Praca domowa 2

Bartosz Siński

In [64]:
```
import pandas as pd
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt
from category_encoders import TargetEncoder
from category_encoders import HashingEncoder
from category_encoders import BinaryEncoder
from sklearn.impute import KNNImputer
from sklearn.metrics import mean_squared_error
```

In [3]:
```
df_trans = pd.read_csv("./src/allegro-api-transactions.csv")
```

In [153…
```
df_trans.head(5)
```

Out[153…

	lp	date	item_id	categories	pay_option_on_delivery	pay_option_transfer	seller	price
0	0	2016-04-03 21:21:08	4753602474	['Komputery', 'Dyski i napędy', 'Nośniki', 'No...	1	1	radzioch666	59.99
1	1	2016-04-03 15:35:26	4773181874	['Odzież, Obuwie, Dodatki', 'Bielizna damska',...	1	1	InwestycjeNET	4.90
2	2	2016-04-03 14:14:31	4781627074	['Dom i Ogród', 'Budownictwo i Akcesoria', 'Śc...	1	1	otostyl_com	109.90
3	3	2016-04-03 19:55:44	4783971474	['Książki i Komiksy', 'Poradniki i albumy', 'Z...	1	1	Matfel1	18.50
4	4	2016-04-03 18:05:54	4787908274	['Odzież, Obuwie, Dodatki', 'Ślub i wesele', '...	1	1	PPHU_RICO	19.90

1.Kodowanie zmiennych kategorycznych

In [6]:
```
df_trans['it_location'] = TargetEncoder().fit_transform(df_trans['it_location'],df_tra
df_trans
```

```
C:\Users\komp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.8_qbz5n2kfra8p0
\LocalCache\local-packages\Python38\site-packages\category_encoders\utils.py:21: Futur
eWarning: is_categorical is deprecated and will be removed in a future version.  Use i
s_categorical_dtype instead
  elif pd.api.types.is_categorical(cols):
```

Out[6]:

	lp	date	item_id	categories	pay_option_on_delivery	pay_option_transfer	
0	0	2016-04-03 21:21:08	4753602474	['Komputery', 'Dyski i napędy', 'Nośniki', 'No...	1	1	radzi
1	1	2016-04-03 15:35:26	4773181874	['Odzież, Obuwie, Dodatki', 'Bielizna damska',...	1	1	Inwesty
2	2	2016-04-03 14:14:31	4781627074	['Dom i Ogród', 'Budownictwo i Akcesoria', 'Śc...	1	1	otos
3	3	2016-04-03 19:55:44	4783971474	['Książki i Komiksy', 'Poradniki i albumy', 'Z...	1	1	
4	4	2016-04-03 18:05:54	4787908274	['Odzież, Obuwie, Dodatki', 'Ślub i wesele', '...	1	1	PPHU
...	...	...	...	...	...	...	
420015	420015	2016-04-03 20:27:13	6099625607	['RTV i AGD', 'Sprzęt audio dla domu', 'Odtwar...	0	0	iwo
420016	420016	2016-04-03 22:35:02	6099634607	['Uroda', 'Makijaż', 'Oczy', 'Tusze do rzęs']	1	1	Dolce_Co
420017	420017	2016-04-03 22:38:57	6099780407	['Odzież, Obuwie, Dodatki', 'Przebrania, kosti...	1	1	pe
420018	420018	2016-04-03 22:44:17	6099801007	['Dla Dzieci', 'Rowery i pojazdy', 'Rowery bie...	1	0	k
420019	420019	2016-04-03 23:08:23	6099873207	['Motoryzacja', 'Części samochodowe', 'Koła, f...	0	0	Mal

420020 rows × 14 columns

Przewagą target encoding nad one-hot encoding jest to, że one-hot encoding dla wielu zmiennych kategorycznych może w sposób znaczący zwiększać wymiarowość naszego zbioru danych. Będzie to szczególnie uciążliwe dla zbioru danych zawierających małą liczbę rekordów.

In [11]:
```
df_trans_onehot = pd.get_dummies(df_trans,prefix="",prefix_sep="", columns=['main_cate
df_trans_onehot
```

Out[11]:

	lp	date	item_id	categories	pay_option_on_delivery	pay_option_transfer	
0	0	2016-04-03 21:21:08	4753602474	['Komputery', 'Dyski i napędy', 'Nośniki', 'No...	1	1	radzi
1	1	2016-04-03 15:35:26	4773181874	['Odzież, Obuwie, Dodatki', 'Bielizna damska',...	1	1	Inwesty
2	2	2016-04-03 14:14:31	4781627074	['Dom i Ogród', 'Budownictwo i Akcesoria', 'Śc...	1	1	otos
3	3	2016-04-03 19:55:44	4783971474	['Książki i Komiksy', 'Poradniki i albumy', 'Z...	1	1	
4	4	2016-04-03 18:05:54	4787908274	['Odzież, Obuwie, Dodatki', 'Ślub i wesele', '...	1	1	PPHU
...	...	...	...	...	...	...	
420015	420015	2016-04-03 20:27:13	6099625607	['RTV i AGD', 'Sprzęt audio dla domu', 'Odtwar...	0	0	iwo
420016	420016	2016-04-03 22:35:02	6099634607	['Uroda', 'Makijaż', 'Oczy', 'Tusze do rzęs']	1	1	Dolce_Co
420017	420017	2016-04-03 22:38:57	6099780407	['Odzież, Obuwie, Dodatki', 'Przebrania, kosti...	1	1	pe
420018	420018	2016-04-03 22:44:17	6099801007	['Dla Dzieci', 'Rowery i pojazdy', 'Rowery bie...	1	0	k
420019	420019	2016-04-03 23:08:23	6099873207	['Motoryzacja', 'Części samochodowe', 'Koła, f...	0	0	Mal

420020 rows × 40 columns

In [76]:
```
df_trans_hashing = HashingEncoder(cols='main_category').fit_transform(df_trans)
df_trans_hashing
```

Out[76]:

	col_0	col_1	col_2	col_3	col_4	col_5	col_6	col_7	lp	date	...	categories	pay_opt
0	0	0	0	0	0	0	1	0	0	2016-04-03 21:21:08	...	['Komputery', 'Dyski i napędy', 'Nośniki', 'No...	
1	0	0	1	0	0	0	0	0	1	2016-04-03 15:35:26	...	['Odzież, Obuwie, Dodatki', 'Bielizna damska',...	
2	0	0	0	0	0	0	1	0	2	2016-04-03 14:14:31	...	['Dom i Ogród', 'Budownictwo i Akcesoria', 'Śc...	
3	0	0	1	0	0	0	0	0	3	2016-04-03 19:55:44	...	['Książki i Komiksy', 'Poradniki i albumy', 'Z...	
4	0	0	1	0	0	0	0	0	4	2016-04-03 18:05:54	...	['Odzież, Obuwie, Dodatki', 'Ślub i wesele', '...	
...	...	...	...	...	...	...	...	...	...	...	...	...	
420015	0	0	0	1	0	0	0	0	420015	2016-04-03 20:27:13	...	['RTV i AGD', 'Sprzęt audio dla domu', 'Odtwar...	
420016	0	0	0	0	1	0	0	0	420016	2016-04-03 22:35:02	...	['Uroda', 'Makijaż', 'Oczy', 'Tusze do rzęs']	
420017	0	0	1	0	0	0	0	0	420017	2016-04-03 22:38:57	...	['Odzież, Obuwie, Dodatki', 'Przebrania, kosti...	
420018	0	0	0	0	0	1	0	0	420018	2016-04-03 22:44:17	...	['Dla Dzieci', 'Rowery i pojazdy', 'Rowery bie...	
420019	0	0	0	0	0	0	0	1	420019	2016-04-03 23:08:23	...	['Motoryzacja', 'Części samochodowe', 'Koła, f...	

420020 rows × 21 columns

In [22]:
```
df_trans_binary = BinaryEncoder(cols='main_category').fit_transform(df_trans)
df_trans_binary
```

```
C:\Users\komp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.8_qbz5n2kfra8p0
\LocalCache\local-packages\Python38\site-packages\category_encoders\utils.py:21: Futur
eWarning: is_categorical is deprecated and will be removed in a future version.  Use i
s_categorical_dtype instead
  elif pd.api.types.is_categorical(cols):
```

Out[22]:

	lp	date	item_id	categories	pay_option_on_delivery	pay_option_transfer	
0	0	2016-04-03 21:21:08	4753602474	['Komputery', 'Dyski i napędy', 'Nośniki', 'No...	1	1	radzi
1	1	2016-04-03 15:35:26	4773181874	['Odzież, Obuwie, Dodatki', 'Bielizna damska',...	1	1	Inwesty
2	2	2016-04-03 14:14:31	4781627074	['Dom i Ogród', 'Budownictwo i Akcesoria', 'Śc...	1	1	otos
3	3	2016-04-03 19:55:44	4783971474	['Książki i Komiksy', 'Poradniki i albumy', 'Z...	1	1	
4	4	2016-04-03 18:05:54	4787908274	['Odzież, Obuwie, Dodatki', 'Ślub i wesele', '...	1	1	PPHU
...	...	...	...	...	...	...	
420015	420015	2016-04-03 20:27:13	6099625607	['RTV i AGD', 'Sprzęt audio dla domu', 'Odtwar...	0	0	iwo
420016	420016	2016-04-03 22:35:02	6099634607	['Uroda', 'Makijaż', 'Oczy', 'Tusze do rzęs']	1	1	Dolce_Co
420017	420017	2016-04-03 22:38:57	6099780407	['Odzież, Obuwie, Dodatki', 'Przebrania, kosti...	1	1	pe
420018	420018	2016-04-03 22:44:17	6099801007	['Dla Dzieci', 'Rowery i pojazdy', 'Rowery bie...	1	0	k
420019	420019	2016-04-03 23:08:23	6099873207	['Motoryzacja', 'Części samochodowe', 'Koła, f...	0	0	Mal

420020 rows × 19 columns

One-hot encoding każdej zmiennej kategorycznej przypisuje wartość 1 lub 0, co może spowodowac powstanie wielu nowych kolumn i zwiększenie wymiarowości naszego zbioru danych. Hash Encoder hashuje wartości naszych zmiennych kategorycznych za pomocą zer i jedynek w n-nowych wymiarach. n jest ustawiane przez użytkownika co pozwala na zmniejszego liczby nowo powstałych kolumn. Jeżeli jednak damy za małe n dla liczby kategorii naszej zmiennej możemy spowodować, że różne kategorie będą miały ten sam klucz hashujący. W binary encoding na początku przekształcamy każdy kategorie na liczbę, a później zapisujemy ją z uzyciem nowych kolumn w formie binarnej.

2. Uzupełnienie braków

Do tego ćwiczenia ograniczami liczbę rekordów, ponieważ czasy wykonywania niektórych poleceń były zbyt długie.

In [94]:
```
df_trans_num = df_trans.loc[0:42000,['price','it_seller_rating','it_quantity']]
df_trans_num
```

Out[94]:

	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
...	...	...	...
41996	15.49	2111	2
41997	2.40	9306	1515
41998	2.40	9306	1515
41999	2.00	15108	0
42000	31.90	58407	10

42001 rows × 3 columns

In [154…
```
np.random.seed = 42
rows = np.random.randint(42000,size=4200)
df_trans_num_nan = df_trans_num.copy()
df_trans_num_nan.loc[rows,['it_seller_rating']] = np.nan
df_trans_num_nan.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 42001 entries, 0 to 42000
Data columns (total 3 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   price             42001 non-null  float64
 1   it_seller_rating  38019 non-null  float64
 2   it_quantity       42001 non-null  int64  
dtypes: float64(2), int64(1)
memory usage: 984.5 KB
```

In [138…
```
df_trans_num0 = KNNImputer(n_neighbors=2, weights="uniform").fit_transform(df_trans_nu
df_trans_num0 = pd.DataFrame(df_trans_num0, columns = ['price','it_seller_rating','it_
mean_squared_error(df_trans_num[['it_seller_rating']],df_trans_num0[['it_seller_rating
```

Out[138…
```
11480.113498137582
```

In [124…
```
results=[]
for i in range(10):
    rows = np.random.randint(42000,size=4200)
    df_trans_num_nan = df_trans_num.copy()
    df_trans_num_nan.loc[rows,['it_seller_rating']] = np.nan
    df_trans_numn = KNNImputer(n_neighbors=5, weights="uniform").fit_transform(df_tran
    df_trans_numn = pd.DataFrame(df_trans_numn, columns = ['price','it_seller_rating',
    results.append(mean_squared_error(df_trans_num[['it_seller_rating']],df_trans_numn
results
```

Out[124…
```
[10564.600834113731,
 9992.760400096075,
 10584.291893799562,
 10451.74108676633,
 10619.484925893386,
 10771.398424160236,
 10478.281088182579,
 10479.164839099258,
 11545.609595426336,
 10162.721629064645]
```

In [131…
```
results2=[]
for i in range(10):
    rows = np.random.randint(42000,size=4200)
    rows2 = np.random.randint(42000,size=4200)
    df_trans_num_nan = df_trans_num.copy()
    df_trans_num_nan.loc[rows,['it_seller_rating']] = np.nan
    df_trans_num_nan.loc[rows2,['it_quality']] = np.nan
    df_trans_numn = KNNImputer(n_neighbors=5, weights="uniform").fit_transform(df_tran
    df_trans_numn = pd.DataFrame(df_trans_numn, columns = ['price','it_seller_rating',
    results2.append(mean_squared_error(df_trans_num[['it_seller_rating']],df_trans_nur
results2
```

Out[131…
```
[11068.384934664069,
 10386.379369630931,
 10367.339464944635,
 9937.964992207473,
 10954.33570429183,
 11032.97871210406,
 10344.344936546402,
 10748.24741873876,
 10723.349179136445,
 11218.042721545351]
```

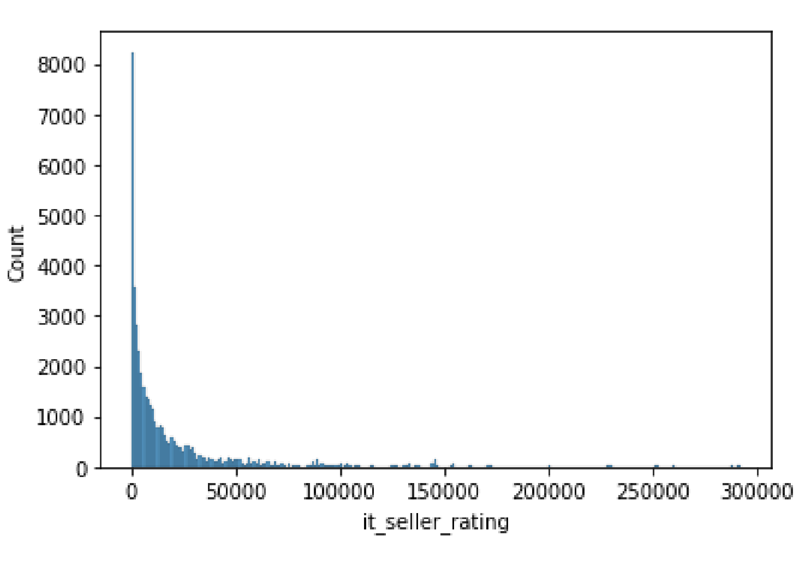
In [132…
```
print("Odchylenie standardowe błędów wartości zmiennej it_seller_rating  dla: jednej z
print("Średni błąd RMSE it_seller_rating dla: jednej zmiennej usuniętej="+str(np.mean(
```

```
Odchylenie standardowe błędów wartości zmiennej it_seller_rating  dla: jednej zmiennej
usuniętej=390.45331409134707, dwóch zmiennych usuniętych=386.51506839255734
Średni błąd RMSE it_seller_rating dla: jednej zmiennej usuniętej=10565.005471660214, d
wóch zmiennych usuniętych=10678.136743380997
```

Wartości błędów są rzędu 10^4, czyli taki samo jak wartości it_seller_rating więc wypełnienie luk nie jest dokładne. Może być to spowodowane tym, że szukamy sąsiadów na podstawie tylko dwóch pozostałych zmiennych. Widzimy także na poniższym wykresie, że wartości są rozłożone nierównomiernie co także może powodować wysoki błąd metody najbliższych sąsiadów. Wynik są też dokładniejsze gdy zmienne na podstawie, których wypełniamy braki, także nie posiadają wartości None lub NaN.

In [144…
```
sns.histplot(data=df_trans_numn,x="it_seller_rating")
```

Out[144…
```
<AxesSubplot:xlabel='it_seller_rating', ylabel='Count'>
```



Poniżej znajduje się histogram błędów RMSE dwóch pomiarów, *results1* to eksperymenty gdzie brakowało jednej zmiennej, a *results2* to eksperymenty, gdzie brakowało dwóch zmiennych.

In [152…
```
df_results=pd.DataFrame({'results1':results,'results2':results2})
df_results= pd.melt(df_results)
sns.histplot(data=df_results,x="value",hue="variable",multiple = "dodge")
```

Out[152…
```
<AxesSubplot:xlabel='value', ylabel='Count'>
```