MSP – druhý projekt

Autor: Matěj Konopík, Prosinec 2023

Důležité: zdrojové kódy python notebooků pro výrazně lepší čtení jsou zde: https://github.com/terrorgarten/MSP---druh-projekt

Část první - Bayesovské odhady

Nejdříve načteme knihovny a data

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import poisson, gamma, truncnorm

data_file_path = '../data/data_1.csv'

data = pd.read_csv(data_file_path)

observations = data['uloha_1 a)']
observations = observations[~np.isnan(observations)].astype(int)
```

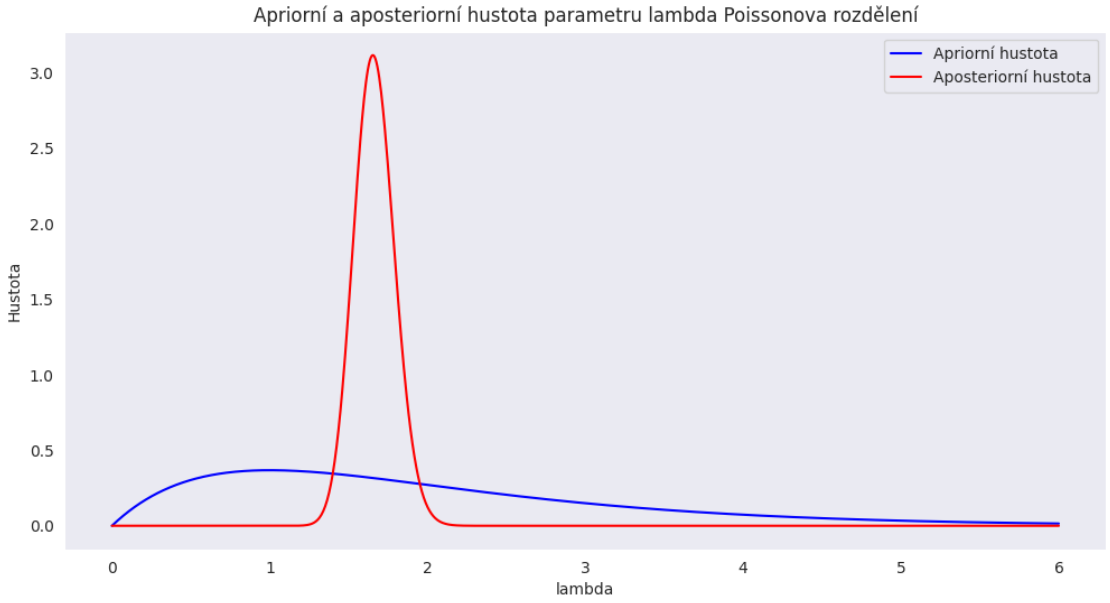
Podčást první

Bayesovská analýza parametru lambdaPoissonova rozdělení

> Do jednoho obrázku vykreslíte apriorní a aposteriorní hustotou parametru Poissonova rozdělení lambda.

Apriorní a aposteriorní hustoty lamba jsou vykresleny pro srovnání dopadu pozorovaných dat na odhad parametru. Hodnoty od "experta" byly poděleny.

```
alpha_prior = 2
beta_prior = 1

alpha_posterior = alpha_prior + observations.sum()
beta_posterior = beta_prior + len(observations)

lambda_values = np.linspace(0, 6, 1000)

apriori_densities = gamma.pdf(lambda_values, a=alpha_prior,
scale=1/beta_prior)
aposteriori_densities = gamma.pdf(lambda_values, a=alpha_posterior,
scale=1/beta_posterior)

plt.figure(figsize=(12, 6))
plt.plot(lambda_values, apriori_densities, label='Apriorní hustota',
color='blue')
plt.plot(lambda_values, aposteriori_densities, label='Aposteriorní
hustota', color='red')
plt.title('Apriorní a aposteriorní hustota parametru lambda Poissonova
rozdělení')
plt.xlabel('lambda')
plt.ylabel('Hustota')
plt.legend()
plt.grid()
plt.show()
```



Prediktivní Hustoty Pozorování Poissonova Rozdělení

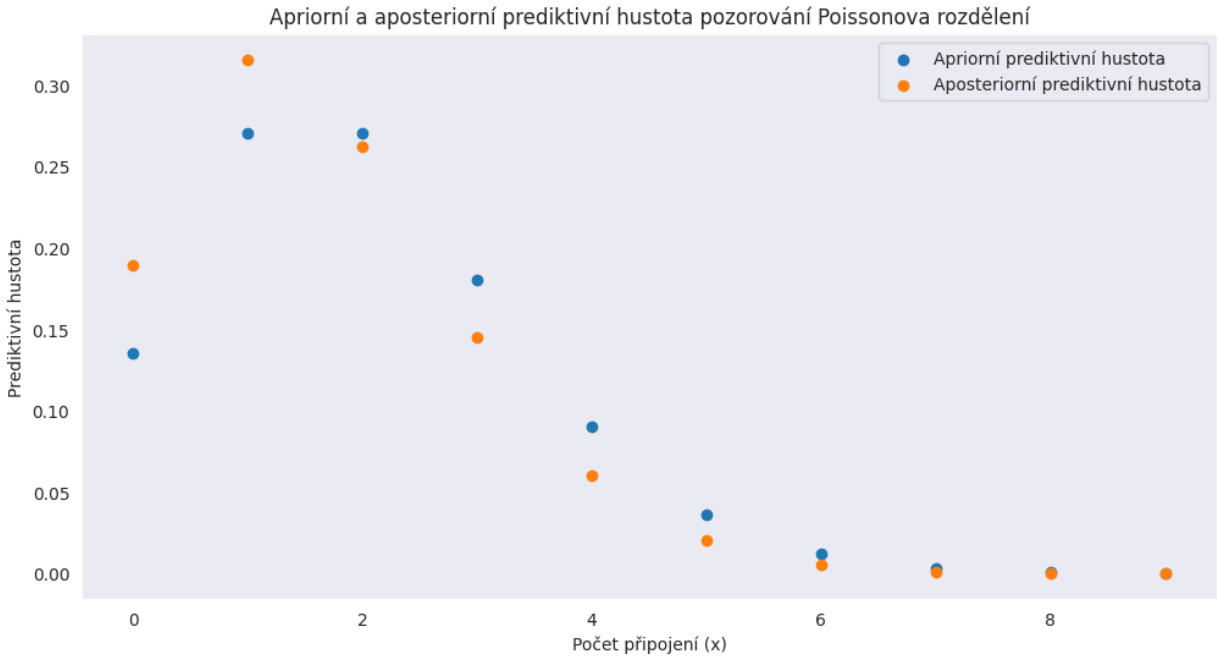
> Do jednoho obrázku vykreslíte apriorní a aposteriorní prediktivní hustotou pozorovaní x za jeden časový interval.

Apriorní prediktivní hustota je založena na průměru apriorního Gamma rozdělení.

Aposteriorní prediktivní hustota využívá průměr aposteriorního Gamma rozdělení.

Obě hustoty jsou vypočítány pomocí Poissonova rozdělení a jsou vykresleny pro srovnání.

```
lambda_apriori_mean = alpha_prior / beta_prior
lambda_aposteriori_mean = alpha_posterior / beta_posterior

x_values = np.arange(0, 10)

apriori_predictive_density = poisson.pmf(x_values,
lambda_apriori_mean)
aposteriori_predictive_density = poisson.pmf(x_values,
lambda_aposteriori_mean)

plt.figure(figsize=(12, 6))
plt.scatter(x_values, apriori_predictive_density, label='Apriorní
prediktivní hustota')
plt.scatter(x_values, aposteriori_predictive_density,
label='Aposteriorní prediktivní hustota')
plt.title('Apriorní a aposteriorní prediktivní hustota pozorování
Poissonova rozdělení')
plt.xlabel('Počet připojení (x)')
plt.ylabel('Prediktivní hustota')
plt.legend()
plt.grid()
plt.show()
```



95% Interval Spolehlivosti pro Parametr lambda

> Sestrojte 95% interval spolehlivosti pro parametr lambda z apriorního a aposteriorního rozdělení a porovnejte je.

Interval spolehlivosti pro Gamma rozdělení se vypočítá pomocí jeho kumulativní distribuční funkce (CDF).

Pro obě rozdělení nalezneme hodnoty lambda, kde CDF Gamma rozdělení dosahuje hodnot 0.025 a 0.975.

```
lower_bound_apriori = gamma.ppf(0.025, a=alpha_prior,
scale=1/beta_prior)
upper_bound_apriori = gamma.ppf(0.975, a=alpha_prior,
scale=1/beta_prior)

lower_bound_aposteriori = gamma.ppf(0.025, a=alpha_posterior,
scale=1/beta_posterior)
upper_bound_aposteriori = gamma.ppf(0.975, a=alpha_posterior,
scale=1/beta_posterior)
```

```
print(f"95% Interval spolehlivosti pro apriorní rozdělení:
({lower_bound_apriori}, {upper_bound_apriori})")
print(f"95% Interval spolehlivosti pro aposteriorní rozdědlení:
({lower_bound_aposteriori}, {upper_bound_aposteriori})")
```

```
95% Interval spolehlivosti pro apriorní rozdělení:
(0.24220927854396496, 5.571643390938898)
95% Interval spolehlivosti pro aposteriorní rozdědlení:
(1.4213461513304455, 1.9241339817546559)
```

Bodové Odhady Parametru lambda

> Vyberte si dva aposteriorní bodové odhady parametru lambda, porovnejte je a okomentujte jejich výběr.

Průměr Gamma rozdělení je alfa/beta. Pro aposteriorní rozdělení je to alfa_posterior/beta_posterior.

Medián Gamma rozdělení získáme pomocí kumulativní distribuční funkce (CDF), jako hodnotu lambda, kde CDF dosahuje 0.5.

Průměr odráží střední tendenci, zatímco medián je odolnější vůči extrémním hodnotám.

```
mean_aposteriori = alpha_posterior / beta_posterior

median_aposteriori = gamma.ppf(0.5, a=alpha_posterior,
scale=1/beta_posterior)

print("Aposteriorní střední hodnota: {}".format(mean_aposteriori))
print("Aposteriorní medián: {}".format(median_aposteriori))
```

```
Aposteriorní střední hodnota: 1.6633663366336633
Aposteriorní medián: 1.6600671732693628
```

Bodové Odhady Počtu Pozorování

> Vyberte si jeden apriorní a jeden aposteriorní bodový odhad počtu pozorovaní a porovnejte je.

Odhady odrážejí očekávaný počet připojení za 1 ms před (průměr Poissonova rozdělení pro apriorní odhad) a po (průměr Poissonova rozdělení pro aposteriorní odhad) pozorování dat. Rozdíl mezi odhady poskytuje přehled o dopadu pozorovaných dat na naše očekávání.

```
apriori_point_estimate = alpha_prior / beta_prior

aposteriori_point_estimate = alpha_posterior / beta_posterior

print(f"Apriorní bodový odhad: {apriori_point_estimate}")
print(f"Aposteriorní bodový odhad: {aposteriori_point_estimate}")
```

```
Apriorní bodový odhad: 2.0
Aposteriorní bodový odhad: 1.6633663366336633
```

Podčást druhá

Aproximace diskrétním rozdělením

Nejdříve načteme data do dataframe.

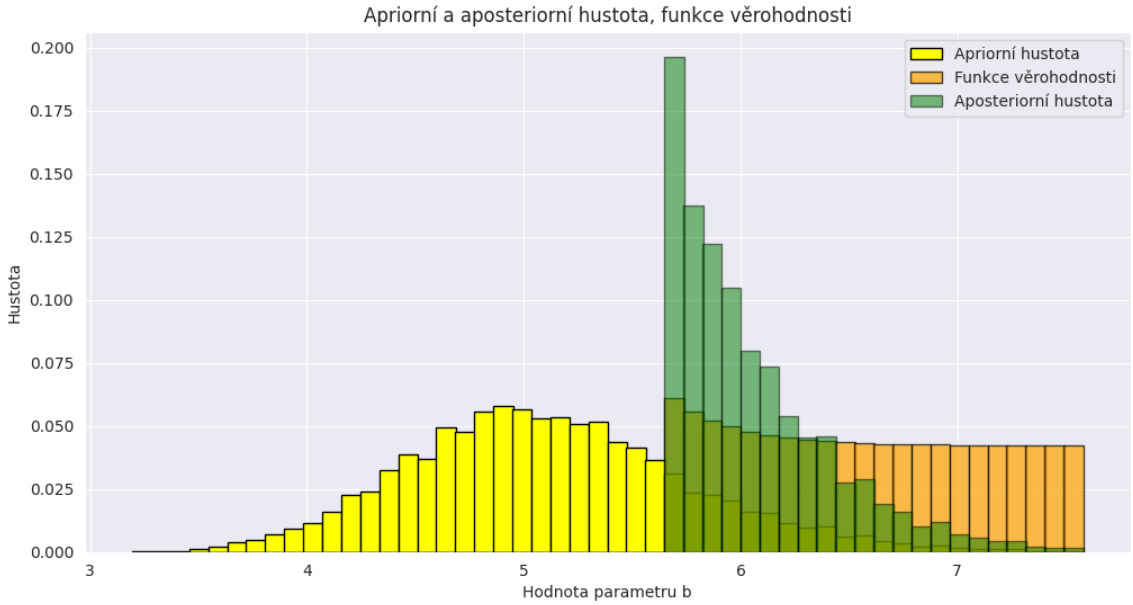
```
file_path = '../data/Projekt-2_Data.xlsx'  # Update this path
data = pd.read_excel(file_path, sheet_name='Úloha 1')
```

Vizualizace Apriorní, Aposteriorní Distribuce a Funkce Věrohodnosti

> Do jednoho grafu vykreslíte apriorní, aposteriorní hustotou a funkci věrohodnosti. Funkci věrohodnosti normujte tak, aby jej součet byl 1 kvůli porovnatelnosti v obrázku.

Zde použijeme pro diskretizaci 50 binů. Výsledkem je graf, který zobrazuje apriorní a aposteriorní hustotu parametru b a funkci věrohodnosti.

```
observations = data['uloha_1 b)_pozorování'].dropna()
grouped_max_values = data.groupby('skupina')['uloha_1 b)_prior'].max()

bin_edges = np.linspace(grouped_max_values.min(),
grouped_max_values.max(), 50 + 1)
bin_centers = (bin_edges[:-1] + bin_edges[1:]) / 2
binned_counts, _ = np.histogram(grouped_max_values, bins=bin_edges)
binned_pmf = binned_counts / binned_counts.sum()

mu = 3
sigma = 1
a = 1
param_index = np.inf
sum = 0
posts = []
likelihoods = []

for param_index, param_probability in zip(bin_centers, binned_pmf):
    norm = truncnorm(a=(a - mu) / sigma, b=(param_index - mu) / sigma,
loc=mu, scale=sigma)
    likelihood = np.prod(norm.pdf(observations))
    likelihoods.append(likelihood)
    posts.append(likelihood * param_probability)
    sum += likelihood * param_probability

normalized_likelihoods = np.array(likelihoods) / np.sum(likelihoods)
posterior = np.array(posts) / sum
normalized_posterior = posterior / np.sum(posterior)

plt.figure(figsize=(12, 6))
plt.bar(bin_centers, binned_pmf, width=0.09, color='yellow',
label='Apriorní hustota', edgecolor='black')
plt.bar(bin_centers, normalized_likelihoods, width=0.09,
color='orange', alpha=0.7, label='Funkce věrohodnosti',
edgecolor='black')
plt.bar(bin_centers, normalized_posterior, width=0.09, color='green',
alpha=0.5, label='Aposteriorní hustota', edgecolor='black')
plt.xlabel('Hodnota parametru b')
plt.ylabel('Hustota')
plt.title('Apriorní a aposteriorní hustota, funkce věrohodnosti')
plt.legend()
plt.show()
```



95% Interval Spolehlivosti pro Parametr b

Z aposteriorní hustoty určete 95% interval spolehlivosti (konfidenční interval) pro parametr b.

```
posterior_cdf = np.cumsum(normalized_posterior)

lower_bound_idx = np.where(posterior_cdf >= 0.025)[0][0]
upper_bound_idx = np.where(posterior_cdf >= 0.975)[0][0]

CI_lower = bin_centers[lower_bound_idx]
CI_upper = bin_centers[upper_bound_idx]

print(f"95% interval spolehlivosti pro parametr b: ({CI_lower};
{CI_upper})")
```

```
95% interval spolehlivosti pro parametr b: (5.693712028182375;
7.008910628347767)
```

Bodové Odhady Parametru b

Vyberte dva bodové odhady parametru b a spočítejte je.

```
posterior_mean = np.sum(bin_centers * normalized_posterior)

mode_index = np.argmax(normalized_posterior)
posterior_mode = bin_centers[mode_index]

print(f"Střední hodnota pro b: {posterior_mean}")
print(f"Modus pro b: {posterior_mode}")
```

```
Střední hodnota pro b: 6.052771319832352
Modus pro b: 5.693712028182375
```

MSP projekt - část druhá - regrese

Nejdříve načteme data a pro kategorický atribut OSType provedeme one-hot encoding. Zároven převedeme výsledné dummy sloupce na celočíselný typ (0 nebo 1). Také použijeme drop_first, abysme se zbavili jednoho dummy sloupce, který je redundantní. Zde odstraníme OSType_Android.

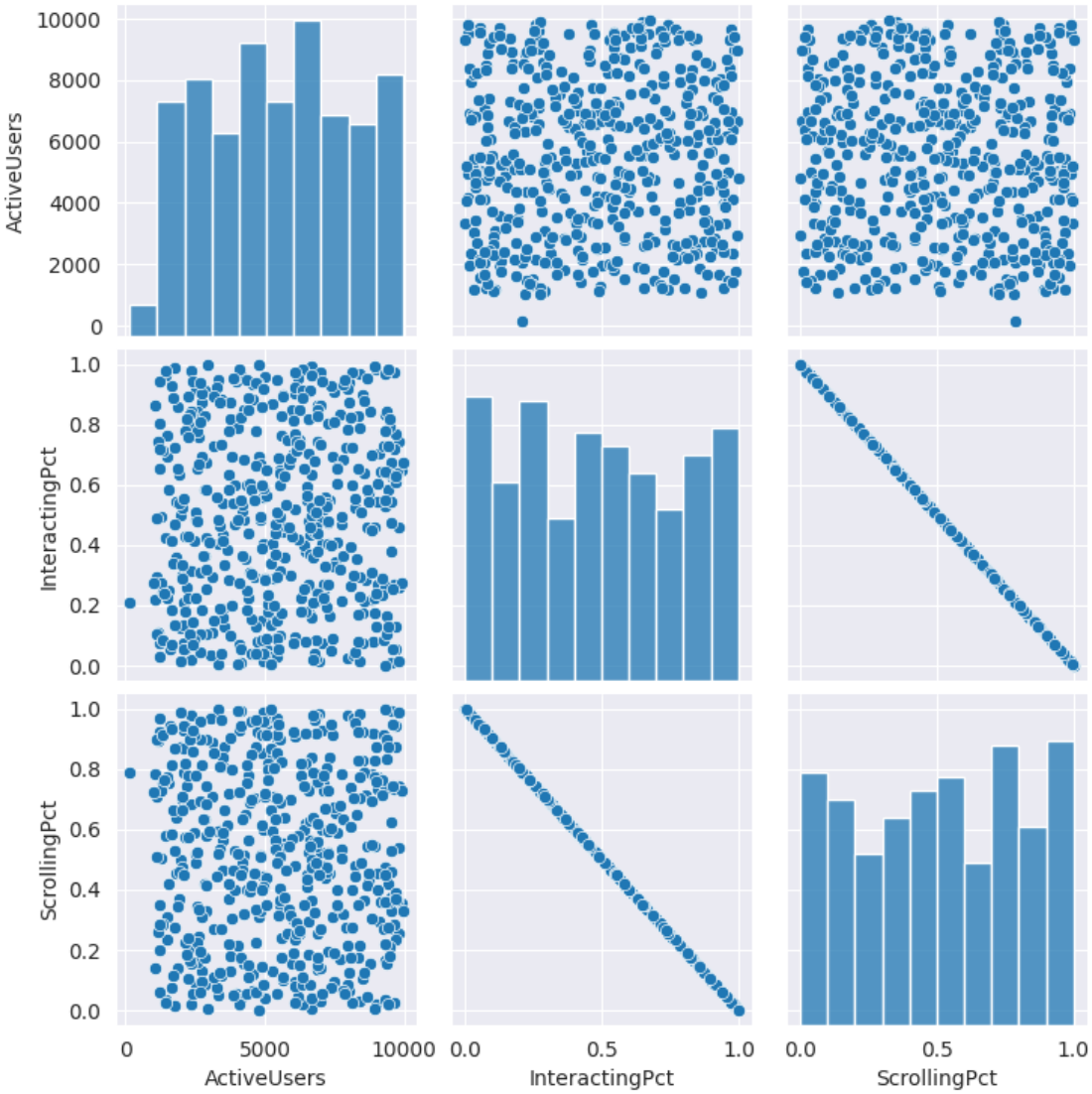
```
import pandas as pd
import statsmodels.api as sm
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.preprocessing import MinMaxScaler
from statsmodels.stats.outliers_influence import
variance_inflation_factor

data = pd.read_csv('../data/data_2.csv')

data_encoded = pd.get_dummies(data, columns=['OSType'],
drop_first=True)

dummy_columns = data_encoded.filter(like='OSType_').columns
data_encoded[dummy_columns] = data_encoded[dummy_columns].astype(int)

print(data_encoded.head())
```

```
   ActiveUsers  InteractingPct  ScrollingPct  Ping [ms]  OSType_MacOS  \
0         4113          0.8283        0.1717         47             0
1         7549          0.3461        0.6539         46             0
2         8855          0.2178        0.7822         55             0
3         8870          0.0794        0.9206         56             0
4         9559          0.7282        0.2718         76             1

   OSType_Windows  OSType_iOS
0               0           1
1               0           1
2               1           0
3               0           0
4               0           0
```

Kolinearita

Dále si vytvoříme korelační matici a následně si ji zobrazíme. Zde použijeme pouze prediktory - ActiveUsers, InteractingPct a ScrollingPct.

```
sns.pairplot(data[['ActiveUsers', 'InteractingPct', 'ScrollingPct']])
plt.show()
correlation_matrix = data[['ActiveUsers', 'InteractingPct',
'ScrollingPct']].corr()
```



V datech byla pozorována dokonalá negativní korelace (-1) mezi "InteractingPct" a "ScrollingPct" - jedná se o atributy, které jsou si doplňky. Tato situace naznačuje lineární závislost těchto dvou proměnných, což v regresní analýze může vést k problémům s multicolinearitou. Proto bylo rozhodnuto odstranit "ScrollingPct" z modelu, čímž se snižuje redundance a zlepšuje odhad regresních koeficientů.

```
data_encoded.drop('ScrollingPct', axis=1, inplace=True)
print(data_encoded.head())
```

```
   ActiveUsers  InteractingPct  Ping [ms]  OSType_MacOS  OSType_Windows  \
0         4113          0.8283         47             0               0
1         7549          0.3461         46             0               0
2         8855          0.2178         55             0               1
3         8870          0.0794         56             0               0
4         9559          0.7282         76             1               0

   OSType_iOS
0           1
1           1
2           0
3           0
4           0
```

Vytvoření plného kvadratického modelu

```
data_encoded['ActiveUsers^2'] = data_encoded['ActiveUsers'] ** 2
data_encoded['InteractingPct^2'] = data_encoded['InteractingPct'] ** 2

data_encoded['ActiveUsers_x_InteractingPct'] = data_encoded['ActiveUsers']
* data_encoded['InteractingPct']

data_encoded['ActiveUsers_x_MacOS'] = data_encoded['ActiveUsers'] *
data_encoded['OSType_MacOS']
data_encoded['ActiveUsers_x_Windows'] = data_encoded['ActiveUsers'] *
data_encoded['OSType_Windows']
data_encoded['ActiveUsers_x_iOS'] = data_encoded['ActiveUsers'] *
data_encoded['OSType_iOS']

data_encoded['InteractingPct_x_MacOS'] = data_encoded['InteractingPct'] *
data_encoded['OSType_MacOS']
data_encoded['InteractingPct_x_Windows'] = data_encoded['InteractingPct']
* data_encoded['OSType_Windows']
data_encoded['InteractingPct_x_iOS'] = data_encoded['InteractingPct'] *
data_encoded['OSType_iOS']

print(data_encoded.head())
```

```
   ActiveUsers  InteractingPct  Ping [ms]  OSType_MacOS  OSType_Windows  \
0         4113          0.8283         47             0               0
1         7549          0.3461         46             0               0
2         8855          0.2178         55             0               1
3         8870          0.0794         56             0               0
4         9559          0.7282         76             1               0

   OSType_iOS  ActiveUsers^2  InteractingPct^2  ActiveUsers_x_InteractingPct  \
0           1       16916769          0.686081                     3406.7979
1           1       56987401          0.119785                     2612.7089
2           0       78411025          0.047437                     1928.6190
3           0       78676900          0.006304                      704.2780
4           0       91374481          0.530275                     6960.8638

   ActiveUsers_x_MacOS  ActiveUsers_x_Windows  ActiveUsers_x_iOS  \
0                    0                      0               4113
1                    0                      0               7549
2                    0                   8855                  0
3                    0                      0                  0
4                 9559                      0                  0

   InteractingPct_x_MacOS  InteractingPct_x_Windows  InteractingPct_x_iOS
0                  0.0000                    0.0000                0.8283
1                  0.0000                    0.0000                0.3461
2                  0.0000                    0.2178                0.0000
3                  0.0000                    0.0000                0.0000
4                  0.7282                    0.0000                0.0000
```

Fitting modelu se základní sadou prediktorů

```
X = data_encoded.drop('Ping [ms]', axis=1)
y = data_encoded['Ping [ms]']

X_with_constant = sm.add_constant(X)
full_model = sm.OLS(y, X_with_constant).fit()

print(full_model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:              Ping [ms]   R-squared:                       0.844
Model:                            OLS   Adj. R-squared:                  0.839
Method:                 Least Squares   F-statistic:                     187.9
Date:                Sun, 17 Dec 2023   Prob (F-statistic):          5.18e-186
Time:                        22:57:27   Log-Likelihood:                -1598.4
No. Observations:                 502   AIC:                             3227.
Df Residuals:                     487   BIC:                             3290.
Df Model:                          14
Covariance Type:            nonrobust
================================================================================================
                                   coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------------------
const                           -0.3388      2.354     -0.144      0.886      -4.965       4.287
ActiveUsers                      0.0100      0.001     17.571      0.000       0.009       0.011
InteractingPct                  37.6062      4.567      8.234      0.000      28.633      46.580
OSType_MacOS                     2.0017      2.260      0.886      0.376      -2.440       6.443
OSType_Windows                   7.8174      2.217      3.526      0.000       3.461      12.174
OSType_iOS                      -0.0483      2.265     -0.021      0.983      -4.499       4.403
ActiveUsers^2                -4.17e-07    4.4e-08     -9.469      0.000   -5.03e-07    -3.3e-07
InteractingPct^2                -3.7258      3.492     -1.067      0.287     -10.587       3.135
ActiveUsers_x_InteractingPct    -0.0031      0.000     -8.532      0.000      -0.004      -0.002
ActiveUsers_x_MacOS              0.0014      0.000      4.536      0.000       0.001       0.002
ActiveUsers_x_Windows           -0.0008      0.000     -2.505      0.013      -0.001      -0.000
ActiveUsers_x_iOS               -0.0011      0.000     -3.369      0.001      -0.002      -0.000
InteractingPct_x_MacOS          -0.3566      2.530     -0.141      0.888      -5.327       4.614
InteractingPct_x_Windows         0.4260      2.721      0.157      0.876      -4.919       5.771
InteractingPct_x_iOS             0.2678      2.691      0.100      0.921      -5.020       5.556
==============================================================================
Omnibus:                      228.442   Durbin-Watson:                   1.933
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             3152.488
Skew:                           1.603   Prob(JB):                         0.00
Kurtosis:                      14.851   Cond. No.                     1.06e+09
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 1.06e+09. This might indicate that there are
strong multicollinearity or other numerical problems.
```

Normalizace prediktorů

Nyní normalizujeme prediktory, abychom dosáhli rovnocenného přínosu všech prediktorů. Využijeme MinMaxScaler, který převede hodnoty do intervalu `[0, 1]`.

```
predictor_columns = X.columns
scaler = MinMaxScaler()
X_normalized = scaler.fit_transform(X)
X_normalized = pd.DataFrame(X_normalized, columns=predictor_columns)
print(X_normalized.head())
```

```
   ActiveUsers  InteractingPct  OSType_MacOS  OSType_Windows  OSType_iOS  \
0     0.404082        0.829376           0.0             0.0         1.0
1     0.754694        0.346258           0.0             0.0         1.0
2     0.887959        0.217714           0.0             1.0         0.0
3     0.889490        0.079050           0.0             0.0         0.0
4     0.959796        0.729085           1.0             0.0         0.0

   ActiveUsers^2  InteractingPct^2  ActiveUsers_x_InteractingPct  \
0       0.170573          0.688006                      0.362253
1       0.575168          0.120121                      0.277700
2       0.791484          0.047570                      0.204859
3       0.794168          0.006322                      0.074494
4       0.922377          0.531763                      0.740683

   ActiveUsers_x_MacOS  ActiveUsers_x_Windows  ActiveUsers_x_iOS  \
0             0.000000               0.000000           0.430455
1             0.000000               0.000000           0.790058
2             0.000000               0.893001           0.000000
3             0.000000               0.000000           0.000000
4             0.960414               0.000000           0.000000

   InteractingPct_x_MacOS  InteractingPct_x_Windows  InteractingPct_x_iOS
0                0.000000                  0.000000              0.829461
1                0.000000                  0.000000              0.346585
2                0.000000                  0.219027              0.000000
3                0.000000                  0.000000              0.000000
4                0.732154                  0.000000              0.000000
```

Opět si zobrazíme summary modelu

```
X_normalized_with_constant = sm.add_constant(X_normalized)
normalized_model = sm.OLS(y, X_normalized_with_constant).fit()
print(normalized_model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:              Ping [ms]   R-squared:                       0.844
Model:                            OLS   Adj. R-squared:                  0.839
Method:                 Least Squares   F-statistic:                     187.9
Date:                Sun, 17 Dec 2023   Prob (F-statistic):          5.18e-186
Time:                        22:57:27   Log-Likelihood:                -1598.4
No. Observations:                 502   AIC:                             3227.
Df Residuals:                     487   BIC:                             3290.
Df Model:                          14
Covariance Type:            nonrobust
================================================================================================
                                   coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------------------
const                            1.1895      2.288      0.520      0.603      -3.306       5.685
ActiveUsers                     98.2358      5.591     17.571      0.000      87.251     109.221
```

```
InteractingPct                  37.5347      4.558      8.234      0.000      28.578      46.491
OSType_MacOS                     2.0017      2.260      0.886      0.376      -2.440       6.443
OSType_Windows                   7.8174      2.217      3.526      0.000       3.461      12.174
OSType_iOS                      -0.0483      2.265     -0.021      0.983      -4.499       4.403
ActiveUsers^2                  -41.2962      4.361     -9.469      0.000     -49.865     -32.727
InteractingPct^2                -3.7153      3.482     -1.067      0.287     -10.557       3.127
ActiveUsers_x_InteractingPct   -28.9853      3.397     -8.532      0.000     -35.660     -22.310
ActiveUsers_x_MacOS             13.9078      3.066      4.536      0.000       7.884      19.932
ActiveUsers_x_Windows           -7.5547      3.015     -2.505      0.013     -13.479      -1.630
ActiveUsers_x_iOS              -10.1126      3.002     -3.369      0.001     -16.011      -4.214
InteractingPct_x_MacOS          -0.3546      2.516     -0.141      0.888      -5.298       4.589
InteractingPct_x_Windows         0.4237      2.705      0.157      0.876      -4.892       5.739
InteractingPct_x_iOS             0.2675      2.688      0.100      0.921      -5.013       5.548
==============================================================================
Omnibus:                      228.442   Durbin-Watson:                   1.933
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             3152.488
Skew:                           1.603   Prob(JB):                         0.00
Kurtosis:                      14.851   Cond. No.                         43.5
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Zpětná eliminace

Provedem zpětnou eliminaci prediktorů, abychom získali model s lepší prediktivní schopností. Zde použijeme signifikanci 0.05.

```
def backward_elimination(data, target, significance_level = 0.05):
    features = data.columns.tolist()
    while len(features) > 0:
        features_with_constant = sm.add_constant(data[features])
        p_values = sm.OLS(target,
features_with_constant).fit().pvalues[1:]
        max_p_value = p_values.max()
        if max_p_value >= significance_level:
            excluded_feature = p_values.idxmax()
            features.remove(excluded_feature)
        else:
            break
    return features

selected_features = backward_elimination(X_normalized,
data_encoded['Ping [ms]'])

X_selected = X_normalized[selected_features]
```

Opět fitneme model a zobrazíme summary

```
X_selected_with_constant = sm.add_constant(X_selected)
back_elim_model = sm.OLS(y, X_selected_with_constant).fit()
print(back_elim_model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:              Ping [ms]   R-squared:                       0.843
Model:                            OLS   Adj. R-squared:                  0.840
Method:                 Least Squares   F-statistic:                     330.9
Date:                Sun, 17 Dec 2023   Prob (F-statistic):          9.30e-193
Time:                        22:57:27   Log-Likelihood:                -1599.7
No. Observations:                 502   AIC:                             3217.
Df Residuals:                     493   BIC:                             3255.
Df Model:                           8
Covariance Type:            nonrobust
================================================================================================
                                   coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------------------
const                            2.3036      1.510      1.526      0.128      -0.663       5.270
ActiveUsers                     97.2593      5.074     19.167      0.000      87.289     107.230
InteractingPct                  34.1917      2.103     16.262      0.000      30.061      38.323
OSType_Windows                   7.3575      1.405      5.236      0.000       4.597      10.118
ActiveUsers^2                  -41.0612      4.317     -9.511      0.000     -49.543     -32.579
ActiveUsers_x_InteractingPct   -29.4115      3.330     -8.832      0.000     -35.955     -22.868
ActiveUsers_x_MacOS             16.5469      1.216     13.603      0.000      14.157      18.937
ActiveUsers_x_Windows           -6.4913      2.395     -2.710      0.007     -11.197      -1.786
ActiveUsers_x_iOS              -10.0448      1.237     -8.118      0.000     -12.476      -7.614
==============================================================================
Omnibus:                      242.580   Durbin-Watson:                   1.931
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             3742.643
Skew:                           1.701   Prob(JB):                         0.00
Kurtosis:                      15.937   Cond. No.                         35.5
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

VIF eliminace

Provedem analýtu VIF pomocí funkce `variance_inflation_factor` z balíčku `statsmodels.stats.outliers_influence`. Pokud bude hodnota VIF některého predikotru nad 10, tak odstraníme daný prediktor s nejvyšší hodnotou VIF.

```
vif_data = pd.DataFrame()
vif_data['feature'] = X_selected_with_constant.columns
vif_data['VIF'] =
[variance_inflation_factor(X_selected_with_constant.values, i) for i
in range(X_selected_with_constant.shape[1])]

vif_data
```

```
                       feature        VIF
0                        const  32.761712
1                   ActiveUsers  24.981312
2                InteractingPct   5.576012
3                OSType_Windows   5.551936
4                 ActiveUsers^2  22.204978
5  ActiveUsers_x_InteractingPct   8.566008
6           ActiveUsers_x_MacOS   1.657555
7         ActiveUsers_x_Windows   6.474746
8             ActiveUsers_x_iOS   1.550683
```

Vidíme, že ActiveUsers má hodnotu nejvyšší hodnotu VIF 24.29. Je jasné, že ActiveUsers a jeho kvadrát jsou silně korelované, jelikož se vlastně jedná jen o hodnotu umocněnou na druhou. Odstraníme ale ActiveUsers^2, jelikož se jedná o dobrou praxi.

```
X_selected_reduced = X_selected.drop('ActiveUsers^2', axis=1)

X_selected_reduced_with_constant = sm.add_constant(X_selected_reduced)
vif_data_cleaned = pd.DataFrame()
vif_data_cleaned['feature'] = X_selected_reduced_with_constant.columns
vif_data_cleaned['VIF'] =
[variance_inflation_factor(X_selected_reduced_with_constant.values, i)
for i in range(X_selected_reduced_with_constant.shape[1])]

vif_data_cleaned
```

```
                       feature        VIF
0                        const  21.405088
1                   ActiveUsers   4.567019
2                InteractingPct   5.561726
3                OSType_Windows   5.546736
4  ActiveUsers_x_InteractingPct   8.550677
5           ActiveUsers_x_MacOS   1.656643
6         ActiveUsers_x_Windows   6.464752
7             ActiveUsers_x_iOS   1.550671
```

Je vidět, že se všechny hodnoty VIF snížily pod 10, což by již mohlo být v pořádku.

Opět fitneme model a zobrazíme summary

```
X_selected_vif_reduced_with_constant =
sm.add_constant(X_selected_reduced)
vif_elim_model = sm.OLS(y, X_selected_vif_reduced_with_constant).fit()
print(vif_elim_model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:              Ping [ms]   R-squared:                       0.814
Model:                            OLS   Adj. R-squared:                  0.812
Method:                 Least Squares   F-statistic:                     309.3
Date:                Sun, 17 Dec 2023   Prob (F-statistic):          4.99e-176
Time:                        22:57:27   Log-Likelihood:                -1642.0
No. Observations:                 502   AIC:                             3300.
Df Residuals:                     494   BIC:                             3334.
Df Model:                           7
Covariance Type:            nonrobust
================================================================================================
                                   coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------------------
const                           10.7593      1.326      8.111      0.000       8.153      13.365
ActiveUsers                     53.6288      2.358     22.743      0.000      48.996      58.262
InteractingPct                  35.2040      2.282     15.426      0.000      30.720      39.688
OSType_Windows                   7.7665      1.526      5.088      0.000       4.767      10.766
ActiveUsers_x_InteractingPct   -30.7515      3.616     -8.504      0.000     -37.856     -23.647
ActiveUsers_x_MacOS             16.8184      1.322     12.726      0.000      14.222      19.415
ActiveUsers_x_Windows           -7.3863      2.601     -2.840      0.005     -12.496      -2.276
ActiveUsers_x_iOS              -10.0117      1.345     -7.445      0.000     -12.654      -7.370
==============================================================================
Omnibus:                      123.688   Durbin-Watson:                   1.872
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              663.242
Skew:                           0.959   Prob(JB):                    9.53e-145
Kurtosis:                       8.294   Cond. No.                         22.3
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Denormalizace

Upravíme scaler a vrátíme hodnoty do původního rozsahu.

```
selected_predictor_columns = X_selected_reduced.columns
selected_indices = [list(predictor_columns).index(col) for col in
selected_predictor_columns]

denorm_scaler = MinMaxScaler()
denorm_scaler.min_ = scaler.min_[selected_indices]
denorm_scaler.scale_ = scaler.scale_[selected_indices]

X_denormalized = denorm_scaler.inverse_transform(X_selected_reduced)
X_denormalized = pd.DataFrame(X_denormalized,
columns=selected_predictor_columns)

print(X_denormalized.head())
```

```
   ActiveUsers  InteractingPct  OSType_Windows  ActiveUsers_x_InteractingPct  \
0       4113.0          0.8283             0.0                     3406.7979
1       7549.0          0.3461             0.0                     2612.7089
2       8855.0          0.2178             1.0                     1928.6190
3       8870.0          0.0794             0.0                      704.2780
4       9559.0          0.7282             0.0                     6960.8638

   ActiveUsers_x_MacOS  ActiveUsers_x_Windows  ActiveUsers_x_iOS
0                  0.0                    0.0             4113.0
1                  0.0                    0.0             7549.0
2                  0.0                 8855.0                0.0
3                  0.0                    0.0                0.0
4               9559.0                    0.0                0.0
```

Opět fitneme model a zobrazíme summary

```
X_denormalized_with_constant = sm.add_constant(X_denormalized)
denorm_model = sm.OLS(y, X_denormalized_with_constant).fit()
print(denorm_model.summary())
```

```
                                OLS Regression Results
==============================================================================
Dep. Variable:              Ping [ms]   R-squared:                       0.814
Model:                            OLS   Adj. R-squared:                  0.812
Method:                 Least Squares   F-statistic:                     309.3
Date:                Sun, 17 Dec 2023   Prob (F-statistic):          4.99e-176
Time:                        22:57:27   Log-Likelihood:                -1642.0
No. Observations:                 502   AIC:                             3300.
Df Residuals:                     494   BIC:                             3334.
Df Model:                           7
Covariance Type:            nonrobust
================================================================================================
                                   coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------------------
const                            9.9196      1.357      7.309      0.000       7.253      12.586
ActiveUsers                      0.0055      0.000     22.743      0.000       0.005       0.006
InteractingPct                  35.2710      2.286     15.426      0.000      30.779      39.763
OSType_Windows                   7.7665      1.526      5.088      0.000       4.767      10.766
ActiveUsers_x_InteractingPct    -0.0033      0.000     -8.504      0.000      -0.004      -0.003
ActiveUsers_x_MacOS              0.0017      0.000     12.726      0.000       0.001       0.002
ActiveUsers_x_Windows           -0.0007      0.000     -2.840      0.005      -0.001      -0.000
ActiveUsers_x_iOS               -0.0010      0.000     -7.445      0.000      -0.001      -0.001
==============================================================================
Omnibus:                      123.688   Durbin-Watson:                   1.872
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              663.242
Skew:                           0.959   Prob(JB):                    9.53e-145
Kurtosis:                       8.294   Cond. No.                     6.56e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 6.56e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

Detekce a odstranění odlehlých pozorování

Z grafu reziduí níže je vidět, že existují dva odlehlé body. Ty ale můžeme dopředu zjistit z reziduí a odstranit je. Zde použijeme 3 * standardní odchylku reziduí jako hranici pro odlehlá pozorování..

```
residuals = denorm_model.resid
residuals_std = residuals.std()

outliers = np.abs(residuals) > 3 * residuals_std

outlier_indices = outliers[outliers].index
outlier_indices
```

```
Index([255, 476], dtype='int64')
```

Po odstranění odlehlých pozorování, majících vliv na model, ze získaných indexů, fitneme model a zobrazíme summary pro finální podobu.

```
X_cleaned = X_denormalized.drop(outlier_indices)
y_cleaned = y.drop(outlier_indices)

X_cleaned_with_constant = sm.add_constant(X_cleaned)
final_model_cleaned = sm.OLS(y_cleaned, X_cleaned_with_constant).fit()

print(final_model_cleaned.summary())
print("\n\n", X_cleaned.head(3))
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:              Ping [ms]   R-squared:                       0.842
Model:                            OLS   Adj. R-squared:                  0.840
Method:                 Least Squares   F-statistic:                     374.2
Date:                Sun, 17 Dec 2023   Prob (F-statistic):          1.71e-192
Time:                        22:57:27   Log-Likelihood:                -1592.3
No. Observations:                 500   AIC:                             3201.
Df Residuals:                     492   BIC:                             3234.
Df Model:                           7
Covariance Type:            nonrobust
================================================================================================
                                   coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------------------
const                            8.5942      1.259      6.827      0.000       6.121      11.067
ActiveUsers                      0.0057      0.000     25.489      0.000       0.005       0.006
InteractingPct                  36.6538      2.107     17.398      0.000      32.514      40.793
OSType_Windows                   8.1826      1.404      5.827      0.000       5.424      10.941
ActiveUsers_x_InteractingPct    -0.0035      0.000     -9.825      0.000      -0.004      -0.003
ActiveUsers_x_MacOS              0.0017      0.000     13.874      0.000       0.001       0.002
ActiveUsers_x_Windows           -0.0008      0.000     -3.513      0.000      -0.001      -0.000
ActiveUsers_x_iOS               -0.0010      0.000     -8.087      0.000      -0.001      -0.001
==============================================================================
Omnibus:                        4.707   Durbin-Watson:                   1.914
Prob(Omnibus):                  0.095   Jarque-Bera (JB):                3.366
Skew:                           0.020   Prob(JB):                        0.186
Kurtosis:                       2.600   Cond. No.                     6.60e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 6.6e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

	ActiveUsers	InteractingPct	OSType_Windows	ActiveUsers_x_InteractingPct	ActiveUsers_x_MacOS	ActiveUsers_x_Windows	ActiveUsers_x_iOS
0	4113.0	0.8283	0.0	3406.7979	0.0	0.0	4113.0
1	7549.0	0.3461	0.0	2612.7089	0.0	0.0	7549.0
2	8855.0	0.2178	1.0	1928.6190	0.0	8855.0	0.0

Kontrola VIF u poslední verze dat

Všechny hodnoty VIF jsou nyní pod 10, což znamená, že neexistuje příliš silná kolinearita mezi prediktory. Sloupec const, respektive konstantní složka modelu má VIF vyšší, ale toto by nemělo být u regresních modelů neobvyklé.

```
vif_data_cleaned = pd.DataFrame()
vif_data_cleaned['feature'] = X_cleaned_with_constant.columns
vif_data_cleaned['VIF'] =
[variance_inflation_factor(X_cleaned_with_constant.values, i) for i in
range(X_cleaned_with_constant.shape[1])]

vif_data_cleaned
```

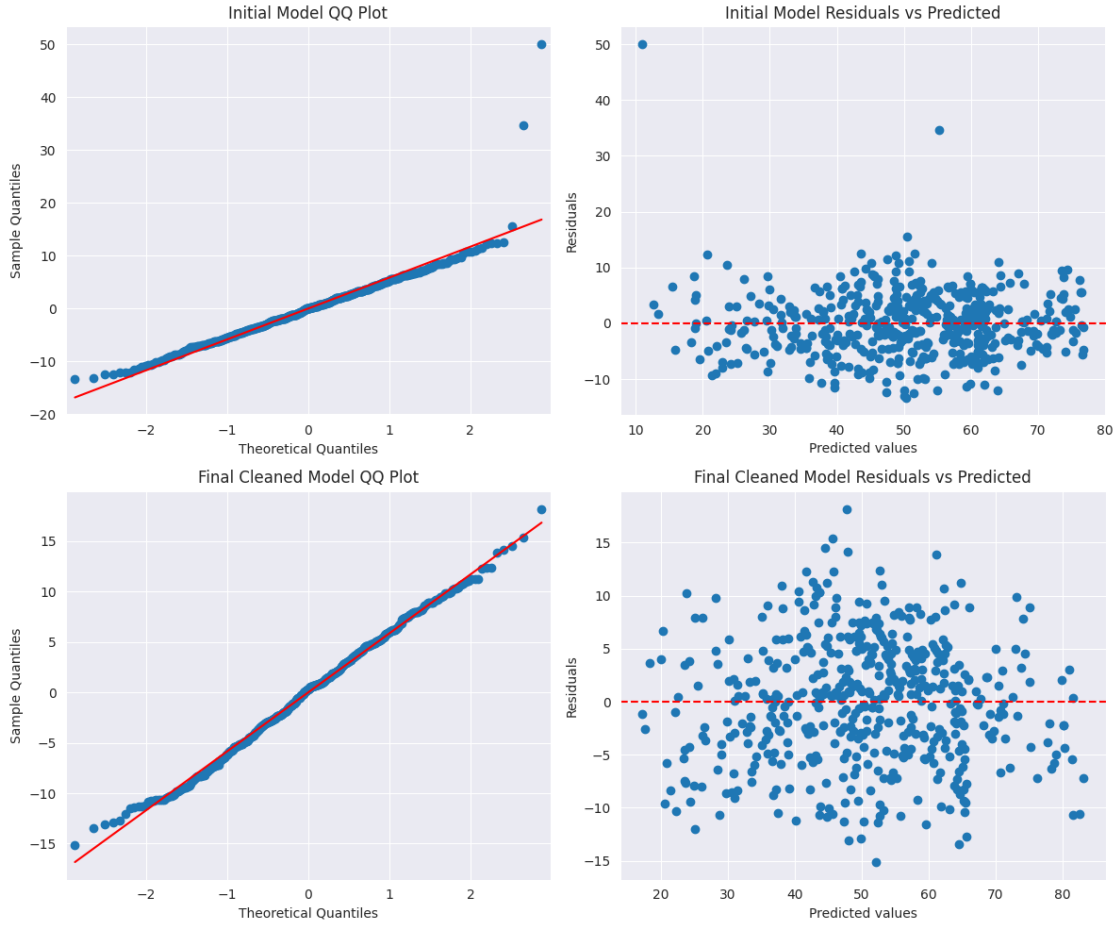
	feature	VIF
0	const	22.817977
1	ActiveUsers	4.598525
2	InteractingPct	5.601527
3	OSType_Windows	5.543347
4	ActiveUsers_x_InteractingPct	8.586170
5	ActiveUsers_x_MacOS	1.654880
6	ActiveUsers_x_Windows	6.455178
7	ActiveUsers_x_iOS	1.549256

Q-Q a resiudální grafy

Nyní se můžeme podívat na Q-Q grafy a grafy reziduí. Zde je vidět, že velmi pravděpodobně z důvodu odstranění odlehlých pozorování se grafy výrazně zlepšily. Odlehlé hodnoty měly zřejmě velký leverage.

```
fig, axs = plt.subplots(2, 2, figsize=(12, 10))

sm.qqplot(full_model.resid, line='s', ax=axs[0, 0])
axs[0, 0].set_title('Initial Model QQ Plot')

axs[0, 1].scatter(full_model.predict(X_with_constant),
full_model.resid)
axs[0, 1].axhline(y=0, color='red', linestyle='--')
axs[0, 1].set_xlabel('Predicted values')
axs[0, 1].set_ylabel('Residuals')
axs[0, 1].set_title('Initial Model Residuals vs Predicted')

sm.qqplot(final_model_cleaned.resid, line='s', ax=axs[1, 0])
axs[1, 0].set_title('Final Cleaned Model QQ Plot')
```

```
axs[1,
1].scatter(final_model_cleaned.predict(X_cleaned_with_constant),
final_model_cleaned.resid)
axs[1, 1].axhline(y=0, color='red', linestyle='--')
axs[1, 1].set_xlabel('Predicted values')
axs[1, 1].set_ylabel('Residuals')
axs[1, 1].set_title('Final Cleaned Model Residuals vs Predicted')

plt.tight_layout()
plt.show()
```



Výsledný model

Výsledný model je `final_cleaned_model`.
Rovnice: **Ping [ms] = 8.5942 + 0.0057 * ActiveUsers + 36.6538 * InteractingPct + 8.1826 * OSType_Windows - 0.0035 * ActiveUsers_x_InteractingPct + 0.0017 * ActiveUsers_x_MacOS - 0.0008 * ActiveUsers_x_Windows - 0.0010 * ActiveUsers_x_iOS**

Problematická pozorování

Pro následující pozorování jsme získali největší absolutní rezidua. Zde je vidět, že se jedná o pozorování, která jsou velmi vzdálená od ostatních pozorování.

```
residuals = final_model_cleaned.resid

data['abs_residuals'] = residuals.abs()
problematic_observations = data.sort_values(by='abs_residuals',
ascending=False)
print(problematic_observations.head())
```

```
      OSType  ActiveUsers  InteractingPct  ScrollingPct  Ping [ms]  \
82   Windows         4222          0.4858        0.5142         66
114    MacOS         4384          0.2231        0.7769         61
490      iOS         8839          0.4492        0.5508         37
254      iOS         5514          0.5974        0.4026         59
228  Android         5037          0.5616        0.4384         62

     abs_residuals
82       18.195429
114      15.367582
490      15.120873
254      14.472975
228      14.112256
```

Odhady pro Windows při průměrných hodnotách ostatních prediktorů

```
observation = {
    'const': 1,
    'ActiveUsers': np.mean(X_cleaned['ActiveUsers']),
    'InteractingPct': np.mean(X_cleaned['InteractingPct']),
    'OSType_Windows': 1,
    'ActiveUsers_x_InteractingPct':
np.mean(X_cleaned['ActiveUsers_x_InteractingPct']),
    'ActiveUsers_x_MacOS': 0,
    'ActiveUsers_x_Windows': np.mean(X_cleaned['ActiveUsers']),
    'ActiveUsers_x_iOS': 0
}

prediction_summary_frame =
final_model_cleaned.get_prediction(pd.DataFrame([observation])).summar
y_frame()

print(f"Predikovaný ping: {prediction_summary_frame['mean'][0]}")
print(f"Konfidenční interval:
({prediction_summary_frame['mean_ci_lower'][0]};
{prediction_summary_frame['mean_ci_upper'][0]})")
print(f"Predikční interval: ({prediction_summary_frame['obs_ci_lower']
[0]}; {prediction_summary_frame['obs_ci_upper'][0]})")
```

```
Predikovaný ping: 51.75873212745883
Konfidenční interval: (50.75405957403871; 52.76340468087894)
Predikční interval: (40.13747224697249; 63.37999200794516)
```

Závěrečné zhodnocení

R-squared a Adjusted R-squared: Hodnoty 0.842 a 0.840 jsou velmi dobré, což naznačuje, že model dobře vysvětluje variabilitu závislé proměnné. Je důležité také zvážit ostatní diagnostické metriky. Vysoká hodnota podmíněného čísla naznačuje problém s kolinearitou atributů, z analýzy VIF hlavně ActiveUsers_x_InteractingPct. Pokud bysme však tento odstranili, došlo by ke snížení R-squared a Adjusted R-squared, proto je dobré na tuto možnost upozornit. Dále by stálo za odstranění také OSType_Windows, jelikož jako jediný z OSType one-hot kodovaných atributů prošel zpětnou eliminací a vytváří také problém s kolinearitou u atributu ActiveUsers_x_Windows. Tyto dva by tedy bylo případně také možné odstranit, pokud to bude stát za snížení R-squared a Adjusted R-squared. Testy Omnibus a Jarque-Bera naznačují, že rezidua nemají významnou odchylku od normálního rozdělení, ale z grafu lze pozorovat, že lehce inklinují doleva a mají lehkou centrální tendenci. Ačkoliv QQ graf pro finální upravený model naznačuje, že rezidua jsou relativně normálně rozdělena, je zde několik bodů, které odchylují od teoretické přímky, což by mohlo signalizovat potenciální odchylky od normálnosti, zejména v krajích distribuce, ale po odstranění odlehlých hodnot je patrné zlepšení :)