměnná $\mathbf{y}$ je binární.
2. Existuje lineární vztah mezi nezávislými proměnnými a logitovou funkcí.
3. Jednotlivá pozorování jsou lineárně nezávislá.
4. Matice $\mathbf{X}$ má plnou hodnost.

Vidíme tedy, že na rozdíl od klasické lineární regrese nemusí být splněna podmínka homoskedasticity ani normality reziduí. Obvykle se můžeme setkat i s podmínkou dostatečně velkého rozsahu výběru.

1.2.3 Odhad parametrů

Parametry zobecněných lineárních modelů jsou nejčastěji odhadovány pomocí **Metody maximální věrohodnosti**. Jejím smyslem je nalezení odhadu vektoru parametrů $\boldsymbol{\theta}$ tak, aby byla maximální pravděpodobnost, že naše pozorované hodnoty náleží jistému typu rozdělení pravděpodobnosti. Funkci L, která je funkcí parametrů $\boldsymbol{\theta}$ a kterou se snažíme maximalizovat, nazýváme **věrohodnostní funkce**. Na obecné úrovni máme její předpis

$$L(\boldsymbol{\theta} \mid \mathbf{y}) = \prod_{i=1}^{n} f(y_i \mid \boldsymbol{\theta}). \tag{1.33}$$

Ze zavedení logistického modelu v předchozí části víme, že hustota $f(y_i)$ pro jednotlivá pozorování y_i z vektoru $\mathbf{y} = (y_1, y_2, \cdots, y_n)^T$ se rovná $f(y_i) = \pi(\mathbf{x}_i)^{y_i}(1 - \pi(\mathbf{x}_i))^{1-y_i}$. Odhadovanými parametry jsou $\boldsymbol{\beta}$ a věrohodnostní funkcí pro logistický model tedy rozumíme

$$L(\boldsymbol{\beta} \mid \mathbf{y}) = \prod_{i=1}^{n} f(y_i) = \prod_{i=1}^{n} \pi(\mathbf{x}_i)^{y_i}(1 - \pi(\mathbf{x}_i))^{1-y_i}. \tag{1.34}$$

Velice často se při řešení metody maximální věrohodnosti setkáme s použitím logaritmizované věrohodnostní funkce $\ln L$. Tu zavedeme následujícím způsobem

$$\begin{aligned} \ln L(\boldsymbol{\beta} \mid \mathbf{y}) &= \ln \prod_{i=1}^{n} \pi(\mathbf{x}_i)^{y_i}(1 - \pi(\mathbf{x}_i))^{1-y_i} \\ &= \sum_{i=1}^{n} \left[y_i \ln \pi(\mathbf{x}_i) + (1 - y_i) \ln(1 - \pi(\mathbf{x}_i)) \right] \\ &= \sum_{i=1}^{n} \left[y_i \ln \left(\frac{\pi(\mathbf{x}_i)}{1 - \pi(\mathbf{x}_i)} \right) \right] + \sum_{i=1}^{n} \ln(1 - \pi(\mathbf{x}_i)). \end{aligned} \tag{1.35}$$

V předchozím vztahu můžeme dále nahradit

$$1 - \pi(\mathbf{x_i}) = \frac{1}{1 + \exp(\sum_{j=0}^{k} \beta_j x_{ij})} \qquad \text{a} \qquad \ln \left(\frac{\pi(\mathbf{x}_i)}{1 - \pi(\mathbf{x}_i)} \right) = \sum_{j=0}^{k} \beta_j x_{ij}. \tag{1.36}$$

A získáváme tedy následující podobu věrohodnostní funkce

$$\begin{aligned} \ln L(\boldsymbol{\beta} \mid \mathbf{y}) &= \sum_{i=1}^{n} \left(y_i \sum_{j=0}^{k} \beta_j x_{ij} + \sum_{i=1}^{n} \ln \left[\frac{1}{1 + \exp(\sum_{j=0}^{k} \beta_j x_{ij})} \right] \right) \\ &= \sum_{i=1}^{n} \left(y_i \sum_{j=0}^{k} \beta_j x_{ij} - \sum_{i=1}^{n} \ln \left[1 + \exp(\sum_{j=0}^{k} \beta_j x_{ij}) \right] \right). \end{aligned} \tag{1.37}$$

Pro nalezení maxima funkce L je opět nutné položit parciální derivaci dle β_j rovnou nule. Získáme tak

$$\frac{d \ln L(\boldsymbol{\beta} \mid \mathbf{y}))}{d\beta_j} = \sum_{i=1}^{n} \left(y_i x_{ij} - \frac{\exp(\sum_{j=0}^{k} \beta_j x_{ij})}{1 + \exp(\sum_{j=0}^{k} \beta_j x_{ij})} x_{ij} \right) = 0. \tag{1.38}$$

Odkud dostáváme

$$\sum_{i=1}^{n} x_{ij} \left(y_i - \frac{\exp(\sum_{j=0}^{k} \beta_j x_{ij})}{1 + \exp(\sum_{j=0}^{k} \beta_j x_{ij})} \right) = 0. \tag{1.39}$$

Takto jsme získaly $k + 1$ rovnic. Tato soustava ale nemá žádné analytické řešení, díky kterému bychom získaly odhad $\widehat{\boldsymbol{\beta}}$ parametrů $\boldsymbol{\beta}$. Tyto odhady lze získat např. za pomocí **iterativně vyvažované metody nejmenších čtverců** (IRLS). Tato metoda se ovšem vymyká rozsahu této práce, její podrobný popis nalezneme např. v [13].

Odhad hodnoty y_i označíme opět jako $\widehat{y_i}$ a dostáváme

$$\widehat{y_i} = \widehat{\pi(\mathbf{x_i})} = \frac{\exp(\widehat{\mu_i})}{1 + \exp(\widehat{\mu_i})} = \frac{\exp(\mathbf{x_i}^T \widehat{\boldsymbol{\beta}})}{1 + \exp(\mathbf{x_i}^T \widehat{\boldsymbol{\beta}})} = \frac{1}{1 + \exp(-\mathbf{x_i}^T \widehat{\boldsymbol{\beta}})}. \tag{1.40}$$

1.2.4 Kvalita modelu

Tato část vychází ze zdrojů [7] a [5].

Klasifikační tabulka

Kvalitu modelu lze hodnotit např. pomocí klasifikační tabulky. Princip spočívá v sestrojení tabulky 2×2, která bude obsahovat počty pozorovaných pozitivních (1) a negativních (0) hodnot y_i spolu s odhadnutými počty. Odhadnuté počty stanovujeme na základě zvolené (*cut-off*) hodnoty pravděpodobnosti $P(y = 1 \mid \mathbf{x_i})$. Obvykle se volí hodnota $0,5$. Jako pozitivní jsou potom hodnoceny případy, kdy pravděpodobnost pozitivního výsledku je alespoň 50%. Tabulka má tedy tvar

	Pozitivní pozorované	Negativní pozorované
Pozitivní odhadnuté	a - Pravé pozitivní	b - Falešné pozitivní
Negativní odhadnuté	c - Falešné negativní	d - Pravé negativní

Je zřejmé, že vhodný model bude mít vysoké hodnoty a, d a naopak malé b, c. S těmito hodnotami jsou dále spojené následující pojmy

- Přesnost (ACC - accuracy) - vyjadřuje poměr všech správně odhadnutých hodnot vzhledem ke všem pozorovaným hodnotám.

$$ACC = \frac{a+d}{a+b+c+d} \tag{1.41}$$

- Sensitivita - vyjadřuje poměr správně odhadnutých pozitivních hodnot vzhledem k pozorovaným pozitivním hodnotám.

$$sensitivita = \frac{a}{a+c} \tag{1.42}$$

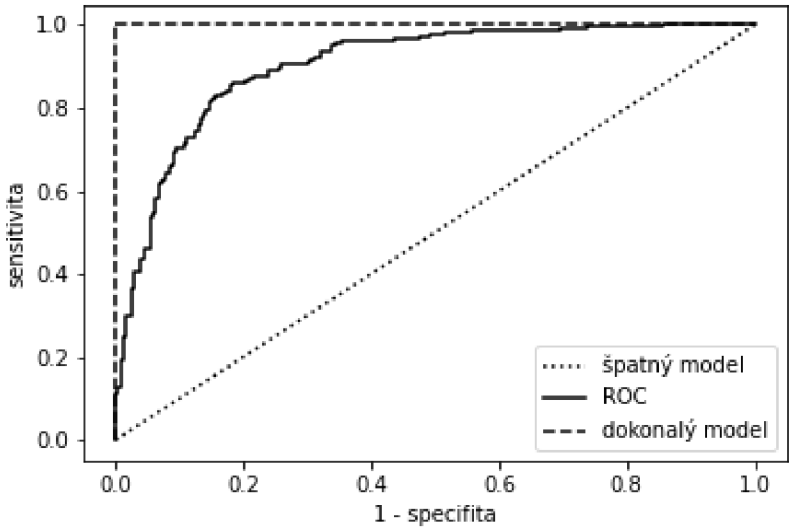
- Specifita - vyjadřuje poměr správně odhadnutých negativních hodnot vzhledem k pozorovaným negativním hodnotám.

$$specifita = \frac{d}{b+d} \tag{1.43}$$

ROC křivka

ROC křívka (*Receiver Operating Characteristics*) přímo vychází ze základů klasifikačních tabulek. Je zřejmé, že poměr pozitivních i negativních odhadnutých hodnot se bude měnit v závislosti na zvolené *cut-off* hodnotě. Pokud bychom zkoumali každou možnou hodnotu (v rozmezí 0 až 1) a jejich odpovídající klasifikační tabulku, bylo by možné zachytit specifitu a sensitivitu pro každou *cut-off* hodnotu ve formě grafu. Tomuto grafu potom říkáme ROC křívka, na x-ové ose najdeme hodnoty ($1 - specifita$ = poměr falešně pozitivních) a na y-ové $sensitivita$ = poměr pravých pozitivních.

Kvalitu modelu potom určujeme na základě plochy pod křivkou (AUC - *Area under curve*). Čím je větší AUC, tím více pravých pozitivních hodnot model zaznamená, zatímco minimalizuje počet falešně pozitivních. Hodnota AUC dobře zvoleného modelu se tedy bude blížit 1, zatímco model s hodnotou $0,5$ nemá žádnou vypovídající sílu.



Obrázek 1.2: ROC křivka

Kapitola 2

Regrese v prostředí Pythonu

V této části vycházíme ze zdrojů [6] a [8].

Kvůli velkému množství dat a složitosti jednotlivých výpočtů se regresní analýza v současnosti přesunula výhradně do počítačového prostředí. Pro správnou aplikaci regresních modelů je tedy nezbytné vhodně zvolit programovací jazyk, který bude obsahovat potřebné nástroje. Těchto jazyků je samozřejmě velké množství, z nejznámějších můžeme jmenovat R, C++, Java, Python, MATLAB apod. Každý z těchto jazyků má své výhody a nevýhody, při volbě je tedy třeba se řídit individuálními preferencemi. V této práci budeme pracovat s programovacím jazykem Python, jehož výhody a možnosti budou shrnuty v následujících odstavcích.

2.1 Seznámení s Pythonem

Python je populární programovací jazyk vyvinutý Guido van Rossumem v Nizozemí v roce 1991. Od té doby prošel mnoha vylepšeními a v současnosti je jedním z nejpoužívanějších programovacích jazyků vůbec. Je využíván i v rámci mezinárodních firem jako Amazon, Google či Facebook. Mezi jeho největší výhody patří relativní jednoduchost syntaxe a jeho dobrá čitelnost. S ní je spojena i rychlost vývoje kódu. Má také velké množství rozšíření a knihoven, což ho dělá vysoce využitelným napříč mnoha obory. Každá z těchto knihoven obsahuje předdefinované kódy, které zjednodušují a urychlují zápis. Další nespornou výhodou Pythonu je jeho rozsáhlá uživatelská komunita, a to nejen v internetovém prostředí.

Představme si nyní knihovny, které jsou pro oblast regrese nejdůležitější.

2.1.1 Knihovny

NumPy

Knihovna NumPy (zkratka pro Numerical Python) je jedním z fundamentálních balíčků pro vědecké výpočty v Pythonu, na kterém staví mnoho jiných knihoven. Jak napovídá název, Numpy obsahuje nezbytné nástroje pro numerické operace, včetně základních algebraických výpočtů a statistických metod. Dokáže navíc pracovat s rozsáhlými datovými soubory, přičemž základním stavebním kamenem této knihovny je n-dimenzionální objekt zvaný pole (ang. `ndarray`). Pole musí být vyplněno daty stejného typu (základními typy jsou: *strings* - písemné vyjádření, *integer* - celá čísla, *float* - reálná čísla, *boolean* - hodnoty True × False, *complex* - komplexní čísla). S tímto objektem lze provádět množství úprav, a to jak mezi více různými poli, tak i v rámci prvků jediného pole. Objekt pole je navíc nezbytný pro později zmíněnou knihovnu **scikit-learn**, která je pro tvoření regresních modelů v Pythonu zásadní.

Pandas

Pandas je další knihovnou, která umožňuje práci s rozsáhlými datovými soubory. Vychází částečně z knihovny NumPy, zde je ale základním prvkem objekt zvaný `dataframe` - datová struktura ve formě tabulky. Na rozdíl od `ndarray` v NumPy ale nemusí být datový typ v rámci jednoho dataframu homogenní, můžeme mít tedy tabulky obsahující jak číselné hodnoty, tak i textové vstupy. Mezi nejdůležitější nástroje Pandas patří:

- Práce s chybějícími daty (*NaN* - Not a Number).
- Mazání a vytváření nových sloupců.
- Operace s jednotlivými prvky sloupců či řad.
- Načítání dat v mnoha formách - SQL, soubory Excelu, csv (*comma separated values*) apod.
- Speciální nástroje pro práci s časovými řadami.

Knihovnu Pandas tedy využijeme zejména při prvotním načtení a přípravě dat na další analýzu.

Matplotlib

Matplotlib je primární knihovnou pro grafické zobrazování. Obsahuje různé typy grafů a jiných vizualizací a má široké využití napříč obory. Důležitou součástí je modul `pyplot`, který poskytuje rozhraní podobné MATLABu. Matplotlib není jedinou knihovnou určenou k vizualizaci dat, je ovšem nejpoužívanější.

Dále se můžeme setkat s knihovnou **Seaborn**, která se specializuje zejména na statistické grafy. Výstupy jsou často oproti Matplotlib grafům vizuálně zajímavější, funkčnost je ovšem podobná. Obě dvě knihovny vychází z objektů `ndarray` a `dataframe`.

SciPy

SciPy pochází ze spojení Science Python. Jak napovídá název, jedná se o knihovnu využívanou pro vědecké výpočty. Obsahuje mimo jiné nástroje pro integrální počet, optimalizaci, pokročilou lineární algebru, statistiku apod. SciPy opět pracuje s objektem `ndarray`.

scikit-learn

Knihovna scikit-learn obsahuje zásadní nástroje pro analýzu dat a strojové učení. Najdeme zde mimo jiné nástroje pro:

- Klasifikaci dat - metoda k-nejbližších sousedů, rozhodovací stromy...,
- Regresi - včetně lineární a logistické,
- Volbu a hodnocení modelů,
- Transformaci dat.

statsmodels

Statsmodels je knihovna Pythonu obsahující nástroje pro statistickou analýzu, a to včetně vlastních nástrojů pro tvorbu regresních modelů. Dále zde najdeme např. moduly pro analýzu rozptylu (ANOVA) či analýzu časových řad. Výstupem po provedení regrese pomocí statsmodels je rozsáhlá tabulka, obsahující mimo jine R^2, R^2_{adj}, odhad paramerů včetně intervalu spolehlivosti a další.

Obecně lze říci, že zatímco scikit-learn je zaměřen zejména na predikci, statsmodels lze využívat spíše pro statistické usuzování.

2.2 Postup tvorby regresního modelu

Tato část vychází ze zdrojů [2] a [9].

2.2.1 Příprava dat

Pro jakoukoli práci v Pythonu je nejprve nutné importovat veškeré potřebné knihovny, pro tento účel existuje příkaz `import` *knihovna*. Pro zjednodušení pozdějšího zápisu jsou nejčastěji názvy knihoven zkracovány. Tyto zkratky uvádíme už během importování, a to ve formě `import` *knihovna* `as` *zkratka*. V následující tabulce nalezneme nejčastěji používané zkratky pro námi používané knihovny.

knihovna	zkratka
`NumPy`	`np`
`Pandas`	`pd`
`Matplotlib.pyplot`	`plt`
`SciPy`	`sp`
`scikit-learn`	`sklearn`
`statsmodels.api`	`sm`
`Seaborn`	`sns`

Načtení a základní přehled

Po načtení knihoven je prvním krokem regresní analýzy v prostředí Pythonu načtení dat. Naše data se budou nacházet ve formě csv, tedy *comma separated values*. Pro prvotní přípravu je načteme ve formě objektu `pd.dataframe`, a to za pomoci příkazu

```
data = pd.read_csv(''nazev.csv'') .
```

Než začneme tvořit regresní model, je třeba se s používanými daty seznámit. Pro zobrazení prvních n záznamů můžeme použít funkci `data.head(n)`. Za pomoci příkazu `data.info()` zjistíme informace o typu dat, tedy jedná-li se o slovní proměnné typu object nebo číselné (int nebo float). Velice užitečný je také příkaz `data.describe()`, který zobrazí tabulku se základními statistickými údaji numerických proměnných. Nalezneme v ní součet všech prvků (`count`), průměr (`mean`), směrodatnou odchylku (`std`), minimum (`min`), maximum (`max`), hodnoty obou kvartilů a mediánu.

Chybějící hodnoty

Aby Python dokázal efektivně pracovat, je třeba se ujistit, že se v souboru nenachází chybějící data. Ta poznáme podle hodnoty `NaN`, tedy "Not a Number". Pracujeme-li s rozsáhlým souborem, je pro nalezení všech prázdných hodnot třeba využít příkazu

`isnull()`. Jeho výstupem jsou boolovské hodnoty, tedy vyplní náš soubor hodnotami `True` × `False`. Pro nalezení všech `NaN` vzhledem ke proměnným můžeme použít například funkci:

```
def nul():
    nul = data.isnull().sum[data.isnull().sum>0]
    return nul
```

Jejím výstupem je výčet sloupců, které chybějící hodnoty obsahují a jejich počet. Zjistíme-li, že se mezi daty chybějící hodnoty nachází, je několik způsobu, jak se s tímto problém vypořádat. Pokud je významná část hodnot u jedné proměnné chybějící, lze danou proměnnou pokládat za nedůležitou a ze souboru ji odstranit. K mazání sloupců či řad `dataframu` obsahujícího prázdné hodnoty lze využít příkaz `data.dropna()`. Dle volitelného parametrů `axis` lze volit mezi mazáním v rámci řad či sloupců. Popřípadě chceme-li odstranit sloupce až od určitého počtu `NaN`, použijeme parametr `thresh`. Další možností je chybějící hodnotu nahradit za pomoci příkazu `fillna()`. Dle kontextu lze zvolit jako novou hodnotu nulu, modus, medián, průměr, popřípadě jakoukoli jinou hodnotu.

Vizualizace a kontrola dat

Dalším krokem je průzkum samotných dat. K tomuto účelu poslouží zejména grafické knihovny `matplotlib` a `seaborn`. Za pomocí vhodných grafů lze získat představu o rozložení dat a vztazích mezi nimi. Pro vykreslení vztahů mezi všemi kombinacemi proměnných můžeme využít příkazu `sns.pairplot`. Ten vykreslí do podoby $n \times n$ bodové grafy, přičemž na diagonále se nachází graf distribuční funkce. Tento příkaz ovšem není zcela vhodný pro velké množství proměnných, jelikož se stává nepřehledným.

Pro vizualizaci mezi dvěma proměnnými lze vybírat z velkého množství grafů. Pro spojité veličiny lze využít např. příkaz `sns.scatterplot`, případně `plt.scatter`. Pro účely regrese nás budou nejvíce zajímat vztahy závislé proměnné a regresorů. Z grafů si lze udělat dobrou představu o tom, jak jednotlivé proměnné regresand ovlivňují. Velice praktickou je v tomto případě funkce `sns.regplot`, která vykreslí bodový graf spolu s regresní přímkou a znázorněním rozptylu bodů kolem této přímky. V případě diskrétních či kategorických proměnných můžeme využít například funkcí `sns.boxplot` nebo `sns.catplot`.

Pro zobrazení rozložení dat použijeme příkaz `sns.distplot`, který současně vykreslí histogram a distribuční funkci. Parametr `fit` zde lze použít pro porovnání s distribuční funkcí známého rozdělení. Vhodným grafem je také `sns.boxplot`, který zobrazí krabicový diagram. Na základě těchto grafů lze nalézt extrémní hodnoty. Tyto hodnoty jsou při tvorbě regresních modelů značně problematické, ovšem neexistuje zcela přesný návod na to, jak se s nimi vypořádat. Pozorování s těmito hodnotami lze odstranit, hrozí však riziko ztráty informace. V literatuře (např. [12]) se zejména u

lineárního modelu často setkáváme s doporučením proměnné s vysokou mírou šikmosti logaritmicky transformovat, díky čemuž se více přiblíží normálnímu rozdělení a zmenší se tím vliv extrémních hodnot. Často je této transformace využíváno zejména v ekonomickém modelování, kde mají data ke vzdáleným hodnotám tendenci. Transformace se dále doporučuje i v případech, kdy se vztahy mezi závislou a určitou nezávislou proměnnou blíží k exponenciálnímu rozdělení.

Jak bylo uvedeno, důležitým předpokladem provedení jakékoli regresní analýzy je absence vysoké míry korelace mezi nezávislými proměnnými. Informace o korelaci mezi jednotlivými proměnnými lze nejlépe zjistit za pomoci korelační matice. V Pythonu pro tyto účely použijeme příkaz `Corr()`, popřípadě nadstavbu `heatmap`, která využívá pro vyjádření vztahu barevnou škálu. Na základě zhodnocení potom nadbytečné proměnné odstraníme.

2.2.2 Tvorba a hodnocení modelu

Výběr proměnných

Výběr proměnných pramení ze snahy o vytvoření co nejjednoduššího modelu, který má co nejlepší predikční vlastnosti. Zároveň s nižším počtem proměnných dosáhneme navíc rychlejšího výpočtu. Metod sloužících k výběru proměnných nalezneme v Pythonu nepřeberné množství. V této práci jsme pro oba modely zvolili výběr na základě eliminace pomocí funkce `RFE`. RFE je zkratkou pro *Recursive feature elimination*. Tato funkce vybere na základě zvoleného počtu proměnných sadu těch nejrelevantnějších pro predikci sledované proměnné. V tomto případě byl napsán program, který postupně pro všechny zvolené počty proměnných zjistí koeficient determinace pro daný model a následně vybere dle nejlepšího výsledku optimální počet proměnných. Předpis této funkce nalezneme v příloze B.

Tvorba

Po kontrole a úpravě dat přichází na řadu tvorba modelu. Data je nejprve třeba rozdělit do dvou objektů - X, obsahující nezávislé proměnné a y, obsahující závislou proměnnou. X je objekt typu `dataframe`, zatímco y `ndarray`. Vhodným krokem je dále rozdělení na dva soubory, a to trénovací a testovací. Trénovacím souborem rozumíme takový, na základě kterého bude model tvořen. Pomocí testovacího souboru budeme potom hodnotit kvalitu modelu tak, že do něj vložíme jeho data a zjistíme tedy, s jakou úspěšností model předpovídá. Rozdělení do těchto dvou souborů provedeme za pomocí příkazu `train_test_split()` takto :

```
X_train, X_test, y_train, y_test
     = train_test_split(X,y, test_size=(n), random_state=(m)).
```

Parametr `test_size` vyjadřuje, jaká část z původního souboru bude tvořit testovací část. Obvykle se setkáme s hodnotou n mezi 0.2 a 0.35. Parametr `random_state` určuje, jak budou data pseudonáhodně rozdělena.

Lineární model

Samotný model pro lineární regresi pomocí metody nejmenších čtverců poté vytvoříme ve `scikit-learn` jako:

```
model = LinearRegression().fit(X_train,y_train).
```

Pro tvorbu modelu můžeme použít také knihovnu `statsmodels`. Zde je ovšem třeba nejprve manuálně k X přidat konstantní parametr pomocí příkazu

```
X = sm.add_constant(X)
```

model potom vytvoříme jako:

```
model=sm.OLS(y,X).fit().
```

Logistický model

Logistický model vytvoříme pomoci `scikit-learn` příkazem

```
model = LogisticRegression().fit(X_train,y_train).
```

Ve `statsmodels` využijeme funkce

```
model=sm.Logit(y,X).fit().
```

Vyhodnocení modelu

Regresní modely vytvářené za pomocí `statsmodels` obsahují funkci `.summary()`, která pro vytvořený model vypíše přehled statistických údajů a testů. Pro lineární model zde nalezneme: R^2, R^2_{adj}, odhady parametrů včetně $p-$hodnoty, intervalových odhadů a další. Pro logistický model tam nalezneme např. hodnotu věrohodnostní funkce, věrohodnostní funkci nulového modelu, odhady parametrů a další. Funkce `.summary()` lze proto využít k vyhodnocení důležitosti zvolených proměnných.

V případě `scikit-learn` lze získat hodnoty statistických testů pomocí modulu `sklearn.metrics`, kde nalezneme např. funkci `.score()`, která u lineární regrese určí koeficient determinace (1.19), zatímco u logistické regrese přesnost (1.41).

Jak jsme už nastínili, k vyhodnocení úspěšnosti modelu lze použít proložení modelu testovacím souborem, tedy X_test, y_test. Jelikož ke stanovení modelu nebyla tato data použita, pokud po proložení bude model ukazovat podobné hodnoty statistických testů jako trénovací soubor, lze předpokládat, že je model vhodně zvolen a má dobrou predikční schopnost.

Kapitola 3

Aplikace lineární a logistické regrese

Na začátku práce jsme stanovili, že regresní analýza je využívána zejména pro svou predikční schopnost. Tuto vlastnost se nyní pokusíme využít. V této části si ukážeme, jak lze regresní analýzu použít na reálných datech v prostředí Pythonu.

Celkem vytvoříme dva modely. Jeden z nich za pomoci lineární regrese odhadne cenu nemovitosti a stanoví nejdůležitější parametry, které ji ovlivňují. Druhý model se skrze logistickou regresi pokusí předpovědět, zda klient splatí či nesplatí úvěr. K analýze bude využit jazyk Python v prostředí aplikace Jupyter notebook dostupné skrze platformu Anaconda.

3.1 Model odhadu cen nemovitostí

Jak víme, lineární regresní model lze pro predikce využít v případě spojité zkoumané veličiny. Typickým příkladem spojité veličiny v oblasti financí je přirozeně cena statků. V našem případě se bude jednat o cenu nemovitostí, kterou se budeme snažit předpovědět. Zároveň budeme zkoumat, které proměnné cenu nejvíce ovlivňují.

3.1.1 Data

Data použitá k výstavbě modelu obsahují informace o prodaných nemovitostech ve městě Ames ve státě Iowa, USA v letech 2006 až 2010. Data byla získána ze stránek Kaggle.com (ke stažení zde), které obsahují veřejně dostupná reálná data z mnoha odvětví.

Náš datový soubor obsahuje celkem 1460 záznamů o prodeji. Nalezeme v něm 47 proměnných, z toho 36 numerických a 9 nominálních. Názvy a popis všech proměnných jsou k nalezení v příloze A.

3.1.2 Úprava dat

Po importování nezbytných knihoven si datový soubor pojmenovaný *domky.cvs* načteme pod názvem 'data' jako objekt `pd.DataFrame` pomocí příkazu

```
data = pd.read_csv('nazev.csv')
```

Jako první si pro představu zobrazíme prvních 15 pozorování příkazem `data.head(15)`. Všimneme si prvního sloupce s názvem *'Id'*, tato proměnná je zřejmě statisticky nevýznamná, a rozhodneme se ji proto odstranit příkazem `.drop()`. Z tabulky dále zjišťujeme, že se mezi daty nachází chybějící hodnoty. Jejich celkový počet zjistíme pomocí dříve definované funkce `nul()`. Z výstupu se dozvíme, že máme chybějící data u celkem 8 proměnných. Jedná se konkrétně o:

- Numerické proměnné: *'LotFrontage','MasVnrArea','GarageYrBlt'*
- Nominální proměnné: *'BsmtQual', 'BsmtCond', 'FireplaceQu', 'GarageQual', 'GarageCond'*

Nulových hodnot se zbavíme následujícím způsobem - u prázdných hodnot proměnných v *'BsmtQual', 'BsmtCond', 'FireplaceQu', 'GarageQual','GarageCond'* budeme předpokládat, že chybí, jelikož se daný objekt (tedy suterén, krb nebo garáž) v domě nenachází. Jsou to proměnné typu *object*, nahradíme je tedy hodnotou *'None'*. Zbylé chybějící proměnné nahradíme mediánem. Tuto úpravu provedeme příkazy

```
k = ['BsmtQual','BsmtCond','FireplaceQu','GarageQual','GarageCond']
for i in data[k]:
    data[i].fillna("None",inplace=True)

data = data.fillna(data.median())
```

Jako další krok je nutné se zbavit nominálních proměnných, se kterými ve slovní podobě nelze efektivně pracovat. V našem případě se jedná o veličiny popisující stav a kvalitu, tedy o data ordinálního charakteru. Můžeme jim tedy přiřadit odpovídající numerickou hodnotu. Toto zakódování provedeme příkazem

```
data.replace({"None" : 0,"Po" : 1, "Fa" : 2, "TA" : 3,
                    "Gd" : 4, "Ex" : 5},inplace=True)
```

Po prvotní úpravě dat se podíváme blíže na závislou proměnnou, tedy *'SalePrice'*. Příkazem `data['SalePrice'].describe()` si necháme zobrazit přehled základních statistických údajů:

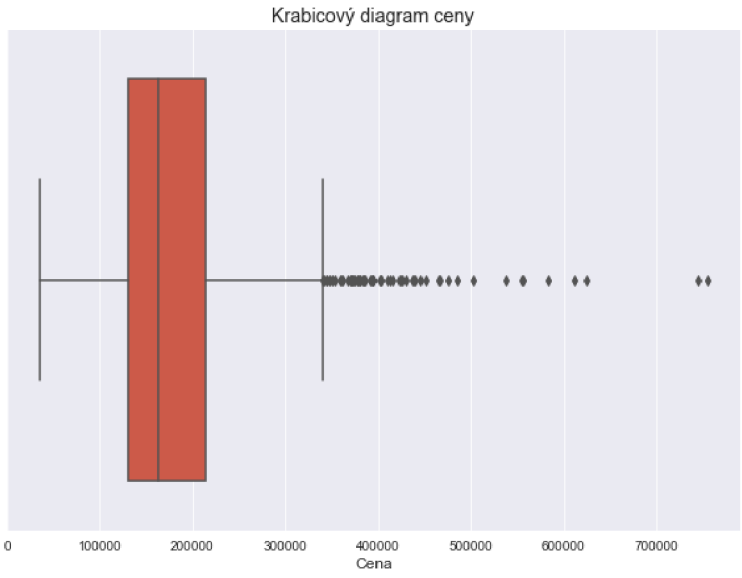
počet	1460
průměr	180921,195890
směrod. odchylka	79442,502883
minimum	34900
Q1	129975
medián	163000
Q3	214000
maximum	755000

Tabulka 3.1: Statistické ukazatele proměnné SalePrice

Vidíme tedy, že průměrná cena nemovitosti v našem souborů je 180921,2 USD. Nejlevnější nemovitost stála 34900 USD a nejdražší 755000 USD. Všimněme si dále velké směrodatné odchylky a faktu, že medián je menší než průměr. Z těchto informací lze soudit, že data budou vykazovat pozitivní zešikmení.

Pro lepší představu o rozložení si příkazem `sns.boxplot` necháme vykreslit krabicový diagram. Vypočítáme také koeficient šikmosti a špičatosti za použití funkcí `.skew()` a `.kurt()`. Dostáváme poté následující hodnoty:

- šikmost: 1,882876
- špičatost: 6,536282



Obrázek 3.1: Krabicový diagram závislé proměnné

Z krabicového diagramu zjišťujeme, že na horní cenové hranici nalezneme velké množství extrémních hodnot. Rozhodneme se tedy aplikovat logaritmickou transformaci, a to na všechny proměnné, jejichž koeficient šikmosti převyšuje v absolutní hodnotě 1. Jelikož se mezi daty nachází mnoho nulových hodnot, zvolíme jako transformační funkci přirozený logaritmus $\ln(x+1)$.

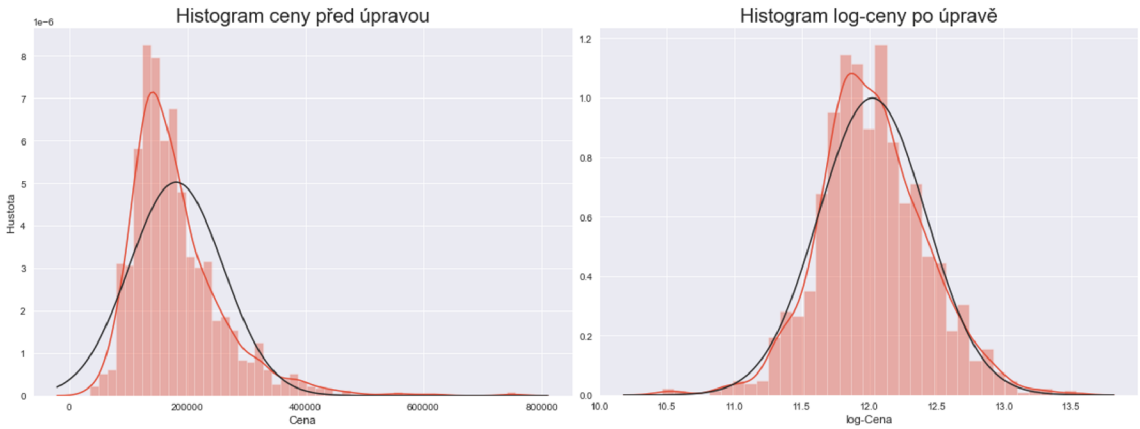
Definujme tedy funkci, která odpovídající proměnné transformuje[1]:

```
num = data.dtypes[data.dtypes !='object'].index
num=num.drop(['YearBuilt','YearRemodAdd','GarageYrBlt'])
skewed = data[num].apply(lambda x:
         sp.stats.skew(x)).sort_values(ascending=False)
nej_sikme = skewed[abs(skewed) > 1]

for i in nej_sikme.index:
    data[i] = np.log1p(data[i])
```

Logaritmizovány byly následující proměnné: *'PoolArea','LotArea','3SsnPorch', 'LowQualFinSF', 'MiscVal', 'KitchenAbvGr','BsmtFinSF2', 'BsmtHalfBath', 'ScreenPorch', 'EnclosedPorch','MasVnrArea', 'LotFrontage', 'OpenPorchSF', 'BsmtFinSF1', 'SalePrice','WoodDeckSF', 'TotalBsmtSF', 'MSSubClass', '1stFlrSF', 'GrLivArea'*

Porovnejme nyní rozložení závislé proměnné s normálním rozložením. Rozdíl mezi stavem před a po logaritmizaci nalezneme na následujícím obrázku.



Obrázek 3.2: Porovnání histogramů ceny před a po logaritmické transformaci

Podívejme se nyní na vztah závislé proměnné a regresorů. Nejprve si stanovíme proměnné s nejvyšší mírou závislostí vůči ceně. Závislostí v tomto případě rozumíme závislost lineární, kterou dokážeme kvantifikovat pomocí Pearsonova korelačního koeficientu. Můžeme nyní použít následující příkaz:

```
kor = abs(data.corr()['SalePrice'])
m_kor=kor[kor>0.5].sort_values(ascending=False).drop('SalePrice')
```

[1]Lineární modely, které prošly logaritmickou transformací, nazýváme log-lineární. Úskalím těchto modelů je interpretace parametrů proměnných, které byly transformovány. Ty potom nevyjadřují přesné množství změny při zvětšení příslušné proměnné o jednotku, ale pouze její poměrnou část.

Jako výstup získáváme proměnné s mírou korelace na ceně větší než $0,5$, seřazené od nejvyšší hodnoty. Výstup je shrnut v následující tabulce:

OverallQual	0,817185
GrLivArea	0,730254
GarageCars	0,680625
ExterQual	0,678840
KitchenQual	0,667893
GarageArea	0,650888
BsmtQual	0,615804
1stFlrSF	0,608955
FullBath	0,594771
YearBuilt	0,586570
YearRemodAdd	0,565608
FireplaceQu	0,546362
TotRmsAbvGrd	0,534422

Tabulka 3.2: Proměnné nejvíce korelované s cenou

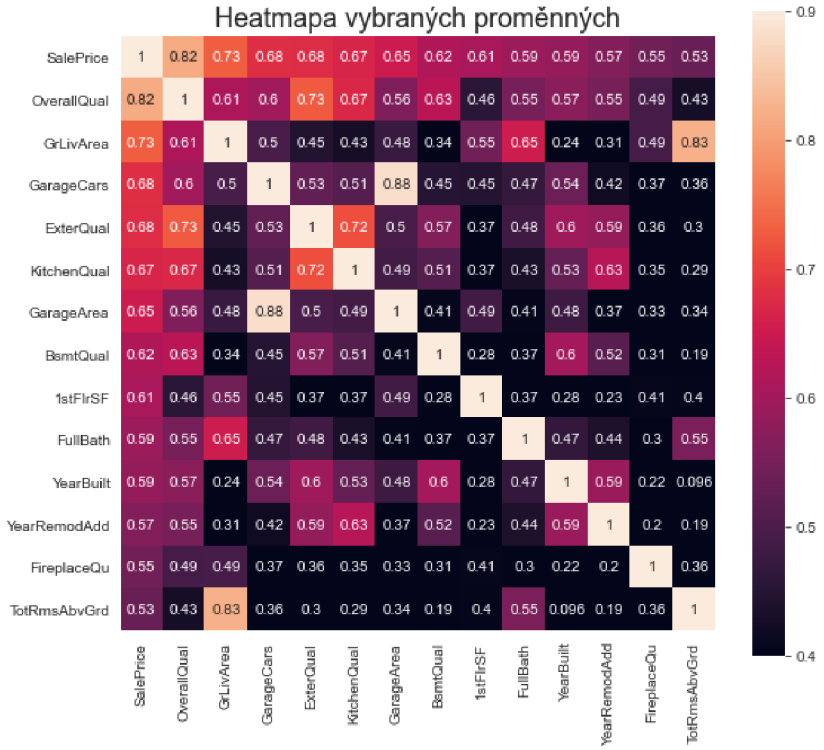
Vidíme tedy, že nejvyšší míru lineární závislosti vzhledem k ceně mají ukazatele celková kvalita, velikost zastavěné plochy, a počet míst v garáži. Jedná se o výsledky, které se shodují logickými úvahami, tedy že se stoupající kvalitou a velikostí nemovitosti její cena roste. Překvapivá je nepřítomnost proměnné *'LotArea'*, o které bychom předpokládali, že o ceně rozhoduje. Po vykreslení jejího grafu zjišťujeme, že závislost sice vykazuje, ale spíše exponenciální, kterou Pearsonův koeficient nedokáže dobře zachytit.

Dříve jsme si dále stanovili, že podmínkou pro lineární regresi je nezávislost regresorů. Podíváme-li se zběžně na seznam všech našich proměnných, je téměř jisté, že některé na sobě závislé budou. Můžeme zmínit například dvojice *'GarageCars','GarageArea'* nebo *'TotRmsAbvGrd','GrLivArea'*. Přehled o vztazích mezi vícero proměnnými najednou lze velmi dobře získat z korelační matice. Pro její vykreslení využijeme teplotní mapu, která jednotlivé míry korelace zobrazí pomocí barevné škály. Z důvodu velkého počtu proměnných pro přehlednost vykreslíme heatmapu pouze těch vybraných. V Pythonu pro tyto účely využijeme příkaz `sns.heatmap()` následujícím způsobem.

```
plt.figure(figsize=(9,8))
kormat = data[kor.index].corr().abs()
sns.heatmap(kormat, vmin=.4,vmax=.9, square=True, annot=True)
plt.title("Heatmapa vybraných proměnných",size=18)

plt.show()
```

Podobu heatmapy pro skupinu proměnných nejvíce korelovaných s cenou můžeme vidět na následujícím obrázku.



Obrázek 3.3: Heatmapa veličin nejvíce korelovaných na cenu

Hned si můžeme všimnout korelace mezi *'TotRmsAbvGrd','GrLivArea'* a *'GarageCars','GarageArea'*, jak jsme ostatně už předpověděli. Necháme si dále vypsat jednotlivé dvojice z celého souboru proměnných spolu s jejich hodnotou korelačního koeficientu, podle kterého je seřadíme. Použijeme k tomu příkaz

```
mat = data.drop('SalePrice',axis=1).corr().abs()
nejkorDvojice = mat.where(np.triu(np.ones(mat.shape),k=1)
    .astype(np.bool)).stack().sort_values(ascending=False)

nejkorDvojice.head(10).
```

Výstup obsahující 10 nejvíce korelovaných dvojic v absolutní hodnotě najdeme v následující tabulce:

GarageQual	GarageCond	0,959172
GarageCars	GarageArea	0,882475
Fireplaces	FireplaceQu	0,863241
TotalBsmtSF	BsmtCond	0,831949
GrLivArea	TotRmsAbvGrd	0,825521
YearBuilt	GarageYrBlt	0,777182
OverallQual	ExterQual	0,777182
KitchenQual	ExterQual	0,716122
TotalBsmtSF	BsmtQual	0,711808
BedroomAbvGr	TotRmsAbvGrd	0,676620

Tabulka 3.3: Korelace mezi dvojicemi proměnných

Každý prvek v této tabulce v sobě zároveň nese část informace o druhém z příslušné dvojice, proto si můžeme dovolit vždy jednu proměnnou odstranit. Odstraníme zpravidla tu, která se v tabulce vyskytuje vícekrát nebo tu, která má nižší hodnotu korelačního koeficientu vzhledem k závislé proměnné. Zabývat se budeme pouze dvojicemi s korelačním koeficientem větším než $0,75$. Odstraněné proměnné jsou: *'GarageCond', 'GarageArea', 'Fireplaces', 'BsmtCond', 'TotalBsmtSF', 'GarageYrBlt'.*

V rámci procesu úpravy dat jsme tedy nejprve odstranili prázdné hodnoty, zakódovali kategorické proměnné, transformovali proměnné s velkou mírou šikmosti a nakonec odstranili nadbytečné proměnné. Po tomto procesu nám z původních 47 proměnných zůstalo 40.

3.1.3 Tvorba modelu

Po úpravě proměnných se můžeme přesunout k samotné tvorbě modelu, která sestává ze dvou částí - výběru vhodných proměnných a samotném sestavení.

Jako první krok je nutné rozdělit existující dataframe na objekt s nezávislými proměnnými a objekt se závislou proměnnou. Tyto dva objekty dále rozdělíme ještě do dvou dalších kategorií - trénovací a testovací, viz 2.2.2. Trénovací bude obsahovat data určená k výstavbě modelu, zatímco pomocí testovací později ohodnotíme, jak dobrou predikční schopnost model má.

Výběr proměnných

Nejprve provedeme výběr proměnných na základě dříve popsané RFE metody. Tato funkce zvolila jako optimální množství proměnných 15, zatímco 25 označila jako nedůležité. Vybrané proměnné jsou následující:

'LotArea', 'OverallQual', 'OverallCond', '1stFlrSF', 'GrLivArea', 'BsmtFullBath', 'FullBath', 'HalfBath', 'KitchenAbvGr', 'GarageCars', 'BsmtQual',

'KitchenQual', 'GarageQual', 'HeatingQC', 'ExterQual'

Sestavení modelu

Tyto proměnné nyní můžeme vložit do modelu lineární regrese pomocí příkazu `LinearRegression().fit(X_train,y_train)`, přičemž poměr trénovacího a testovacího souboru stanovíme jako 4 : 1. Následujícím příkazem model vytvoříme a necháme si zobrazit veličiny spolu s jejich parametry.

```
X_train, X_test, y_train, y_test = train_test_split(data[promenne],
    y,test_size=0.2, random_state=100)

model = LinearRegression()
model.fit(X_train, y_train)
odhadXtest = model.predict(X_test)

df_coefs = pd.DataFrame([model.coef_][0], data[promenne].columns,
    columns = ['Koeficienty'])
df_coefs.sort_values(by='Koeficienty',ascending=False)
```

Proměnné seřazené od nejvyššího koeficientu nalezneme v následující tabulce.

Proměnné	Koeficienty	Proměnné	Koeficienty
GrLivArea	0,214850	BsmtQual	0,053378
1stFlrSF	0,203420	KitchenQual	0,045601
LotArea	0,084291	ExterQual	0,037257
BsmtFullBath	0,76257	OverallCond	0,037012
HalfBath	0,073462	HeatingQC	0,025720
FullBath	0,073190	GarageQual	0,011915
OvearallQual	0,071939	KitchenAbvGr	-0,236968
GarageCars	0,063011		

Tabulka 3.4: Koeficienty lineárního modelu

Konstantní parametr získáváme funkcí `model.intercept()` jako 6.890692004847792. Finální model lze tedy zapsat v podobě následující rovnice:

$$\begin{aligned}\ln(y+1) =& 0,214850\ln(GrLivArea+1) + 0,203420\ln(1stFlrSF+1) + \\ & 0,084291\ln(LotArea+1) + 0,76257(BsmtFullBath) + \\ & 0,073462(HalfBath) + 0,073190(FullBath) + \\ & 0,071939(OverallQual) + 0,063011(GarageCars) + \\ & 0,053378(BsmtQual) + 0,045601(KitchenQual) + \\ & 0,037257(ExterQual) + 0,037012(OverallCond) + \\ & 0,025720(HeatingQC) + 0,011915(GarageQual) - \\ & 0,236968(KitchenAbvGr) + 6,890692004847792\end{aligned}$$

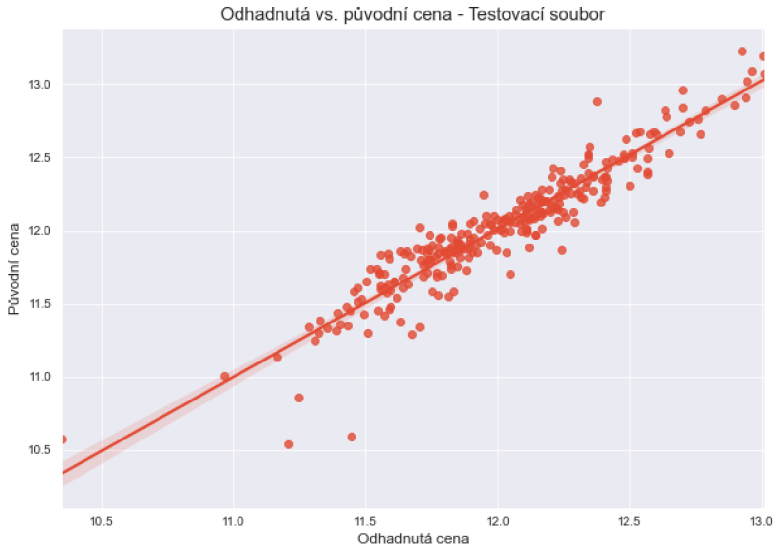
Evaluace

Dříve jsme si stanovili jako kritérium kvality lineárního modelu statistiky R^2 a R^2_{adj}. Jejich hodnotu pro trénovací a testovací soubor zvlášť získáme pomocí příkazu `model.score()` takto.

- R^2 trénovacího souboru : 0,8663964149621939
- R^2 testovacího souboru : 0,8906012827973073
- R^2_{adj} trénovacího souboru : 0,8646567849486808
- R^2_{adj} testovacího souboru : 0,8846557003406392

S hodnotami těchto skóre můžeme být poměrně spokojeni. Vzhledem k tomu, že jsou navíc na obou souborech velice blízké, lze předpokládat, že náš model dobře pracuje s novými daty.

Vykresleme si nyní velikost reziduálních hodnot vzhledem k ceně. Jak víme, konstantnost rozptylu u reziduí je jedním z předpokladů lineárního modelu. Z grafu lze usoudit, že jejich rozptyl je poměrně konstantní a k větším odchylkám dochází zejména u krajních hodnot.



Obrázek 3.4: Graf odhadnutých vs. původních logaritmizovaných cen

Predikce

Pokusme se nyní podle našeho modelu odhadnout ceny nemovitostí z pozorování v testovací skupině X_test. Tato data nyní proložíme naším modelem a následně výsledek porovnáme s hodnotami uloženými v y_test. Odhady získáme pomocí následujícího příkazu.

```
odhadXtest = model.predict(X_test)
odhadXtren = model.predict(X_train)

real_y_test = np.expm1(y_test)
real_odhad = np.expm1(odhadXtest)
rozdil = abs(real_y_test - real_odhad)
porovnaniR = pd.DataFrame({'Původní cena': real_y_test,
    'Odhad ceny' : real_odhad, 'Rozdíl' : rozdil})
porovnaniR = porovnaniR.astype(int)
```

Všimněme si, že jsme provedli zpětnou transformaci pomocí exponenciální funkce. Porovnání hodnot odhadnutých a reálných pro náhodných 12 pozorování z testovacího souboru uvidíme v následující tabulce.

	Původní cena	Odhad ceny	Rozdíl
1436	120499	104110	16389
57	196499	205853	9353
780	176000	167439	8560
382	213499	207467	6032
1170	171000	136746	34253
726	221999	267617	45617
258	231500	213681	17818
888	267999	286745	18745
532	107500	94217	13282
1055	180000	169935	10064
1246	186500	190320	3820
191	184000	191986	7986

Tabulka 3.5: Porovnání odhadnutých a reálných cen

Zjistili jsme tedy, že cena nemovitosti je nejvíce závislá na velikosti obytné plochy a na pozemku. Důležitý je také počet koupelen a celková kvalita. Poněkud překvapivě náš model nepracuje s rokem stavby, jak by se dalo předpokládat. I přesto si model vedl na základě statistického zhodnocení poměrně dobře a můžeme ho považovat za úspěšný. Predikce odhalila po zpětné transformaci větší odchylky zejména u vyšších cen, tento fakt nicméně bude pravděpodobně souviset s extrémy cen v původním datovém souboru.

3.2 Model splácení úvěru

Dříve jsme si stanovili, že logistickou regresi využíváme k popsání vztahu binární závislé proměnné na proměnných nezávislých. Jako tuto binární proměnnou můžeme chápat právě řádné splacení či nesplacení úvěru. Opět budeme zkoumat důležité proměnné, které splácení ovlivňují, a také se pokusíme využít vytvořený model k předpovědi, zda daný subjekt splatí či nesplatí úvěr.

3.2.1 Data

Data v tomto případě opět pochází z veřejně přístupné databáze dat na stránkách Kaggle.com (ke stažení zde). Jedná se původně o data z americké stránky Lending Company, která se zabývá nebankovními půjčkami. Datový soubor se znovu nachází ve formátu csv a obsahuje celkem 39717 pozorování a 88 proměnných. Jeho obsahem je informace o stavu splacení úvěrů a parametry s těmito úvěry související. Seznam proměnných a jejich popis je opět k nalezení v příloze C.1.

3.2.2 Úprava dat

Data si opět načteme pomocí dříve uvedeného příkazu `pd.read_csv()`, který náš soubor uloží do objektu `dataframe`. Jako první si po načtení všímáme, že závislá proměnná nabývá tří hodnot:

- 'Fully Paid', tedy splacená
- 'Current', tedy probíhající
- 'Charged Off', tedy nesplacená

Nás budou zajímat pouze krajní dvě hodnoty, všechny řádky s hodnotou 'Current' tedy odstraníme funkcí `.drop()`. Zbývající dvě hodnoty si transformujeme do podoby

- 'Fully Paid' = 1
- 'Charged Off' = 0.

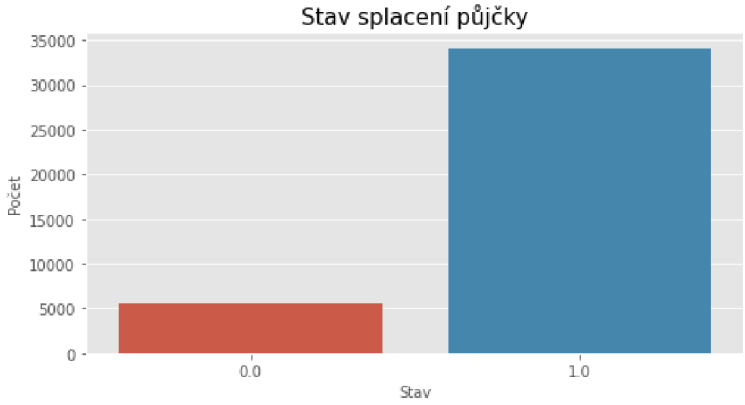
Další úpravou, kterou provedeme, je převedení proměnných typu *string* do číselné podoby, a to odstraněním znaků pro procenta a měsíce. K tomuto účelu lze využít funkce `.str.strip()`.

Všechny proměnné máme nyní v číselném vyjádření, a můžeme se tedy podívat na chybějící hodnoty. Po zavolání funkce `nul()` zjišťujeme, že nám chybí údaje celkem u 61 proměnných. Odstraníme tedy proměnné s více než 85% chybějících hodnot. Zbytek opět nahradíme mediánem. Několik sloupců navíc obsahuje pouze jedinou hodnotu, i tyto sloupce tedy pro jejich nízkou vypovídající hodnotu odstraníme. Celý proces provedeme následujícím příkazem.

```
data.dropna(axis=1, how='any', thresh=len(data.index)*0.25,
    inplace=True)
data.drop(columns=data.columns[data.nunique()==1], inplace=True)
data.fillna(data.median(),inplace=True)
```

Odstraníme dále také proměnné, které souvisí pouze s nesplacenými úvěry, a pro tvorbu modelu jsou tedy nepodstatné. Jedná se o následující proměnné: *'recoveries','collection_recovery_fee','out_prncp_involvement'*.

Po této počáteční úpravě nám zbylo 24 proměnných. Nejprve se znovu podíváme na závislou proměnnou, tedy *'loan_status'*. Po vykreslení sloupcového grafu příkazem `sns.countplot()` vidíme následující rozložení hodnot.



Obrázek 3.5: Počty splacených a nesplacených úvěrů

Vidíme tedy, že většina našich pozorovaní patří ke splaceným úvěrům, konkrétně jich je 85.83%. Podívejme se nyní na vztah mezi splácením úvěru a nezávislými proměnnými. Nejprve zjistíme míru vzájemné míry korelace pomocí Spearmanova koeficientu, který kvantifikuje i nelineární vztahy. K tomuto účelu použijeme znovu funkci `.corr()` a jako parametr zadáme `method='spearman'`. Nejvíce korelované proměnné se stavem splacení nalezneme v následující tabulce.

total_rec_prncp	0,404776
last_pymnt_amnt	0,307108
total_pymnt	0,293827
int_rate	-0,190623
total_rec_late_fee	-0,186332

Tabulka 3.6: Proměnné nejvíce korelované s se stavem splacení

Nejvyšší míru korelace vidíme u proměnné vyjadřující celkovou vypůjčenou částku za všechny úvěry, dále u velikost poslední splátky nebo u již splacené částky. Zároveň si všímáme, že čím je vyšší úroková míra, tím počet nesplacených úvěrů lehce stoupá. Stejně tak pokud klientu byly někdy účtovány poplatky za pozdní splacení, tím spíše byl jeho úvěr nesplacen.

Podívejme se také na korelaci mezi nezávislými proměnnými. Pro vizualizaci lze opět použít heatmapa, popř. můžeme pokračovat tabulkou 20 nejvíce korelovaných dvojic pomocí příkazu:

```
mat = data.drop('loan_status',axis=1).corr().abs()
nejkorDvojice = mat.where(np.triu(np.ones(mat.shape),k=1).astype
    (np.bool)).stack().sort_values(ascending=False)
nejkorDvojice.head(20)
```

Výstup nalezneme v příloze D.1. Zjišťujeme, že mezi proměnnými se nachází velké množství korelace. Nejvíce zastoupeny mezi korelovanými dvojicemi jsou následující proměnné, které jsme se proto rozhodli odstranit.

'total_pymnt','funded_amnt','installment'.

Původní počet proměnných jsme nyní eliminovali z 88 na 21 a můžeme začít s výstavbou modelu.

3.2.3 Tvorba modelu

Výběr proměnných

Pomocí dříve definované funkce `RFE_prom()` zjistíme optimální počet proměnných . Jejím výstupem je následujících 11:

'loan_amnt', 'funded_amnt_inv', 'dti', 'inq_last_6mths',
'mths_since_last_delinq', 'out_prncp', 'total_rec_prncp',
'total_rec_late_fee', 'last_pymnt_amnt', 'revol_util', 'term'

Sestavení modelu

Vybrané proměnné nyní vložíme do modelu logistické regrese pomocí funkce `LogisticRegression().fit()`. Celý příkaz, který nám zároveň vypíše i tabulku proměnných s koeficienty, je následující.

```
X_train, X_test, y_train, y_test = train_test_split(data[promenne],
y,test_size=0.25, random_state=10)

model = LogisticRegression()
model.fit(X_train, y_train)

predictionsTest = model.predict(X_test)
predictionsTrain = model.predict(X_train)

df_coefs = pd.DataFrame(model.coef_[0], index=data
```

```
    [promenne].columns,
columns = ['Koeficienty'])
df_coefs.sort_values(by='Koeficienty',ascending=False)
```

Získáváme tak následující tabulku parametrů spolu s jejich odhady.

Proměnné	Koeficienty	Proměnné	Koeficienty
out_prncp	0,105118	loan_amnt	-0,000922
term	0,031970	funded_amnt_inv	-0.001590
mths_since_last_delinq	0,011599	revol_util	-0,001610
dti	0,004099	inq_last_6mths	-0,002245
total_rec_prncp	0,002846	total_rec_late_fee	-0,059347
last_pymnt_amnt	0,001052		

Tabulka 3.7: Koeficienty logistického modelu

Vidíme tedy, že nejdůležitějšími parametry byla zvolena zbývající nesplacená částka, období splátek nebo výše poplatků za pozdní splácení.

Konstantní parametr získáváme funkcí `model.intercept_` jako $0,00100289456$

Výsledný model splácení úvěru, kde $l\_s = loan\_status$ jsme tedy nalezli jako:

$$\begin{aligned}\ln\left(\frac{P(l\_s=1)}{1-P(l\_s=1)}\right) =& 0,105118(out\_prncp) + 0,031970(term)\\ &+ 0,011599(mths\_since\_last\_delinq) + 0,004099(dti)\\ &+ 0,002846(total\_rec\_prncp) + 0,001052(last\_pymnt\_amnt)\\ &- 0,000922(loan_amnt) - 0,001590(funded\_amnt\_inv)\\ &- 0,001610(revol\_util) - 0,002245(inq\_last\_6mths)\\ &- 0,059347(total\_rec\_late\_fee) + 0,00100289456\end{aligned}$$

Evaluace

Uvedli jsme, že jedním z hodnotících kritérií logistických modelů je ROC křivka, tu si v prostředí Pythonu můžeme vykreslit za použití funkce `roc_curve()`. Příkaz, který ji vykreslí je následující.

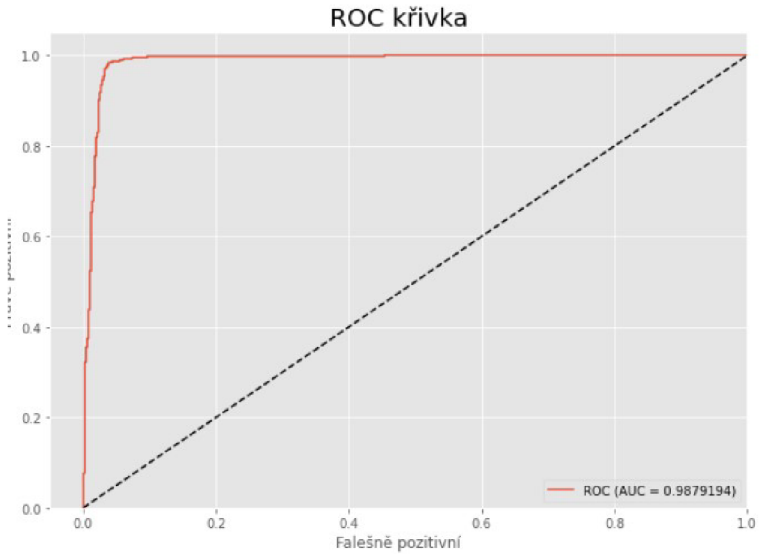
```
probPreds=model.predict_proba(X_test)
plt.figure(figsize=(10,7))

d, h, thr = roc_curve(y_test, probPreds[:,1])
roc = auc(d, h)

plt.plot(d, h, label='ROC (AUC = %0.4f)' % roc)
plt.plot([0, 1], [0, 1], 'k--')
```

```
plt.xlim([-0.05, 1.0])
plt.ylim([0.0, 1.05])
```

Vykreslená křivka má následující podobu.



Obrázek 3.6: ROC křivka modelu splacení úvěru

Hodnota AUC byla určena jako 0,988, což je velmi vysoká hodnota, její vypovídající hodnota ovšem není jednoznačná. Pravděpodobně hodnotu ovlivnil nepoměr mezi počty splacených a nesplacených úvěrů. Přesto se můžeme dále podívat na další evaluaci tohoto modelu.

Podívejme se nyní na klasifikační tabulku. Pro cut-off hodnotu 0,5 ji získáme pomocí funkce `confusion_matrix`.

	Pozitivní pozorované	Negativní pozorované
Pozitivní odhadnuté	8484	23
Negativní odhadnuté	142	1281

Vidíme, že model úspěšně odhaduje skutečné pozitivní hodnoty. U negativních hodnot má o něco větší problémy. Nižší hodnotu lze opět přiřadit faktu, že v původním souboru se nacházelo velice málo záznamů obsahují nesplácení úvěru, model tedy neměl dostatečné podklady.

Můžeme si dále stanovit hodnoty přesnosti, specifity a senzitivity jako:

- přesnost = 0,8686
- senzitivita = 0,9835
- specifita = 0,9824

Predikce

Pro předpověď znovu využijeme testovací pozorování uložená v proměnné X_test. Tyto hodnoty vložíme do vytvořeného modelu a porovnáme je se skutečnými výsledky známé z proměnné y_test. Pro tyto účely využijeme následující příkaz:

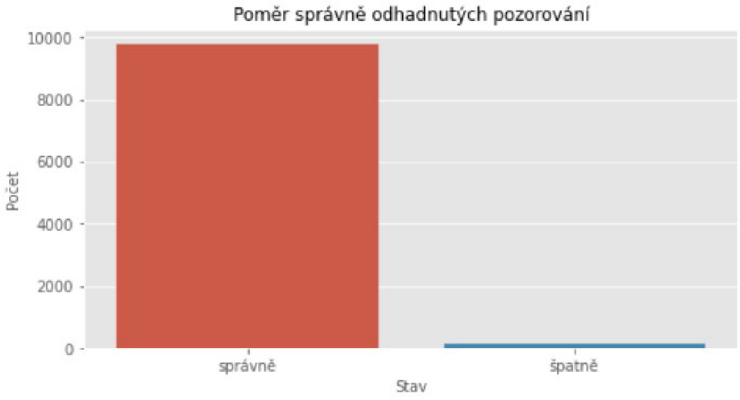
```
diff = abs(y_test - predictionsTest)

compare_actual = pd.DataFrame({'Původní splacení': y_test,
  'Odhadnuté splacení' : predictionsTest,
  'Správnost odhadu' : diff}).astype(int)
compare_actual['Správnost odhadu']=
  compare_actual['Správnost odhadu'].replace({0:'správně',
    1:'špatně'})
```

Prvních 15 výstupů nalezneme v tabulce na následující straně, celkový počet správně a špatně odhadnutých hodnot nalezneme na následujícím grafu.

ID	Původní hodnota	Odhad hodnoty	Rozdíl
28618	0	1	správně
35755	0	0	správně
30621	1	1	správně
14643	0	0	správně
38938	1	1	správně
9681	1	1	správně
1487	1	1	správně
11188	1	1	správně
30661	0	0	správně
13510	1	1	správně

Tabulka 3.8: Porovnání odhadnutých a reálných hodnot splácení



Obrázek 3.7: Porovnání počtu správně a špatně odhadnutých pozorování

Náš model odhadu splácení si vedl na základě zhodnocení relativně dobře. Zejména hodnota AUC byla velmi vysoká. Jak jsme uvedli, je pravděpodobné, že výsledek byl ovlivněn nepoměrem mezi pozorováními pro splacení a neplacení úvěru. Lze tedy říci, že pro predikci daných data je model vhodný, s novým datovým souborem obsahujícím více nesplacených úvěrů by ovšem mohl mít problém.

Závěr

Práce si kladla za cíl popsat a aplikovat dva modely regresní analýzy. Byli jimi lineární a logistický model. Nejprve jsme si v první části oba modely popsali z matematického hlediska. Jako první jsme se věnovali lineárnímu regresnímu modelu, u nějž jsme si stanovili jeho základní vlastnosti. Ukázali jsme si také postup nalezení hledaných parametrů modelu pomocí metody nejmenších čtverců. Zjistili jsme také, že lineární model je omezen mnoha podmínkami, patří mezi ně lineární závislost mezi regresory a regresandem, nezávislost regresorů nebo konstantnost rozptylu reziduí. Dále jsme si stanovili metody ohodnocení kvality vytvořeného modelu. Následně jsme se přesunuli ke krátkému pojednání o zobecněných lineárních modelech. Ty se liší od obyčejné lineární regrese zejména tím, že nepředpokládají lineární závislost mezi nezávislými a závislou proměnnou. My jsme se blíže zabývali pouze logistickým modelem. Ten pracuje na rozdíl od lineární regrese s diskrétní závislou proměnnou. V našem případě jsme si vyložili pouze případ, kdy závislá proměnná nabývá pouze dvou hodnot, kterým můžeme přiřadit hodnoty 0 a 1. Jedná se tedy o model odhadující pravděpodobnost, zda jistá skutečnost nastane nebo nenastane. Opět jsme si popsali vlastnosti tohoto modelu a nastínili postup hledání parametrů pomocí metody maximální věrohodnosti. Nakonec jsme si stanovili několik nejpoužívanějších metod pro ohodnocení kvality modelu, mezi které můžeme zařadit například ROC křivku a klasifikační tabulku.

Následně jsme si krátce představili programovací jazyk Python. Zjistili jsme, že se jedná o populární jazyk pro vědecké programování a modelování. Popsali jsme také rozšiřující knihovny tohoto jazyka, které jsou pro správnou aplikaci regresních modelů nezbytné. Zjistili jsme, že mezi nejdůležitější patří zejména knihovny pandas, díky kterým můžeme pracovat s datovými soubory a `scikit-learn`, který slouží právě k vytváření modelů a predikcím na nich založených. Ukázali jsme si také obecný postup tvorby modelu v Pythonu, který sestává z prvotní úpravy dat, tvorby modelu a nakonec evaluace vytvořeného modelu. Nakonec jsme si v poslední části předvedli reálnou aplikaci dříve popsaných modelů na reálných datech z oblasti trhu nemovitostí a úvěrů. Jako první jsme se pokusili vytvořit model, který stanoví nejdůležitější parametry ovlivňující cenu nemovitosti a následně ji předpoví. Podařilo se nám vytvořit poměrně úspěšný model, díky kterému jsme zjistili, že pro odhad ceny je důležitá zejména velikost obytné části, velikost pozemku či počet koupelen. Následně jsme se zabývali splatností úvěrů a parametrů, které ji ovlivňují. Tento model se potýkal s problémy nerovnovážného poměru mezi hodnotami splacení a nesplacení. Dokázali

jsme přesto vytvořit model který pro daná data poměrně úspěšně dokázal předpovědět splácení zejména na základě období splátek či výše poplatků za pozdní splácení. Zároveň jsme si během tvorby modelů ukázali vhodné příkazy v prostředí jazyka Python.

Literatura

[1] Bingham, N. H. a Fry, J. M. *Regression*. Springer London. Springer Undergraduate Mathematics Series. ISBN 978-1-84882-968-8 978-1-84882-969-5.

[2] Brownlee, J. *Data Preparation for Machine Learning: Data Cleaning, Feature Selection, and Data Transforms in Python*. Machine Learning Mastery, 2020.

[3] Drábek, P. a Míka, S. *Matematická analýza II*. Plzeň: Západočeská univerzita, 2003. ISBN 978-80-7082-977-6.

[4] Greene, W. H. *Econometric analysis*. 5th ed. Prentice Hall. ISBN 978-0-13-066189-0.

[5] Hilbe, J. M. *Practical Guide to Logistic Regression*. Taylor Francis Group, LLC. ISBN 978-1-4987-0958-3.

[6] McKinney, W. *Python for Data Analysis*. 2nd ed. O'Reilly Media, Inc. ISBN 978-1-491-95766-0.

[7] Montgomery, D. C., Peck, E. A. a Vining, G. *Introduction to Linear Regression Analysis*. 5th edition. Wiley. ISBN 978-0-470-54281-1.

[8] Müller, A. C. a Guido, S. *Introduction to Machine Learning with Python*. 1st ed. O'Reilly Media, Inc. ISBN 978-1-449-36941-5.

[9] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B. et al. Scikit-learn: Machine learning in Python. *Journal of machine learning research*. 2011, sv. 12, Oct, s. 2825–2830.

[10] Rawat, A. S. What is Regression Analysis? Types and Applications Analytics Steps. Dostupné z: https://www.analyticssteps.com/blogs/what-regression-analysis-types-and-applications.

[11] Rencher, A. C. a Schaalje, G. B. *Linear models in statistics*. 2nd ed. Wiley-Interscience. ISBN 978-0-471-75498-5. OCLC: ocn144331522.

[12] Wooldridge, J. M. *Introductory Econometrics: A Modern Approach*. 5th ed. South-Western Cengage Learning. ISBN 978-1-111-53104-1.

[13] Zvára, K. *Regrese*. Druhé vydání. Praha: Matfyzpress. ISBN 978-80-7378-406-5.

Příloha A

Název	Popis	Hodnoty
Id	ID	číslo do 1460
MSSubClass	Druh budovy	druh dle velikosti a počtu pater
LotFrontage	Část ulice sousedící s pozemkem	ve stopách
LotArea	Pozemek	ve stopách čtverečních
OverallQual	Celková kvalita	hodnoty 1-10 (nejlepší)
OverallCond	Celkový stav	hodnoty 1-10 (nejlepší)
YearBuilt	Rok postavení	údaj v letech
YearRemodAdd	Rok rekonstrukce	údaj v letech
MasVnrArea	Povrch fasády	ve stopách čtverečních
BsmtFinSF1	Rozloha dokončeného suterénu	ve stopách čtverečních
BsmtUnfSF	Rozloha nedokončeného suterénu	ve stopách čtverečních
TotalBsmtSF	Celková rozloha suterénu	ve stopách čtverečních
1stFlrSF	Rozloha prvního patra	ve stopách čtverečních
2ndFlrSF	Rozloha druhého patra	ve stopách čtverečních
LowQualFinSF	Rozloha nedokončené stavby	ve stopách čtverečních
GrLivArea	Rozloha mimo suterén	ve stopách čtverečních
BsmtFullBath	Počet koupelen se sprchou v suterénu	v celých číslech
BsmtHalfBath	Počet koupelen beze sprchy v suterénu	v celých číslech
FullBath	Počet koupelen se sprchou	v celých číslech
HalfBath	Počet koupelen beze sprchy	v celých číslech
BedroomAbvGr	Počet ložnic mimo suterén	v celých číslech
KitchenAbvGr	Počet kuchyní mimo suterén	v celých číslech
TotRmsAbvGrd	Počet pokojů mimo suterén	v celých číslech
Fireplaces	Počet krbů	v celých číslech
GarageYrBlt	Rok stavby garáže	v letech
GarageCars	Počet míst pro auta	v celých číslech
GarageArea	Rozloha garáže	ve stopách čtverečních
WoodDeckSF	Rozloha terasy	ve stopách čtverečních
OpenPorchSF	Rozloha otevřené verandy	ve stopách čtverečních
EnclosedPorch	Rozloha uzavřené verandy	ve stopách čtverečních

3SsnPorch	Rozloha prosklené verandy	ve stopách čtverečních
ScreenPorch	Rozloha polootevřené verandy	ve stopách čtverečních
PoolArea	Rozloha bazénu	ve stopách čtverečních
MiscVal	Různé	doplňující údaj o nadstandardním vybavení
MoSold	Měsíc prodeje	v měsících
YrSold	Rok prodeje	v letech
SalePrice	Cena	v USD
BsmtQual	Kvalita sklepa	Ex = výborný Gd = dobrý TA = průměrný Fa = Dostačující Po = špatný NA = chybí
BsmtCond	Stav sklepa	
FireplaceQu	Kvalita krbu	
KitchenQual	Kvalita kuchyně	
GarageQual	Kvalita garáže	
GarageCond	Stav garáže	
HeatingQC	Kvalita topení	
ExterCond	Stav exteriéru	
ExterQual	Kvalita exteriéru	

Příloha B

```
skore=[]
var=[]
def RFEProm():
    for num in range (1,len(data.columns)):
        model = Logistic(Linear)Regression()
        rfe = RFE(model, num)
        fit = rfe.fit(X, y)

        vystup = pd.DataFrame(fit.support_, index=X.columns,
            columns = ['Koeficienty'])

        koef=vystup[vystup[vystup==True]].dropna()

        X_train, X_test, y_train, y_test = train_test_split(
            data[koef.index], y,test_size=0.2, random_state=100)

        sc_X = StandardScaler()
        X_train = sc_X.fit_transform(X_train)
        X_test = sc_X.transform(X_test)

        model.fit(X_train, y_train)
        var.append([koef.index])
        skore.append([num,model.score(X_test,y_test)])

    print('Počet vybraných proměnných je:',max(skore,
        key=lambda x: x[1])[0], 'a skóre testovacího souboru je:',
            max(skore,key=lambda x: x[1])[1])

    global promenne
    promenne=[item for sublist in var[(max(skore, key=lambda x:
        x[1])[0])-1] for item in sublist]
    print('Vybrané proměnné jsou:',promenne)

RFEProm()
```

Příloha C

Název	Popis
loan_amnt	výše úvěru
funded_amnt	financovaná částka
funded_amnt_inv	financovaná částka investory
installment	výše měsíční splátky
annual_inc	roční příjem žadatele o úvěr
dti	ukazatel Debt to Income
delinq_2yrs	případy nesplácení za dva roky
inq_last_6mths	kontroly kredi. skóra za 6 měsíců
mths_since_last_delinq	měsíce od posl. příp. nesplácení
mths_since_last_record	měsíce od kontroly veř. záznamů
open_acc	počet současných úvěrů
pub_rec	negativní veřejné záznamy
revol_bal	bilance revolvingových úvěrů
total_acc	počet kreditních linek klienta
out_prncp	nesplacená část jistin
out_prncp_inv	nesplacená část jistin financ. investory
total_pymnt	splacená částka
total_rec_prncp	celková vypůjčená částka
total_rec_int	výše úroků
total_rec_late_fee	výše poplatků za pozdní splácení
recoveries	návrat za nesplacené úvěry
collection_recovery_fee	poplatky za nesplacený úvěr
last_pymnt_amnt	výše poslední splátky
collections_12_mths_ex_med	počet vymožených dluhů za rok
mths_since_last_major_derog	měsíce od posl. negativního záznamu
policy_code	označení veřejné přístupnosti půjčky
annual_inc_joint	roční příjem spolužadatelů
dti_joint	ukazatel Debt to Income spolužadatelů
verification_status_joint	ukazatel ověření přijmů spolužadatelů
acc_now_delinq	počet aktuálních úvěrů s delikvencí
tot_coll_amt	výše vymožených dluhů
tot_cur_bal	současná bilance všech účtů
open_acc_6m	počet úvěrů za 6 měsíců
open_il_6m	počet úvěrů na splátky za 6 měsíců
open_il_12m	počet úvěrů na splátky za 12 měsíců
open_il_24m	počet úvěrů na splátky za 24 měsíců
mths_since_rcnt_il	měsíce od uzavření posledního úvěru na splátky

total_bal_il	bilance úvěrů na splátky
il_util	poměr celk. bilance/celk. úvěr. limitu úvěrů na splátky
open_rv_12m	počet revolvingových úvěrů za rok
open_rv_24m	počet revolvingových úvěrů za 2 roky
max_bal_bc	nejvyšší v současnosti dlužená částka
all_util	poměr celkové bilance/celkového úvěrového limitu
total_rev_hi_lim	nejvyšší revolvingový úvěr/úvěrový limit
inq_fi	počet kontrol kreditního skóre
total_cu_tl	počet všech úvěrů
inq_last_12m	kontroly kreditního skóra za rok
acc_open_past_24mths	počet všech úvěrů za 2 roky
avg_cur_bal	průměrná současná bilance všech účtů
bc_open_to_buy	zůstatek na všech kreditních účtech
bc_util	současná celková bilance/úvěrový limit kreditních úvěrů
chargeoff_within_12_mths	počet nesplacených úvěrů za rok
delinq_amnt	dlužná částka za všechna pozdní splácení
mo_sin_old_il_acct	měsíce od schválení nejstaršího úvěru na splátky
mo_sin_old_rev_tl_op	měsíce od schválení nejstaršího revol. úvěru
mo_sin_rcnt_rev_tl_op	měsíce od schválení posledního revol. úvěru
mo_sin_rcnt_tl	měsíce od schválení posledního úvěru
mort_acc	počet hypoték
mths_since_recent_bc	měsíce od otevření posl. kreditního účtu
mths_since_recent_bc_dlq	měsíce od posl. delikvence na kreditní. účtu
mths_since_recent_inq	měsíce od posl. kontroly kreditního skóre
mths_since_recent_revol_delinq	měsíce od posl. delikvence na revolv. účtu
num_accts_ever_120_pd	počet úvěrů se zpožděním ve splácení min. 120 dní
num_actv_bc_tl	počet současně otevřených kreditních účtů
num_actv_rev_tl	počet současných revolv. úvěrů
num_bc_sats	počet kredit. úvěrů bez problémů se splácením
num_bc_tl	počet všech kreditních účtů
num_il_tl	počet všech úvěrů na splátky
num_op_rev_tl	počet právě otevřených revol. účtů
num_rev_accts	počet všech revolv. účtů
num_rev_tl_bal_gt_0	počet revolv. účtů ze kterých je právě čerpáno
num_sats	počet úvěrů bez problémů se splácením
num_tl_120dpd_2m	počet úvěrů se zpožd. ve splác. min. 120 dní (posl. 2 měs.)
num_tl_30dpd	počet úvěrů se zpožd. ve splác. min. 30 dní (posl. 2 měs.)
num_tl_90g_dpd_24m	počet úvěrů se zpožd. ve splác. min. 90 dní (posl. 2 roky.)
num_tl_op_past_12m	počet úvěrů za rok
pct_tl_nvr_dlq	procento úvěrů bez delikvence
percent_bc_gt_75	procento kredit. úvěrů s čerpáním min 75% z limitu
pub_rec_bankruptcies	počet osobních bankrotů
tax_liens	počet exekucí
tot_hi_cred_lim	celkový úvěrový limit
total_bal_ex_mort	celková kreditní bilance bez hypoték
total_bc_limit	celkový limit kreditních úvěrů
total_il_high_credit_limit	nejvyšší vypůjčená částka na splátky /úvěrový limit
revol_util	poměr úvěrového využití
term	délka splátkového období
int_rate	úroková míra
loan_status	stav půjčky

Příloha D

loan_amnt	funded_amnt	0.981578
total_pymnt	total_rec_prncp	0.971472
funded_amnt	funded_amnt_inv	0.958422
	installment	0.956159
loan_amnt	funded_amnt_inv	0.940034
	installment	0.930288
funded_amnt_inv	installment	0.905039
funded_amnt	total_pymnt	0.903160
loan_amnt	total_pymnt	0.886613
funded_amnt_inv	total_pymnt	0.881228
funded_amnt	total_rec_prncp	0.870255
installment	total_pymnt	0.856928
loan_amnt	total_rec_prncp	0.852021
installment	total_rec_prncp	0.850773
funded_amnt_inv	total_rec_prncp	0.845848
pub_rec	pub_rec_bankruptcies	0.843032
total_pymnt	total_rec_int	0.828758
funded_amnt	total_rec_int	0.737469
funded_amnt_inv	total_rec_int	0.730914
loan_amnt	total_rec_int	0.729726
open_acc	total_acc	0.686635
total_rec_prncp	total_rec_int	0.684027
installment	total_rec_int	0.634725
total_rec_prncp	last_pymnt_amnt	0.543408
total_rec_int	int_rate	0.529913

Obrázek D.1: Porovnání korelovaných dvojic

Seznam obrázků

1.1 Znázornění rozkladu rozptylu 10
1.2 ROC křivka 19

3.1 Krabicový diagram závislé proměnné 30
3.2 Porovnání histogramů ceny před a po logaritmické transformaci . . . 31
3.3 Heatmapa veličin nejvíce korelovaných na cenu 33
3.4 Graf odhadnutých vs. původních logaritmizovaných cen 36
3.5 Počty splacených a nesplacených úvěrů 39
3.6 ROC křivka modelu splacení úvěru 42
3.7 Porovnání počtu správně a špatně odhadnutých pozorování 43

D.1 Porovnání korelovaných dvojic 54

Seznam tabulek

3.1 Statistické ukazatele proměnné SalePrice 30
3.2 Proměnné nejvíce korelované s cenou 32
3.3 Korelace mezi dvojicemi proměnných 34
3.4 Koeficienty lineárního modelu 35
3.5 Porovnání odhadnutých a reálných cen 37
3.6 Proměnné nejvíce korelované s se stavem splacení 39
3.7 Koeficienty logistického modelu 41
3.8 Porovnání odhadnutých a reálných hodnot splácení 43