Pawlak_Dominik_HW2

March 30, 2021

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
np.random.seed = 42
import category_encoders as ce
```

[2]:
```
allegro_df = pd.read_csv('allegro-api-transactions.csv')
allegro_df.head()
```

[2]:

	lp	date	item_id	categories	pay_option_on_delivery	pay_option_transfer	seller	price	it_is_allegro_standard	it_quantity	it_is_brand_zone	it_seller_rating	it_location
0	0	2016-04-03 21:21:08	4753602474	['Komputery', 'Dyski i napędy', 'Nośniki', 'No…	1	1	radzioch666	59.99	1	997	0	50177	Warszawa
1	1	2016-04-03 15:35:26	4773181874	['Odzież, Obuwie, Dodatki', 'Bielizna damska',…	1	1	InwestycjeNET	4.90	1	9288	0	12428	Warszawa
2	2	2016-04-03 14:14:31	4781627074	['Dom i Ogród', 'Budownictwo i Akcesoria', 'Śc…	1	1	otostyl_com	109.90	1	895	0	7389	Leszno
3	3	2016-04-03 19:55:44	4783971474	['Książki i Komiksy', 'Poradniki i albumy', 'Z…	1	1	Matfel1	18.50	0	971	0	15006	Wola Krzysztoporska
4	4	2016-04-03 18:05:54	4787908274	['Odzież, Obuwie, Dodatki', 'Ślub i wesele', '…	1	1	PPHU_RICO	19.90	1	950	0	32975	BIAŁYSTOK

```
             main_category
0                Komputery
1  Odzież, Obuwie, Dodatki
2              Dom i Ogród
3        Książki i Komiksy
4  Odzież, Obuwie, Dodatki
```

1 Target encoding

[3]:
```
means = allegro_df.groupby('it_location')['price'].mean()
print('Liczba kategorii: ' + str(len(allegro_df['it_location'].unique())))
print(means)
```

```
Liczba kategorii: 10056
it_location
 \tWarszawa,Wrocław,Częstochowa     192.070000
    *                               160.000000
    Dzierżoniów                    3700.000000
  WARSZAWA JANKI                     66.000000
 - Ząbki  koło Warszawy              29.000000
                                       …
żyrardów                            68.864615
żyrardów / jaktorów                 60.000000
żyrzyn                               9.000000
żywiec                             135.475000
↑    ←                            2.070000
Name: price, Length: 10056, dtype: float64
```

[4]:
```
TG = allegro_df.copy()
TG['it_location'] = allegro_df['it_location'].map(means)
TG.head()
```

[4]:
```
   lp                 date     item_id  \
0   0  2016-04-03 21:21:08  4753602474
1   1  2016-04-03 15:35:26  4773181874
2   2  2016-04-03 14:14:31  4781627074
3   3  2016-04-03 19:55:44  4783971474
4   4  2016-04-03 18:05:54  4787908274

                                          categories  pay_option_on_delivery  \
0  ['Komputery', 'Dyski i napędy', 'Nośniki', 'No…                       1
1  ['Odzież, Obuwie, Dodatki', 'Bielizna damska',…                       1
2  ['Dom i Ogród', 'Budownictwo i Akcesoria', 'Śc…                       1
3  ['Książki i Komiksy', 'Poradniki i albumy', 'Z…                       1
4  ['Odzież, Obuwie, Dodatki', 'Ślub i wesele', '…                       1
```

```
   pay_option_transfer         seller   price  it_is_allegro_standard  \
0                    1    radzioch666   59.99                       1
1                    1  InwestycjeNET    4.90                       1
2                    1    otostyl_com  109.90                       1
3                    1        Matfel1   18.50                       0
4                    1      PPHU_RICO   19.90                       1

   it_quantity  it_is_brand_zone  it_seller_rating  it_location  \
0          997                 0             50177    85.423398
1         9288                 0             12428    85.423398
2          895                 0              7389    61.990914
3          971                 0             15006    35.433365
4          950                 0             32975   117.191956

             main_category
0                Komputery
1  Odzież, Obuwie, Dodatki
2              Dom i Ogród
3        Książki i Komiksy
4  Odzież, Obuwie, Dodatki
```

Przewagą nad one-hot encoding jest mniejsza zajętość pamięciowa, nietworzenie dodatkowej ilości kolumn dla każdej kategorii. W tym zadaniu zmienna ma 6291 kategorii, a więc tyle dodatkowych kolumn by powstało. Jego kolejną zaletą jest, że zmienia zmienną kategoryczną na zmienną celu.

2 One-hot endoding

[5]:
```
OHE = pd.concat([
    pd.get_dummies(allegro_df.main_category, prefix='MainCategory'),
    allegro_df], axis=1).drop(['main_category'], axis=1)
print(OHE.head())
```

```
   MainCategory_Antyki i Sztuka  MainCategory_Bilety  \
0                             0                    0
1                             0                    0
2                             0                    0
3                             0                    0
4                             0                    0

   MainCategory_Biuro i Reklama  MainCategory_Biżuteria i Zegarki  \
0                             0                                 0
1                             0                                 0
2                             0                                 0
3                             0                                 0
4                             0                                 0

   MainCategory_Delikatesy  MainCategory_Dla Dzieci  MainCategory_Dom i Ogród  \
```

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

   MainCategory_Filmy  MainCategory_Fotografia  MainCategory_Gry  …  \
0                   0                        0                 0  …
1                   0                        0                 0  …
2                   0                        0                 0  …
3                   0                        0                 0  …
4                   0                        0                 0  …

                                          categories  pay_option_on_delivery  \
0  ['Komputery', 'Dyski i napędy', 'Nośniki', 'No…                       1
1  ['Odzież, Obuwie, Dodatki', 'Bielizna damska',…                       1
2  ['Dom i Ogród', 'Budownictwo i Akcesoria', 'Śc…                       1
3  ['Książki i Komiksy', 'Poradniki i albumy', 'Z…                       1
4  ['Odzież, Obuwie, Dodatki', 'Ślub i wesele', '…                       1

   pay_option_transfer         seller   price  it_is_allegro_standard  \
0                    1    radzioch666   59.99                       1
1                    1  InwestycjeNET    4.90                       1
2                    1    otostyl_com  109.90                       1
3                    1        Matfel1   18.50                       0
4                    1      PPHU_RICO   19.90                       1

   it_quantity  it_is_brand_zone  it_seller_rating          it_location
0          997                 0             50177             Warszawa
1         9288                 0             12428             Warszawa
2          895                 0              7389               Leszno
3          971                 0             15006  Wola Krzysztoporska
4          950                 0             32975            BIAŁYSTOK

[5 rows x 40 columns]
```

3 Leave One Out Encoder

[6]:
```
loe = ce.LeaveOneOutEncoder(cols = ['main_category'], return_df = True)
LOE = loe.fit_transform(allegro_df, allegro_df['price'])

print(LOE.head())
```

```
   lp                 date     item_id  \
0   0  2016-04-03 21:21:08  4753602474
1   1  2016-04-03 15:35:26  4773181874
2   2  2016-04-03 14:14:31  4781627074
```

```
3   3  2016-04-03 19:55:44  4783971474
4   4  2016-04-03 18:05:54  4787908274

                                          categories  pay_option_on_delivery  \
0  ['Komputery', 'Dyski i napędy', 'Nośniki', 'No…                       1
1  ['Odzież, Obuwie, Dodatki', 'Bielizna damska',…                       1
2  ['Dom i Ogród', 'Budownictwo i Akcesoria', 'Śc…                       1
3  ['Książki i Komiksy', 'Poradniki i albumy', 'Z…                       1
4  ['Odzież, Obuwie, Dodatki', 'Ślub i wesele', '…                       1

   pay_option_transfer         seller   price  it_is_allegro_standard  \
0                    1    radzioch666   59.99                       1
1                    1  InwestycjeNET    4.90                       1
2                    1    otostyl_com  109.90                       1
3                    1        Matfel1   18.50                       0
4                    1      PPHU_RICO   19.90                       1

   it_quantity  it_is_brand_zone  it_seller_rating          it_location  \
0          997                 0             50177             Warszawa
1         9288                 0             12428             Warszawa
2          895                 0              7389               Leszno
3          971                 0             15006  Wola Krzysztoporska
4          950                 0             32975            BIAŁYSTOK

   main_category
0     121.814331
1      75.859357
2      72.434342
3      25.028061
4      75.859080
```

4 Count Encoding

[7]:
```
pe = ce.CountEncoder(cols = ['main_category'], return_df = True)
PE = pe.fit_transform(allegro_df, allegro_df['price'])

print(PE.head())
```

```
   lp                 date     item_id  \
0   0  2016-04-03 21:21:08  4753602474
1   1  2016-04-03 15:35:26  4773181874
2   2  2016-04-03 14:14:31  4781627074
3   3  2016-04-03 19:55:44  4783971474
4   4  2016-04-03 18:05:54  4787908274

                                          categories  pay_option_on_delivery  \
0  ['Komputery', 'Dyski i napędy', 'Nośniki', 'No…                       1
```

```
1  ['Odzież, Obuwie, Dodatki', 'Bielizna damska',…                      1
2  ['Dom i Ogród', 'Budownictwo i Akcesoria', 'Śc…                      1
3  ['Książki i Komiksy', 'Poradniki i albumy', 'Z…                      1
4  ['Odzież, Obuwie, Dodatki', 'Ślub i wesele', '…                      1

   pay_option_transfer         seller   price  it_is_allegro_standard  \
0                    1    radzioch666   59.99                       1
1                    1  InwestycjeNET    4.90                       1
2                    1    otostyl_com  109.90                       1
3                    1        Matfel1   18.50                       0
4                    1      PPHU_RICO   19.90                       1

   it_quantity  it_is_brand_zone  it_seller_rating          it_location  \
0          997                 0             50177             Warszawa
1         9288                 0             12428             Warszawa
2          895                 0              7389               Leszno
3          971                 0             15006  Wola Krzysztoporska
4          950                 0             32975            BIAŁYSTOK

   main_category
0          14491
1          54257
2          91042
3          11572
4          54257
```

Wybrałem *LeaveOneOut Encoding* oraz *CountEncoding*. Pierwszy z nich działa podobnie jak *Target Encoding*, z tą różnicą, że przy wyliczaniu średniej nie bierze pod uwagę danego rekordu. Działa tylko, jeżeli zmienna celu jest typu binarnego albo ciągłego. Natomiast *CountEncoding* zamienia nazwę kategorii na liczbę jej wystąpień, dlatego nie ma żadnych ograniczeń co do typu zmiennej celu.

5 Zad 2

[8]:
```
allegro_df_mod = allegro_df.loc[:10000, ['price', 'it_seller_rating',␣
 ↪'it_quantity']]
print(allegro_df_mod)
cols = allegro_df_mod.columns

rmse = [0] * 10
sr = np.mean(allegro_df_mod['it_seller_rating'])
print(sr)
```

```
       price  it_seller_rating  it_quantity
0      59.99             50177          997
1       4.90             12428         9288
2     109.90              7389          895
3      18.50             15006          971
```

```
4       19.90              32975          950
…       …                  …              …
9996    11.97                110         9833
9997    11.97                110         9833
9998    11.97                110         9833
9999    11.97                110         9833
10000   28.00                877            2

[10001 rows x 3 columns]
20079.188381161883
```

[9]:
```
from sklearn.impute import KNNImputer

remove_n = int(0.1 * len(allegro_df_mod))

for r in range (len(rmse)):
    allegro_tmp = allegro_df_mod.copy()
    r_temp = 0
    drop_indices = np.array(np.random.choice(allegro_df_mod.index, remove_n,␣
 ↪replace=False))

    for i in drop_indices:
        allegro_tmp.loc[i, 'it_seller_rating'] = np.NaN

    imputer = KNNImputer(n_neighbors=3, weights="uniform")
    NN = pd.DataFrame(imputer.fit_transform(allegro_tmp), columns = cols)

    for i in range (len(NN)):
        r_temp += (sr - NN.loc[i, 'it_seller_rating'])**2
    r_temp /= len(NN)
    r_temp = np.sqrt(r_temp).round(2)

    rmse[r] = r_temp
print(rmse)
std_r = np.std(rmse)
mean_r = np.mean(rmse)
```

```
[34611.88, 34521.23, 34639.38, 35024.98, 34764.05, 34555.94, 34639.74, 34577.16,
35133.01, 34675.88]
```

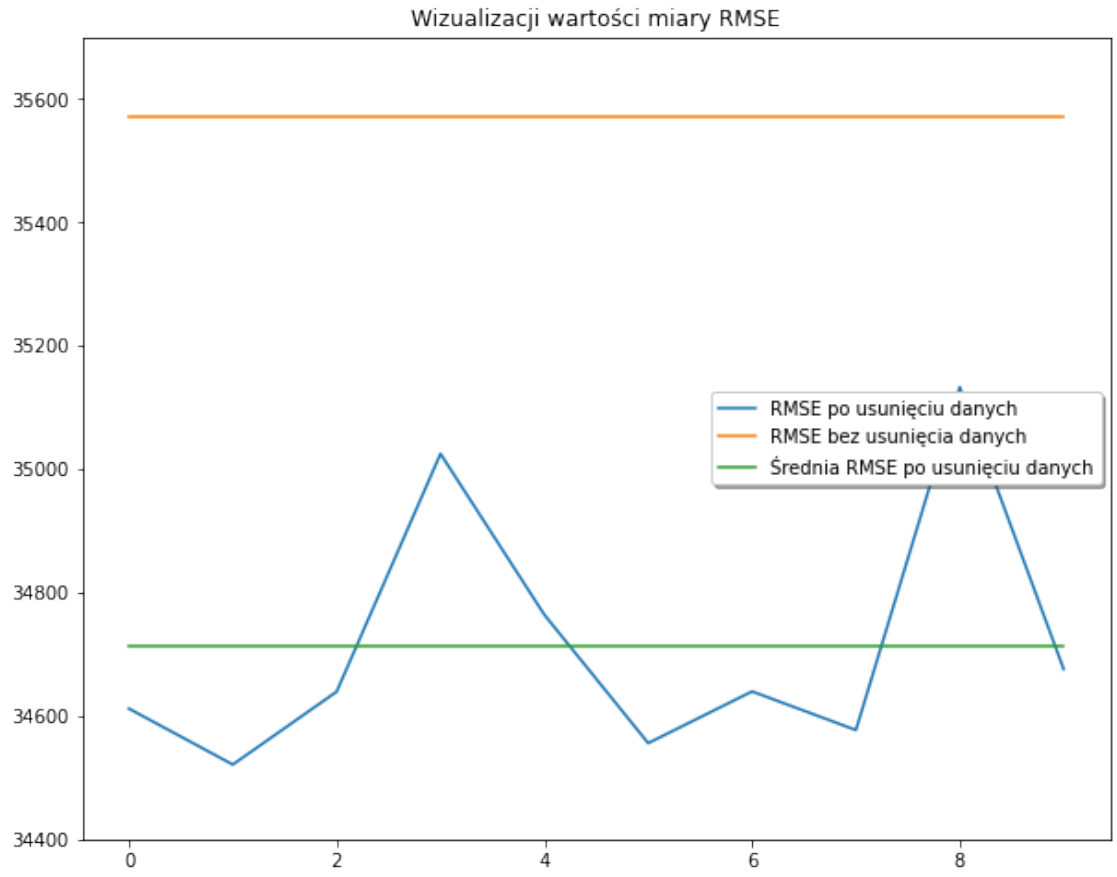
[10]:
```
rmse_org = 0
for i in range (len(allegro_df_mod)):
    rmse_org += (sr - allegro_df_mod.loc[i, 'it_seller_rating'])**2
rmse_org = rmse_org / len(allegro_df_mod)
rmse_org = np.sqrt(rmse_org).round(2)

rmse_org_IQ = 0
for i in range (len(allegro_df_mod)):
```

```
    rmse_org_IQ += (sr - allegro_df_mod.loc[i, 'it_quantity'])**2
rmse_org_IQ = rmse_org_IQ / len(allegro_df_mod)
rmse_org_IQ = np.sqrt(rmse_org_IQ).round(2)
```

[11]:
```
print(f'Miara RMSE dla oryginalnych wartości: {rmse_org}.')
print(f'Miary RMSE dla losowo usuniętych: {rmse}.')
print(f'Średnia miary RMSE dla losowo usuniętych: {mean_r:.2f}.')
print(f'Odchylenie standardowe miary RMSE dla losowo usuniętych: {std_r:.2f}.')
```

```
Miara RMSE dla oryginalnych wartości: 35571.06.
Miary RMSE dla losowo usuniętych: [34611.88, 34521.23, 34639.38, 35024.98,
34764.05, 34555.94, 34639.74, 34577.16, 35133.01, 34675.88].
Średnia miary RMSE dla losowo usuniętych: 34714.32.
Odchylenie standardowe miary RMSE dla losowo usuniętych: 194.58.
```

[12]:
```
fig, ax = plt.subplots(1, 1, figsize=(10, 8))
plt.plot(rmse, label = 'RMSE po usunięciu danych')
plt.plot(range(10), [rmse_org]*10, label = 'RMSE bez usunięcia danych')
plt.plot(range(10), [mean_r]*10, label = 'Średnia RMSE po usunięciu danych')
ax.set_title('Wizualizacji wartości miary RMSE')
ax.set_ylim([34400, 35700])
ax.legend()
legend = ax.legend(loc='center right', shadow=True, fontsize='medium')
plt.show()
```



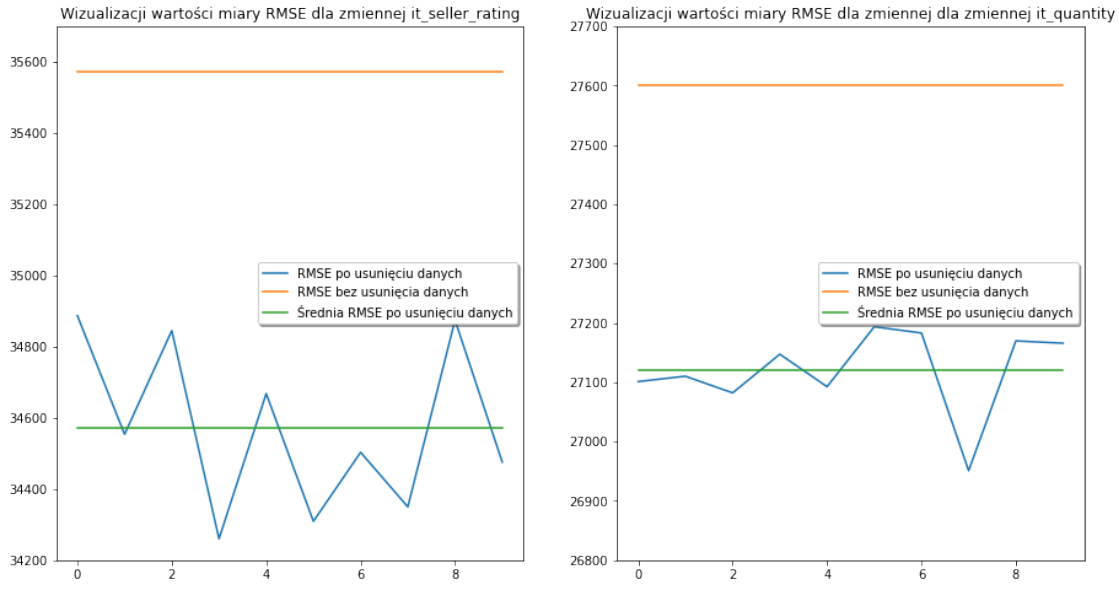
[13]:
```
rmse_SR = [0]*10
rmse_IQ = [0]*10

for r in range (len(rmse)):
    allegro_tmp = allegro_df_mod.copy()
    r_temp_SR = 0
    r_temp_IQ = 0

    drop_indices = np.array(np.random.choice(allegro_df_mod.index, remove_n,␣
 ↪replace=False))
    for i in drop_indices:
        allegro_tmp.loc[i, 'it_seller_rating'] = np.NaN

    drop_indices = np.array(np.random.choice(allegro_df_mod.index, remove_n,␣
 ↪replace=False))
    for i in drop_indices:
        allegro_tmp.loc[i, 'it_quantity'] = np.NaN

    imputer = KNNImputer(n_neighbors=3, weights="uniform")
```

```
    NN = pd.DataFrame(imputer.fit_transform(allegro_tmp), columns = cols)

    for i in range (len(NN)):
        r_temp_SR += (sr - NN.loc[i, 'it_seller_rating'])**2
        r_temp_IQ += (sr - NN.loc[i, 'it_quantity'])**2
    r_temp_SR /= len(NN)
    r_temp_SR = np.sqrt(r_temp_SR).round(2)
    r_temp_IQ /= len(NN)
    r_temp_IQ = np.sqrt(r_temp_IQ).round(2)

    rmse_SR[r] = r_temp_SR
    rmse_IQ[r] = r_temp_IQ
std_SR = np.std(rmse_SR)
mean_SR = np.mean(rmse_SR)
std_IQ = np.std(rmse_IQ)
mean_IQ = np.mean(rmse_IQ)

mean_SR_org = allegro_df_mod['it_seller_rating'].mean()
mean_IQ_org = allegro_df_mod['it_quantity'].mean()
```

[14]:
```
x = 'it_seller_rating'
print(f'Miara RMSE dla oryginalnych wartości zmiennej {x}: {rmse_org}.')
print(f'Miary RMSE dla losowo usuniętych zmiennej {x}: {rmse_SR}.')
print(f'Średnia miary RMSE dla losowo usuniętych zmiennej {x}: {mean_SR:.2f}.')
print(f'Odchylenie standardowe miary RMSE dla losowo usuniętych zmiennej {x}:␣
 ↪{std_SR:.2f}.')
print()

x = 'it_quantity'
print(f'Miara RMSE dla oryginalnych wartości zmiennej {x}: {rmse_org_IQ}.')
print(f'Miary RMSE dla losowo usuniętych zmiennej {x}: {rmse_IQ}.')
print(f'Średnia miary RMSE dla losowo usuniętych zmiennej {x}: {mean_IQ:.2f}.')
print(f'Odchylenie standardowe miary RMSE dla losowo usuniętych zmiennej {x}:␣
 ↪{std_IQ:.2f}.')
```

```
Miara RMSE dla oryginalnych wartości zmiennej it_seller_rating: 35571.06.
Miary RMSE dla losowo usuniętych zmiennej it_seller_rating: [34886.92, 34553.85,
34844.96, 34260.43, 34668.0, 34309.46, 34503.19, 34350.02, 34874.24, 34475.89].
Średnia miary RMSE dla losowo usuniętych zmiennej it_seller_rating: 34572.70.
Odchylenie standardowe miary RMSE dla losowo usuniętych zmiennej
it_seller_rating: 224.46.

Miara RMSE dla oryginalnych wartości zmiennej it_quantity: 27600.3.
Miary RMSE dla losowo usuniętych zmiennej it_quantity: [27101.13, 27110.15,
27082.0, 27147.23, 27092.48, 27193.42, 27182.96, 26950.97, 27169.64, 27165.76].
Średnia miary RMSE dla losowo usuniętych zmiennej it_quantity: 27119.57.
Odchylenie standardowe miary RMSE dla losowo usuniętych zmiennej it_quantity:
67.73.
```

[15]:
```
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(15, 8))

ax1.plot(rmse_SR, label = 'RMSE po usunięciu danych')
ax1.plot(range(10), [rmse_org]*10, label = 'RMSE bez usunięcia danych')
ax1.plot(range(10), [mean_SR]*10, label = 'Średnia RMSE po usunięciu danych')
ax1.set_title('Wizualizacji wartości miary RMSE dla zmiennej it_seller_rating')
ax1.set_ylim([34200, 35700])
ax1.legend()
legend = ax1.legend(loc='center right', shadow=True, fontsize='medium')

ax2.plot(rmse_IQ, label = 'RMSE po usunięciu danych')
ax2.plot(range(10), [rmse_org_IQ]*10, label = 'RMSE bez usunięcia danych')
ax2.plot(range(10), [mean_IQ]*10, label = 'Średnia RMSE po usunięciu danych')
ax2.set_title('Wizualizacji wartości miary RMSE dla zmiennej dla zmiennej␣
 ↪it_quantity')
ax2.set_ylim([26800, 27700])
ax2.legend()
legend = ax2.legend(loc='center right', shadow=True, fontsize='medium')

plt.show()
```



Widzimy, że usunięcie wartości zmniejsza wartość miary RMSE dla obu zmiennych. W drugim przypadku, po usunięciu danych z dwóch kolumn wartość miary RMSE dla zmiennej *it_seller_rating* była niższa.