Vysoké učení technické v Brně
Fakulta informačních technologií

MSP

Bayesovské odhady a regresia

17. mája 2024 Adam Zvara (xzvara01)

Projekt bol vypracovaný v prostredí Jupyter[1] notebooku.

Úloha 1 – Bayesovské odhady

a) Konjugované apriórne a aposteriórne rozdelenie, prediktívne rozdelenie

1. Apriórna a aposteriórna hustota

V prvom rade je potrebné načítať dáta z priloženého dokumentu do pandas[2] dataframu.

```
import pandas as pd
import numpy as np

# Load excel sheet into pandas dataframe
df = pd.read_excel('Projekt-2_Data.xlsx', sheet_name='Úloha 1')

# Dataframe used for a) section of task 1
df1 = df[['uloha_1 a)']].dropna().astype(int)
```

Náhodná veličina X pochádza z Poissonovho rozdelenia s parametrom λ. Zo zadania je známa apriórna informácia, že za každých 5ms by malo nastať 10 pripojení. Apriórnu pravdepodobnosť môžeme modelovať pomocou Gamma rozdelenia a na základe tabuliek uvedených v prednáškach dokážeme zistiť apriórnu a aposteriórnu hodnotu parametru λ.

$$
\begin{aligned}
h(\lambda) &= \Gamma(\alpha, \beta) \\
k(\lambda|\mathbf{x}) &= \Gamma(\alpha', \beta') \\
\alpha' &= \alpha + \sum_{i}^{n} x \\
\beta' &= \beta + n
\end{aligned}
$$

Na základe tabuliek[3] a apriórnej informácie môžeme zvoliť hyperparametre $\alpha = 10$ a $\beta = 5$.

```
from scipy.stats import gamma, nbinom
import matplotlib.pyplot as plt

x = np.linspace(1, 4, 100)
a = 10
b = 5

ax.plot(x, gamma.pdf(x, a, scale=1/b), label='aprior')
ax.plot(x, gamma.pdf(x, a + df1.sum(), scale=1/(b + len(df1))), label='aposterior')
```

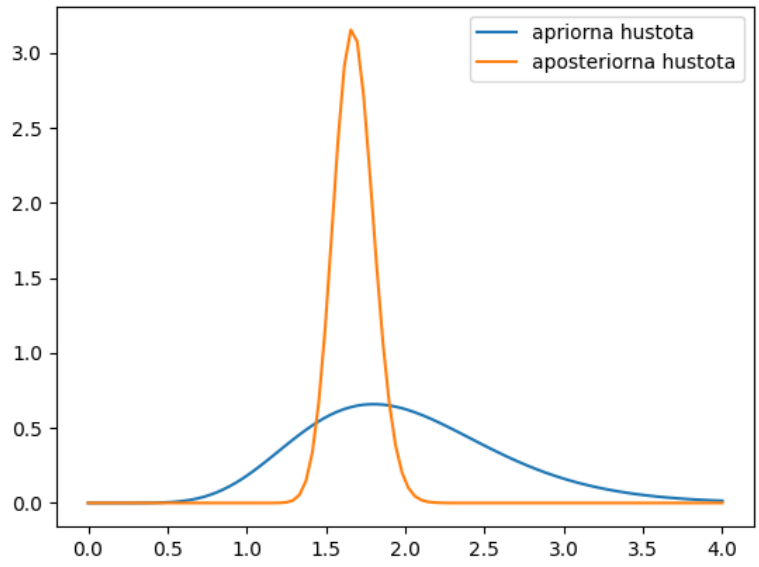
z čoho dostávame apriórnu a aposteriórnu hustotu parametru λ Poissonoveho rozdelenia na obrázku 1.

2. Apriórna a aposteriórna prediktívna hustota

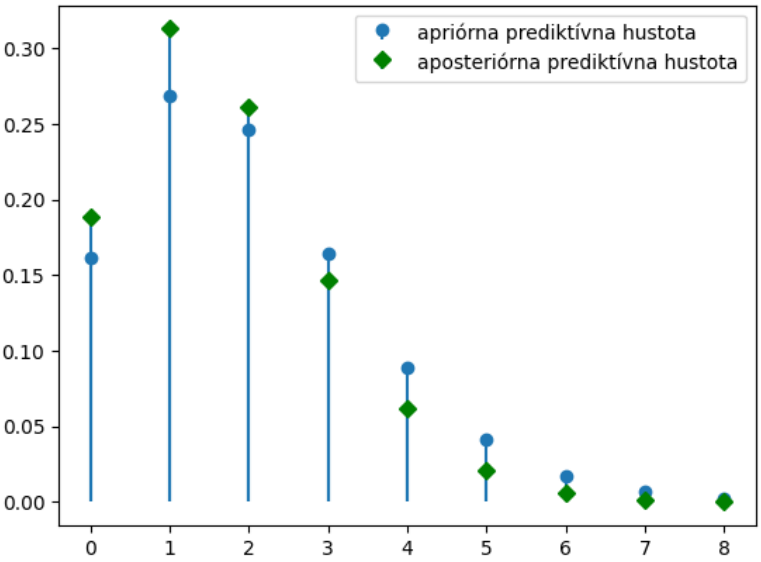
Podobne z tabuliek vieme vyčítať, že aposteriórna prediktívna hustota patrí do $NB(\tilde{x} \,|\, \alpha', \frac{\beta'}{1+\beta'})$ a apriórna prediktívna hustota do $NB(\tilde{x} \,|\, \alpha, \frac{\beta}{1+\beta})$. Teda môžeme vykresliť ich grafy, ktoré vidíme na obrázku 2.

```
x = np.arange(1, 9)
a2 = a + df1.sum()
b2 = b + len(df1)

nbinom_apost = nbinom(a2, b2 / (1 + b2))
nbinom_apri = nbinom(a, b / (1 + b))

ax.stem(x, nbinom_apri.pmf(x) ...)
ax.stem(x, nbinom_apost.pmf(x) ...)
```

[1] https://jupyter.org/
[2] https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html
[3] https://en.wikipedia.org/wiki/Conjugate_prior



Obr. 1: Apriórna a aposteriórna hustota parametru λ



Obr. 2: Apriórne a aposteriórne prediktívne hustoty za jeden časový interval

3. 95% interval spoľahlivosti parametru λ

Môžeme spočítať 95% interval spoľahlivosti pre parameter λ z apriórneho a aposteriórneho rozdelenia pomocou inverznej distribučnej funkcie jednotlivých aposteriórnych rozdelení [1]. Uvažujeme tzv. *equal-tailed* interval, ktorý môžeme spočítať ako:

```
apost_int = gamma.ppf(0.025, a2, scale=1/b2), gamma.ppf(0.975, a2, scale=1/b2)
apri_int = gamma.ppf(0.025, a, scale=1/b), gamma.ppf(0.975, a, scale=1/b)
```

z čoho získavame intervaly $\langle 0.9590, 3.4169\rangle$ pre apriórne rozdelenie a $\langle 1.4376, 1.9327\rangle$ pre aposteriórne rozdelenie. Môžeme vidieť, že interval pre apriórne rozdelenie je omnoho väčší, čo vyplýva z toho, že vychádzame z menšieho množstva informácií. Naopak interval aposteriorného rozdelenia berie do úvahy aj namerané hodnoty a preto je tento interval užší.

4. Dva aposteriórne bodové odhady parametru λ
Realizujeme dva aposteriorné bodové odhady parametru λ – teda bodové odhady z vychádzajúceho aposteriorného gamma rozloženia. Môžeme si vybrať bodové odhady strednej hodnoty a mediánu, ktoré spočítame následovne:

```
gamma_aposterior = gamma(a + df1.sum(), scale=1/(b + len(df1)))
aposterior_mean = gamma_aposterior.mean()
aposterior_median = gamma_aposterior.median()
```

Stredná hodnota = 1.6761
Medián = 1.6730

Vidíme, že bodové odhady strednej hodnoty a mediánu sú si veľmi podobné. Z toho môžeme usúdiť, že sa jedná o približne symetrické rozdelenie s jediným centrálnym vrcholom, ako môžeme vidieť na obrázku 1.

5. Jeden apriórny a jeden aposteriórny bodový odhad parametru λ
Podobne môžeme realizovať bodový odhad počtu pozorovaní z apriórnej a aposteriórnej prediktívnej hustoty.

```
apost_predict_mean = nbinom_apost.mean()
apri_predict_mean = nbinom_apri.mean()
```

z ktorého získavame apriórnej bodový odhad počtu pozorovaní 1.9999 a aposteriórny bodový odhad počtu pozorovaní 1.6761. Opäť môžeme vidieť, že odhad na základe apriórnej pravdepodobnosti sa líši od aposteriórneho odhadu v dôsledku pridania nových pozorovaní do aposteriórneho odhadu.

b) Aproximácia diskrétnym rozdelením

1. Apriórna, aposteriórna hustota a funkcia vierohodnosti
Keďže je apriorná informácia vo forme nameraných hodnôt a naším cieľom je odhadnúť aposteriornú hustotu parametru b, musíme si z apriorných dát vybrať tie, ktoré nám o parametri b niečo hovoria. Keďže je parameter b maximálna doba trvania procesu, vyberieme si z jednotlivých skupín meraní hodnotu s najdlhším trvaním procesu následovne.

```
bins = 100

prior = df["uloha_1 b)_prior"].groupby(df["skupina"]).max()
prior_hist, prior_bins = np.histogram(prior, bins=bins, density=True)
```

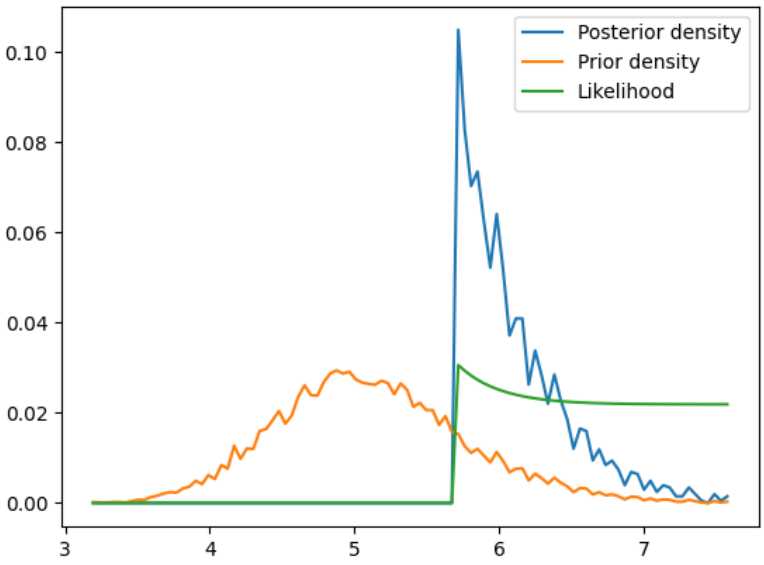
Likelihood funkcia udáva pravdepodobnosť výskytu *observation* dát pri dosadení konkrétnej hodnoty za parameter b. Hodnotu parametru b budeme skúmať na rovnakom intervale, ktorý je daný apriórnou hustotou (`prior_bins`). Na základe týchto obmedzení si môžeme vykresliť likelihood funkciu pomocou funkcie hustoty odseknutého normálneho rozdelenia, kde dosadíme naše pozorovania.

```
from scipy.stats import truncnorm

obs = df["uloha_1 b)_pozorování"].dropna()
x = np.linspace(prior_bins[0], prior_bins[-1], bins)  # our parameter b
likelihood = []                                       # our likelihood

for b in x:
    b_truncnorm = (b - 3)
    likelihood.append(truncnorm.pdf(obs, -2, b_truncnorm, 3, 1).prod())

likelihood = np.array(likelihood)
np.nan_to_num(likelihood, copy=False)
likelihood = likelihood / likelihood.sum()
```

teda v podstate počítame likelihood ako (*pdf* je *probability density function* odseknutého normálneho rozdelenia):

$$p(\mathbf{x} \mid b) = \Pi_i^n \, \mathrm{pdf}(x)$$

Nakoniec môžeme vypočítať aposteriórnu hustotu a vykresliť všetko v jednom grafe, ktorý je zobrazený na obrázku 3.



Obr. 3: Apriórna, aposteriórna hustota a likelihood funkcia

```
df2 = pd.DataFrame(prior_h, x, columns=["Prior"])
df2["Posterior"] = (prior_h * likelihood) / (prior_h * likelihood).sum()
df2["Likelihood"] = likelihood

df2.plot()
```

2. 95% interval spoľahlivosti parametru b
Intervalový odhad získame z aposteriórnej hustoty:

```
posterior = np.array(df2["Posterior"])

cumulative_sum = np.cumsum(posterior)
c0 = prior_bins[np.argmax(cumulative_sum > 0.025)]
c1 = prior_bins[np.argmax(cumulative_sum > 0.975) - 1]
print(f"95% conf. interval: {c0} - {c1}")
```

Intervalový odhad parametra b je $\langle 5.69, 6.96 \rangle$.

3. Dva bodové odhady parametru b
Za bodové odhady si môžeme zvoliť strednú hodnotu a medián.

```
nonzero = np.argwhere(posterior > 0)
mean = (prior_bins[nonzero].ravel() * posterior[nonzero].ravel()).sum()
median = prior_bins[np.argmax(cumulative_sum > 0.5) - 1]
print(f"mean: {mean}, median = {median}")
```

stredná hodnota: 6.07
medián: 5.91

Úloha 2 – Regresia

1. Vytvorenie rovnice modelu pomocou spätnej eliminácie
Pred vytvorením modelu načítame dáta do pandas dataframe a preskúmame ich.

```
df = pd.read_excel('Projekt-2_Data.xlsx', sheet_name='Úloha 2')
```

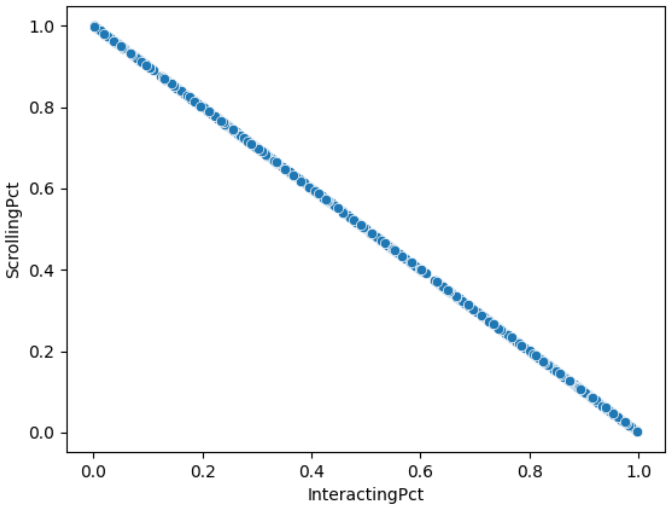
Vychádzame z plného kvadratického modelu, teda počítame do úvahy všetky interakcie druhého rádu a všetky druhé mocniny (X1-4 predstavujú nami skúmané vlastnosti).

$$Y = \beta_1 + \beta_2 X_1 + \beta_3 X_2 + \ldots + \beta_6 X_1 X_2 + \beta_7 X_1 X_3 + \ldots + \beta_{12} X_1^2 + \ldots$$

Ihneď si môžeme všimnúť, že stĺpec *ActiveUsers* sa pohybuje v iných rádoch jednotiek ako zvyšné spojité veličiny. Preto je nutné tento stĺpec normalizovať (napríklad na hodnoty z intervalu 0-1). Názvy stĺpcov boli zároveň vhodne premenované na pohodlnejšiu prácu s nimi.

```
df.rename(columns={
    "ActiveUsers": "users",
    "ScrollingPct": "scroll",
    "Ping [ms]": "ping",
    "InteractingPct": "inter",
    "OSType":"os"}, inplace=True)
df["users"] = (df["users"] - df["users"].min()) / (df["users"].max() - df["users"].min())
```

Po jednoduchom vykreslení si závislostí medzi jednotlivými stĺpcami (seaborn pairplot[4]) si môžeme všimnúť na obrázku 4 veľkú mieru lineárnej závislosti medzi scrollovaním a interakciou užívateľov.



Obr. 4: Závislosť medzi scrollovaním a interakciou užívateľov

Z toho vyplýva, že sa môžeme zbaviť jednej z týchto premenných – zvolil som premennú *InteractingPct* - teda zo základného modelu môžeme tento prediktor (a všetky jeho interakcie) vynechať. Zároveň môžeme vynechať kvadrát prediktoru *OSType*, keďže sa jedná o kategorickú premennú a druhá mocnina neprináša žiadne nové informácie. Z plného modelu sme sa teda dostali ku modelu s predpisom:

$$Y = \beta_1 + \beta_2 X_1 + \beta_3 X_2 + \beta_4 X_3 + \beta_5 X_1 X_2 + \beta_6 X_1 X_3 + \beta_7 X_2 X_3 + \beta_8 X_1^2 + \beta_9 X_2^2$$

kde X_1 je počet užívateľov, X_2 je percento scrollujúcich užívateľov a X_3 je použitý OS. Túto formulu môžeme použiť na vytvorenie modelu lineárnej regresie ako bolo prezentované na demonstračnom cvičení.

[4] https://seaborn.pydata.org/generated/seaborn.pairplot.html

```
UsersInt  = '+ users*scroll + users*C(os) '
ScrollInt = '+ scroll*C(os) '
Quadratic = '+ I(users**2) + I(scroll**2) '
formula = 'ping ~ users + scroll + C(os) ' + UsersInt + ScrollInt + Quadratic
model=smf.ols(formula=formula, data=df)
results=model.fit()
print(results.summary())
```

Tento model môžeme vylepšiť pomocou metódy spätnej eliminácie, pri ktorej sa pozeráme na p-hodnoty jednotlivých prediktorov a postupne mažeme také, ktoré nemajú štatisticky významný vplyv na predikciu (zvolíme si prahovú hodnotu p-value, typicky 0.05 alebo 0.1 a odstránime prediktory s p-hodnotou vyššou ako nami zvolená prahová hodnota). V prvej iterácii si môžeme všimnúť niekoľko takýchto prediktorov:

predictor	**p-value**
C(os)[T.MacOS]	0.415
C(os)[T.iOS]	0.981
users:C(os)[T.Windows]	0.013
scroll:C(os)[T.MacOS]	0.888
scroll:C(os)[T.Windows]	0.876
scroll:C(os)[T.iOS]	0.921
I(scroll ** 2)	0.287

Tabuľka 1: Kritické p-hodnoty prediktorov v prvej iterácii

Pri hodnotách kategorických premenných však musíme byť opatrní, keďže nie vždy ich môžeme jednoznačne vymazať. Napríklad v tabuľke 1 môžeme vidieť, že 2 hodnoty kategorickej premennej použitého OS nie sú štatisticky významné, no tretia môže byť významná. V takýchto prípadoch môžeme overiť vplyv kategorickej premennej ako celku na výsledný model pomocou združeného F-testu.

```
hypothesis='(C(os)[T.MacOS]=0),(C(os)[T.iOS]=0),(C(os)[T.Windows]=0)'
F_test=results.f_test(hypothesis)
```

P-hodnota tohto testu je 0.000844 a teda nezamietame nulovú hypotézu a typ OS má štatisticky významný vplyv v našom modeli. Môžeme sa však pozrieť na interakciu medzi scrollovaním a typom OS (všetky hodnoty majú p-value > 0.05). Opäť vytvoríme združený F-test na zistenie vplyvu tejto interakcie na predikcie.

```
h='(scroll:C(os)[T.MacOS]=0),(scroll:C(os)[T.iOS]=0),(scroll:C(os)[T.Windows]=0)'
F_test=results.f_test(h)
```

P-hodnota tohto testu je 0.98, čo je výrazne nad prahom 0.05. Túto interakciu teda môžeme eliminovať. Tabuľka problematických p-hodnôt sa zmenší na:

predictor	**p-value**
C(os)[T.MacOS]	0.415
C(os)[T.iOS]	0.981
users:C(os)[T.Windows]	0.013
I(scroll ** 2)	0.287

Tabuľka 2: Kritické p-hodnoty prediktorov v druhej iterácii

Ďalej môžeme eliminovať prediktor *scroll**2*, keďže je jeho p-hodnota nad prahovou hodnotou 0.05. Na základe metódy spätnej eliminácie už nemusíme žiadne prediktory eliminovať, keďže hodnoty všetkých prediktorov sú menšie ako prahová hodnota.

VIF hodnoty (určujúce mieru multikolinearity prediktorov) sú však pomerne veľké (*users* a *users**2* majú VIF hodnoty > 10), čo indikuje existenciu multikolinearity. Ďalším krokom je teda eliminácia tejto multikolinearity tak, že neuvažujeme prediktor *users**2*. Hodnota R^2 síce klesne (iba o 0.028) ale výsledkom je jednoduchší model, ktorý je lepšie interpretovateľný. V rámci experimentov som sa pokúšal odstrániť prediktor *users*, keďže model stále vykazoval známky multikolinerity a dospel som k záveru, že vymazanie tohto prediktoru nemá vplyv na R^2 hodnotu modelu a zároveň po jeho vymazaní sa hodnoty VIF znížia.

Rovnica finálneho modelu (po spätnej eliminácii a odstránení multikolinerity) je v tvare:

$$Y = \beta_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_1 X_2 + \beta_5 X_1 X_3$$

kde X_1 je počet aktívnych užívateľov, X_2 je percento scrollujúcich užívateľov a X_3 je použitý OS. Hodnoty parametrov (po odstránení "outliers" v ďalšej kapitole) môžeme vidieť v tabuľke 3.

Predictor	Coefficient
(β_1) Intercept	46.9796
(β_3) C(os)[T.MacOS]	-1.6013
(β_3) C(os)[T.Windows]	6.6933
(β_3) C(os)[T.iOS]	-2.0221
(β_2) scroll	-36.1880
(β_4) users:scroll	34.2056
(β_5) users:C(os)[Android]	19.3930
(β_5) users:C(os)[T.MacOS]	38.7120
(β_5) users:C(os)[T.Windows]	13.0970
(β_5) users:C(os)[T.iOS]	11.9400

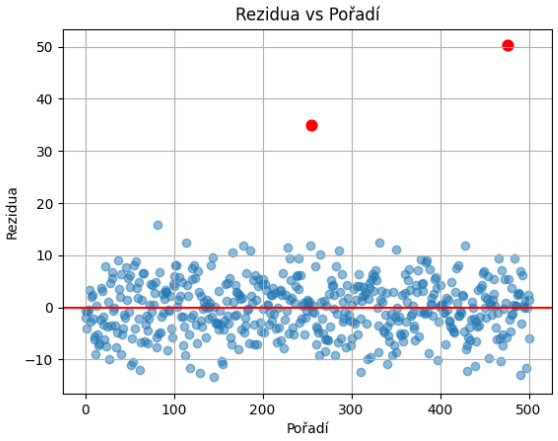
Tabuľka 3: Vypočítané koeficienty modelu lineárnej regresie

1c) Detekovanie a odstránenie odľahlých hodnôt

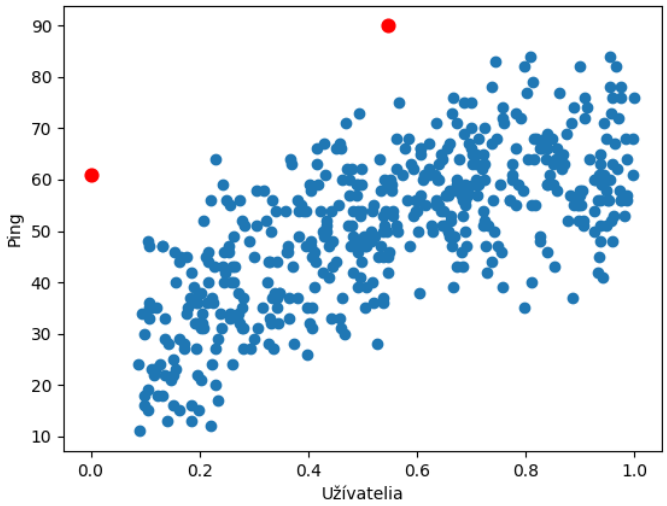
Po identifikácii vhdného modelu sa môžeme pozrieť na niektoré základné regresné diagnostiky. Pri vykreslení grafu rezíduí (obrázok 5) si môžeme všimnúť dve pomerne odľahlé hodnoty. Ich indexy môžeme zistiť prostredníctvom štandardizovaných rezíduí, pomocou ktorých vidíme potenciálne odľahlé hodnoty. V tabuľke 4 si môžeme všimnúť, že pre dva indexy sa štandardizované rezíduá pomerne výrazne vzdiaľujú od typickej hodnoty, s ktorou ich porovnávame (typicky porovnávame absolútnu hodnotu rezíduí s hodnotou 2). Tieto hodnoty môžeme považovať za podozrivé a taktiež ich môžeme vidieť v grafe rezíduí na obrázku 5, ako aj v grafe závislosti počtu užívateľov a odozvy (obrázok 6). V prvom prípade (index 255) si môžeme všimnúť, že hodnota je okrajová pretože sa vzťahuje na veľmi malý počet užívateľov. Druhý extrémny prípad (index 476) je hodnota, pre ktorú bola odozva nadštandardne vysoká – môže sa jednať o ojedinelý prípad, kedy na základe rôznych iných faktorov bola odozva vysoká. Keďže na lineárnu regresiu môžu mať outliers veľký vlyv, nemali by sme tieto hodnoty brať do úvahy (v ďalšej diskusii sú tieto hodnoty odstránené z dataframu).

Index	Stand. residuals
62	-2.036977
82	2.699228
114	2.111260
129	-2.141213
145	-2.292470
178	2.054883
254	2.011917
255	5.945469
310	-2.111115
332	2.124928
428	2.048785
430	-2.080739
476	8.830417
490	-2.230330

Tabuľka 4: Štandardizované rezíduá a indexy



Obr. 5: Graf rezíduí s označenými outliers na indexoch 255 a 476

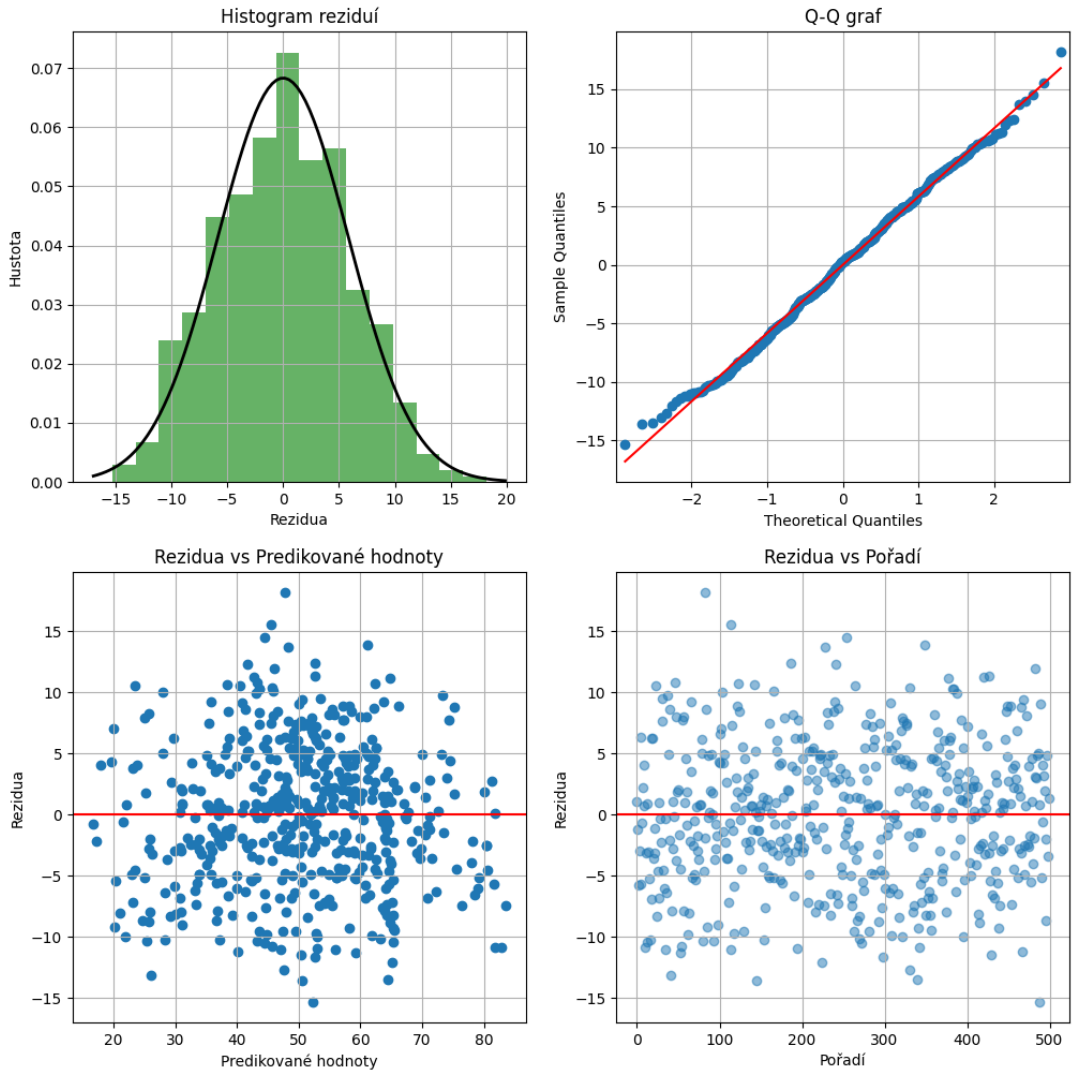


Obr. 6: Graf závislosti počtu užívateľov a odozvy

1b) Diskutovanie splnenia predpokladov lineárnej regresie

Ďalej budeme uvažovať splnenie predpokladov lineárnej regresie na základe regresných diagnostík:

1. matica plánu má hodnosť $m < n$ – platí, hodnosť matice plánu (`np.linalg.matrix_rank(model.exog)`) je 10, počet prediktorov je taktiež 10 a počet meraní je 500
2. stredná hodnota chyby $E(\epsilon) = 0$ - vizuálne môžeme overiť na histograme reziduí (obrázok 7) kde vidíme, že jeho tvar je podobný normálnemu rozdeleniu so stredom v hodnote 0, zároveň vidíme na grafe rezíduí voči predikovaným hodnotám symetrickosť okolo osy x
3. homoskedasticita $D(\epsilon) = \sigma^2$ – môžeme si vizuálne overiť v grafe reziduí vs predikovaných hodnôt (obrázok 7), kedy vidíme približne rovnaké (a náhodné) rozloženie okolo osy x
4. nezávislosť $C(\epsilon_i, \epsilon_j) = 0$ – overíme pomocou Durbin-Watson koeficientu, ktorý vychádza 1.913 (je približne 2, teda môžeme predpokladať nezávislosť)
5. normálne rozdelenie náhodných veličín Y_i – vidíme zhodu s teoretickým normálnym rozdelením v Q-Q grafe a zároveň silné črty normálneho rozdelenia v histograme reziduí



Obr. 7: Základné regresné diagnostiky

2. Identifikácia nastavenia parametrov modelu a problémových predikcií
Na základe hodnôt koeficientov môžeme usúdiť závery o vplyve skúmaných vlastností na odozvu. Najvyššie hodnoty koeficientov sú pri prediktoroch interakcie užívateľov a percenta scrollujúcich užívateľov a pri interakcii užívateľov a operačného systému MacOS. Vhodné nastavenie hodnôt týchto parametrov nám teda ukáže maximálnu hodnotu predikcie odozvy (podobnú hodnotu dostávame pre maximálnu odozvu zo skúmaných dát):

```
# theoretical maximum ping
theory = pd.DataFrame({"os": ["MacOS"], "users": [1], "scroll": [0]})
pred = results.get_prediction(theory)
pred.summary_frame(alpha=0.05) # prediction mean = 84.09

# maximum prediction from observed data
obs = results.predict().argmax()
results.predict()[obs]         # highest prediction = 83.44
```

Pozoruhodný je fakt, že čím máme väčší počet užívateľov, tým menší vplyv má percento scrollujúcich užívateľov – ak nastavíme `users = 0.5` a rôzne hodnoty `scroll` tak dostávame väčšie odchylky odozvy, než pri `users = 1` (vyplýva to z toho, že pri veľkom počte užívateľov je prediktor interakcie užívateľov a scrollovania `users:scroll` "protiváhou" prediktoru `scroll` – viď. tabuľka 3).

3. Odhadnutie hodnoty odozvy s konkrétnymi parametrami

```
input_val = pd.DataFrame({
    "os": ["Windows"], "users": [df["users"].mean()], "scroll": [df["scroll"].mean()]})
pred = results.get_prediction(input_val)
pred.summary_frame(alpha=0.05)
```

odhadnutá odozva: 51.8
konfidenčný interval: $\langle 50.8, 52.8 \rangle$
predikčný interval: $\langle 40.2, 63.4 \rangle$

4. Vhodnosť modelu
V predchádzajúcich častiach sme diskutovali spôsob vytvorenia modelu, elimináciu odľahlých hodnôt a splnenie základných regresných predpokladov pre náš model. Skúmali sme, ako sa daný model chová za rôznych okolností. Jednu z metrík, ktorú sme doposiaľ neuvažovali je koeficient determinácie. Pri našom finálnom modele dosahuje hodnôt 0.842 – čo môžeme interpretovať tak, že náš model je schopný rozpoznať a vhodne odvodiť hodnotu odozvy na cca 84%. Celkový F-test ukazuje, že model skutočne obsahuje prediktory, ktoré vplývajú na hodnotu odozvy. Výsledok modelu po spätnej eliminácii sme ďalej zjednodušovali, aby sme znížili mieru multikolinerity prediktorov. Výsledný model teda spĺňa základné predpoklady, aby sme ho mohli použiť na predikciu odozvy pre rôzne nastavenia parametrov.

Literatúra

[1] Hyvönen, V.; Tolonen, T.: Marec 2019, (získané 30.11.2023).
URL https://vioshyvo.github.io/Bayesian_inference/index.html