profiles

October 20, 2024

[1]:
```
# -*- coding: utf-8 -*-
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
# http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
# implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#
```

[2]:
```
def test():
    print("Success")
```

0.0.1 Adam Candrák/Mária Matušisková - 50%/50%

1 Imports

[3]:
```
import pandas as pd
import numpy as np
import scipy.stats as stats
from matplotlib import pyplot as plt
import seaborn as sns
```

2 Phase 1 - Exploratory analysis

2.1 1.1 Basic description of the data together with characteristics

2.2 ### EDA with visualization

Analysis of data structures such as files (structures and relations, number, types, ...), records (structures, number of records, number of attributes, types, ...)

1. Load dataset Profiles

[4]:
```
connections_file = "../data/Connections.csv"
devices_file = "../data/Devices.csv"
processes_file = "../data/Processes.csv"
profiles_file = "../data/Profiles.csv"

connections = pd.read_csv(connections_file, sep='\t')
devices = pd.read_csv(devices_file, sep='\t')
processes = pd.read_csv(processes_file, sep='\t')
profiles = pd.read_csv(profiles_file, sep='\t')
```

- **Profiles** - The dataset contains user data. The dataset has these attributes:
  - username
  - ssn - the social security number
  - mail - email of the user
  - residence
  - birthdate
  - imei - International Mobile Equipment Identity - It is a unique number that helps identify device or track it when it is lost. Furthermore, it is preventing from unauthorized network access.
  - user_id
  - registration
  - job
  - company
  - address
  - name

[5]:
```
profiles.columns
```

[5]:
```
Index(['username', 'ssn', 'mail', 'residence', 'birthdate', 'imei', 'user_id',
       'registration', 'job', 'company', 'address', 'name'],
      dtype='object')
```

Types of the columns:

[6]:
```
profiles.dtypes
```

[6]:
```
username        object
ssn             object
mail            object
residence       object
birthdate       object
imei             int64
user_id          int64
registration    object
job             object
company         object
address         object
```

```
name            object
dtype: object
```

The size of the dataset is 31 044.

[7]:
```
profiles.size
```

[7]:
```
31044
```

Shows the first lines of the dataset.

[8]:
```
profiles.head()
```

[8]:
```
       username          ssn                     mail  \
0         joy86  312-79-2503       marc49@gmail.com
1  andrewturner  275-09-2121    sedwards@hotmail.com
2  rogersrobert  592-96-3718  martinhuber@gmail.com
3     bethany94  171-66-6416    kwilliams@yahoo.com
4   nathanmoore  149-50-1186       shardy@yahoo.com

                                    residence   birthdate                 imei  \
0                                         NaN  1943-08-29  3590433799317661966
1  75135 Smith Square\nPort Nathan, AR 79925  1993-03-09  3590433799317661792
2                                         NaN  1935-10-23  8630330696303482253
3                                         NaN  2021-04-12   863033069630348065
4                                         NaN  2019-11-10   863033069630348081

   user_id          registration                                job  \
0     1888  04/07/2021, 00:00:00                Market researcher
1     1942            2021-05-21  Administrator, local government
2      598  03/24/2024, 00:00:00                               NaN
3     1070            2021-03-31                               NaN
4     1369            2022-06-26                 Sports therapist

                     company                                          address  \
0               Simpson Inc  9105 Cox Curve Apt. 055\nPricemouth, AR 23186
1  Deleon, Duncan and Garcia                                              NaN
2               Ramirez PLC                                              NaN
3                  Lane Ltd                      USNV Castro\nFPO AE 61560
4               Smith Group   4986 Richard Ford\nPort Susanport, DC 06966

              name
0     Dakota Stark
1  Valerie Mitchell
2    Michael Martin
3    Bruce Williams
4         Tara Wood
```

See more info about the dataset… There is a rule that the columns should not have null values.

[9]:
```
profiles.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2587 entries, 0 to 2586
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   username      2587 non-null   object
 1   ssn           2587 non-null   object
 2   mail          2587 non-null   object
 3   residence     905 non-null    object
 4   birthdate     1423 non-null   object
 5   imei          2587 non-null   int64 
 6   user_id       2587 non-null   int64 
 7   registration  2587 non-null   object
 8   job           776 non-null    object
 9   company       2587 non-null   object
 10  address       2199 non-null   object
 11  name          2587 non-null   object
dtypes: int64(2), object(10)
memory usage: 242.7+ KB
```

Let's see the descriptive statistics for data distribution: - count - the final number of the non-null values - mean - the average of the values in the each column - std - the standard deviation (how spread out the data are) - min - the smallest value in the each column - 25% - the value closest to the 25% metric of data - 50% - the value closest to the 50% metric of data - 75% - the value closest to the 75% metric of data - max - the highest value in the each column

[10]:
```
profiles.describe()
```

[10]:

	imei	user_id
count	2.587000e+03	2587.000000
mean	3.978629e+18	1309.511403
std	3.361321e+18	748.646640
min	3.590434e+17	0.000000
25%	8.630331e+17	659.500000
50%	3.590434e+18	1292.000000
75%	8.630331e+18	1953.000000
max	8.630331e+18	2586.000000

[11]:
```
profiles.describe(exclude=np.number)
```

[11]:

	username	ssn	mail	\
count	2587	2587	2587	
unique	2547	2587	2574	
top	nanderson	312-79-2503	csmith@hotmail.com	
freq	3	1	2	

```
                                          residence   birthdate registration  \
count                                           905        1423          2587
unique                                          905        1394          2291
top     75135 Smith Square\nPort Nathan, AR 79925  1911-05-11   2023/02/02
freq                                              1           2             3

                            job    company  \
count                       776       2587
unique                      451       2427
top     Chartered loss adjuster  Smith PLC
freq                          7          7

                                                address               name
count                                            2199               2587
unique                                           2199               2530
top     9105 Cox Curve Apt. 055\nPricemouth, AR 23186  Kimberly Williams
freq                                                1                  4
```

Number of rows and columns:

[12]:
```
profiles.shape
```

[12]:
```
(2587, 12)
```

3. Analyze Data Structure

- **Profiles**

Count elements (distinct)

[13]:
```
profiles.nunique()
```

[13]:
```
username        2547
ssn             2587
mail            2574
residence        905
birthdate       1394
imei             497
user_id         1633
registration    2291
job              451
company         2427
address         2199
name            2530
dtype: int64
```

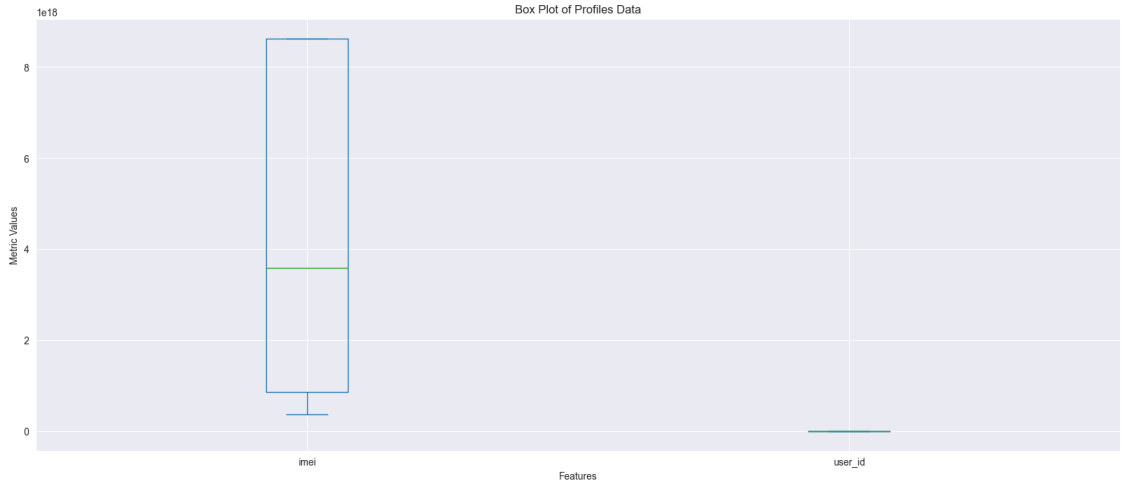
Analysis of individual attributes: for selected significant attributes (min 10), analyze their distributions and basic descriptive statistics. Summarize the distribution of various usages of apps, while excluding imei and ts, because those are not numerical values.

[14]:
```
fig, ax = plt.subplots(figsize=(20, 8))

profiles[profiles.columns.difference(['username', 'ssn', 'mail', 'residence',␣
↪'birthdate', 'registration', 'job', 'company', 'address', 'name'])].plot.
↪box(ax=ax)

plt.xlabel('Features')
plt.ylabel('Metric Values')
plt.title('Box Plot of Profiles Data')
```

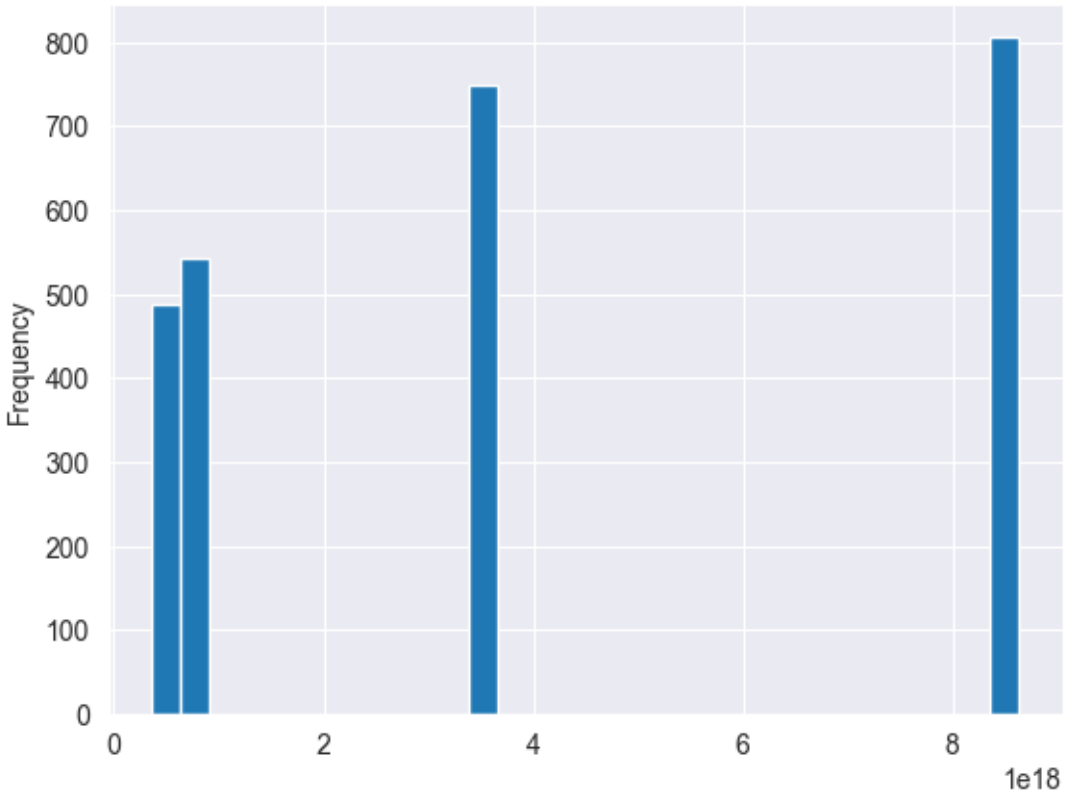
[14]:
```
Text(0.5, 1.0, 'Box Plot of Profiles Data')
```



Occurrence of values from the column imei:

[15]:
```
profiles['imei'].plot.hist(bins=30)
```

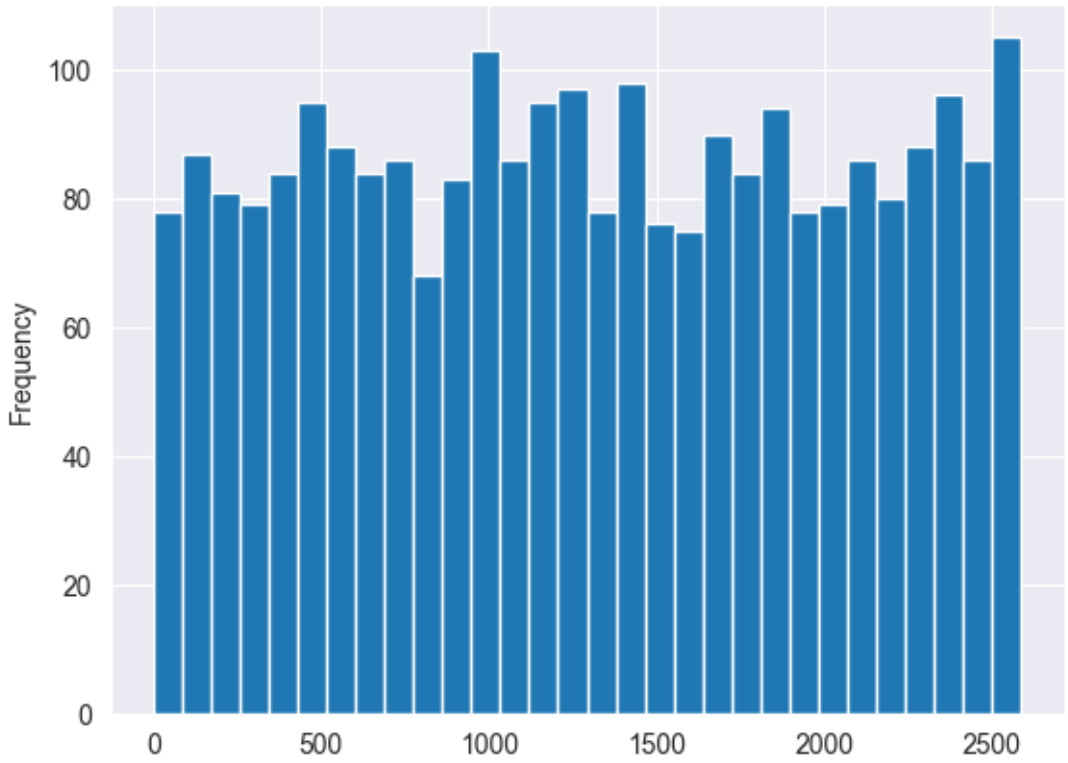
[15]:
```
<Axes: ylabel='Frequency'>
```



Occurrence of values from the column user_id:

[16]:
```
profiles['user_id'].plot.hist(bins=30)
```

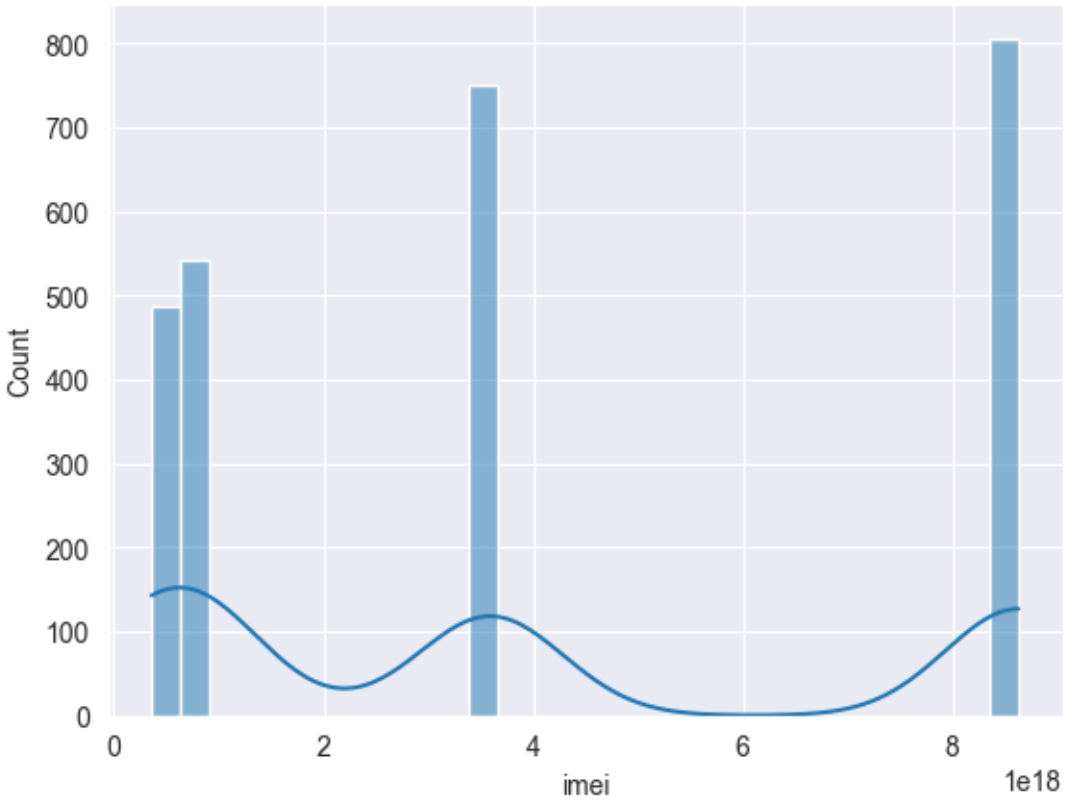
[16]:
```
<Axes: ylabel='Frequency'>
```



Data distribution:

[17]:
```
sns.histplot(profiles['imei'], kde=True, bins=30)
```

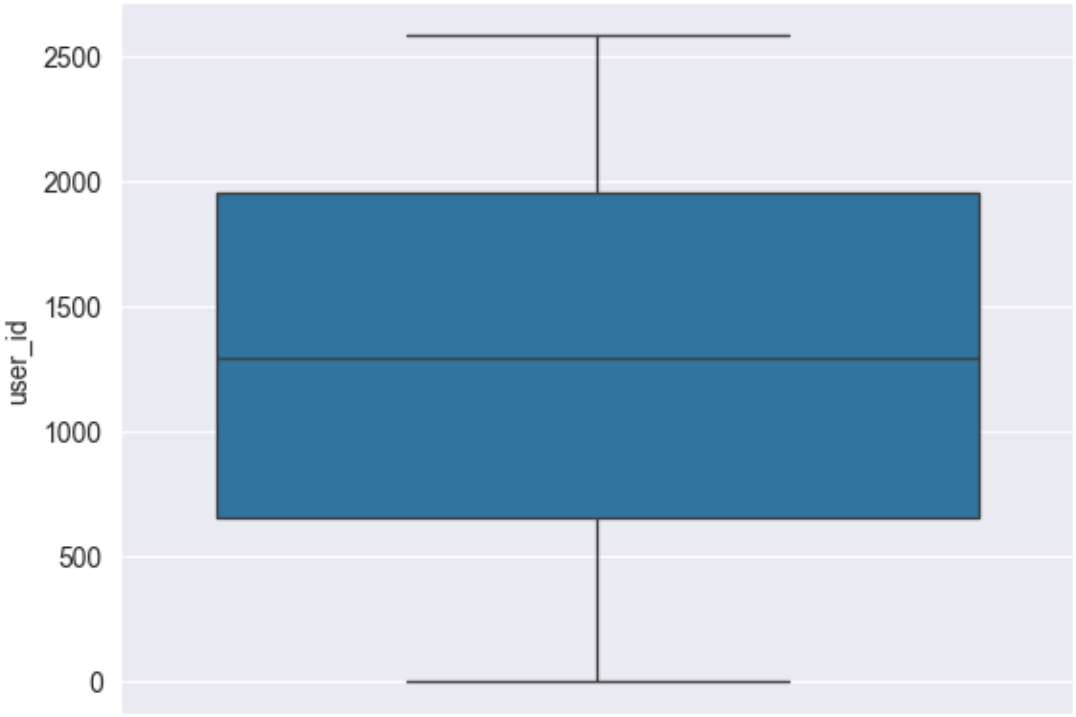
[17]:
```
<Axes: xlabel='imei', ylabel='Count'>
```



Create a boxplot for column user_id, the median is between 1500 and 1000:

[18]:
```
sns.boxplot(profiles['user_id'])
```

[18]:
```
<Axes: ylabel='user_id'>
```



Pairwise data analysis: Identify relationships and dependencies between pairs of attributes. Calculate correlations:

[19]:
```
profiles_data = profiles[['imei', 'user_id']]
profiles_data.corr()
```

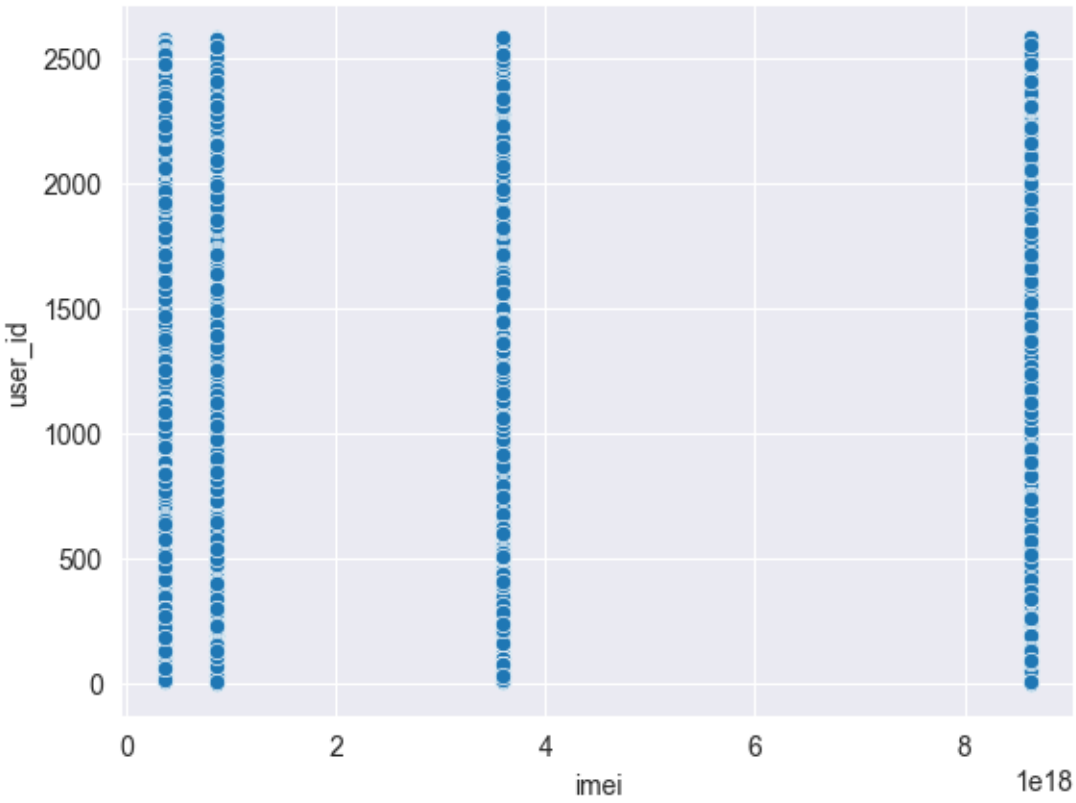
[19]:

	imei	user_id
imei	1.000000	0.009667
user_id	0.009667	1.000000

Bivariate analysis = Pair analysis. To see correlation between two variables/attributes

[20]:
```
sns.scatterplot(data=profiles, x='imei', y='user_id')
```

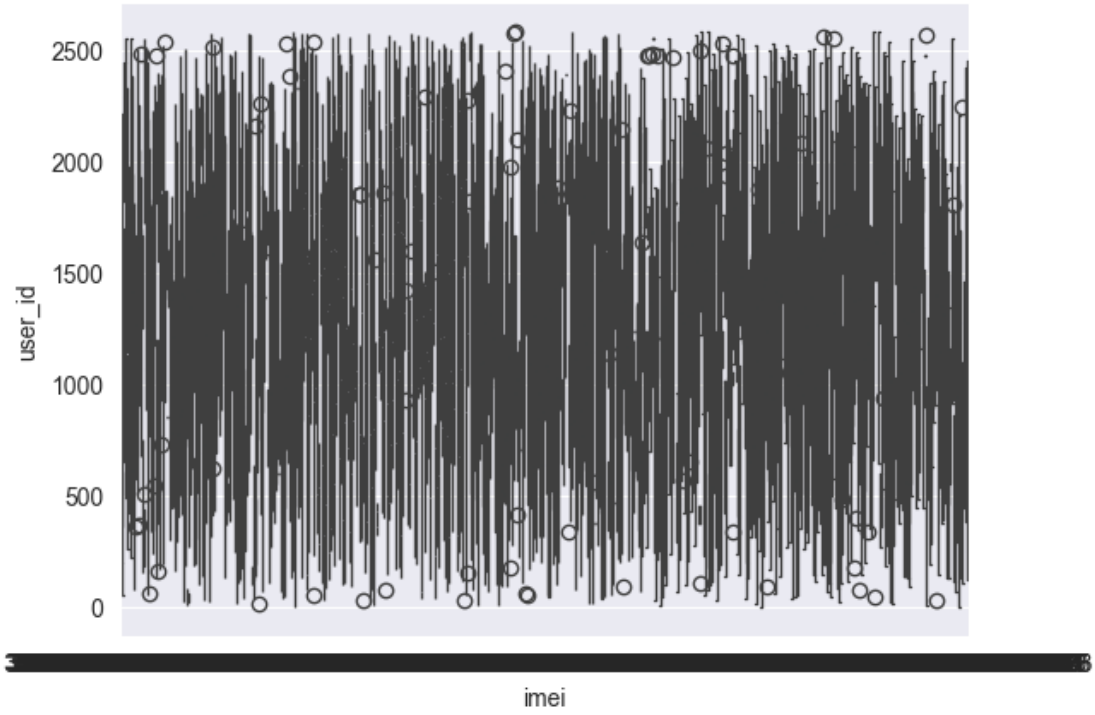
[20]:
```
<Axes: xlabel='imei', ylabel='user_id'>
```



Compare the distribution between columns imei and user_id:

[21]:
```
sns.boxplot(x='imei', y='user_id', data=profiles)
```

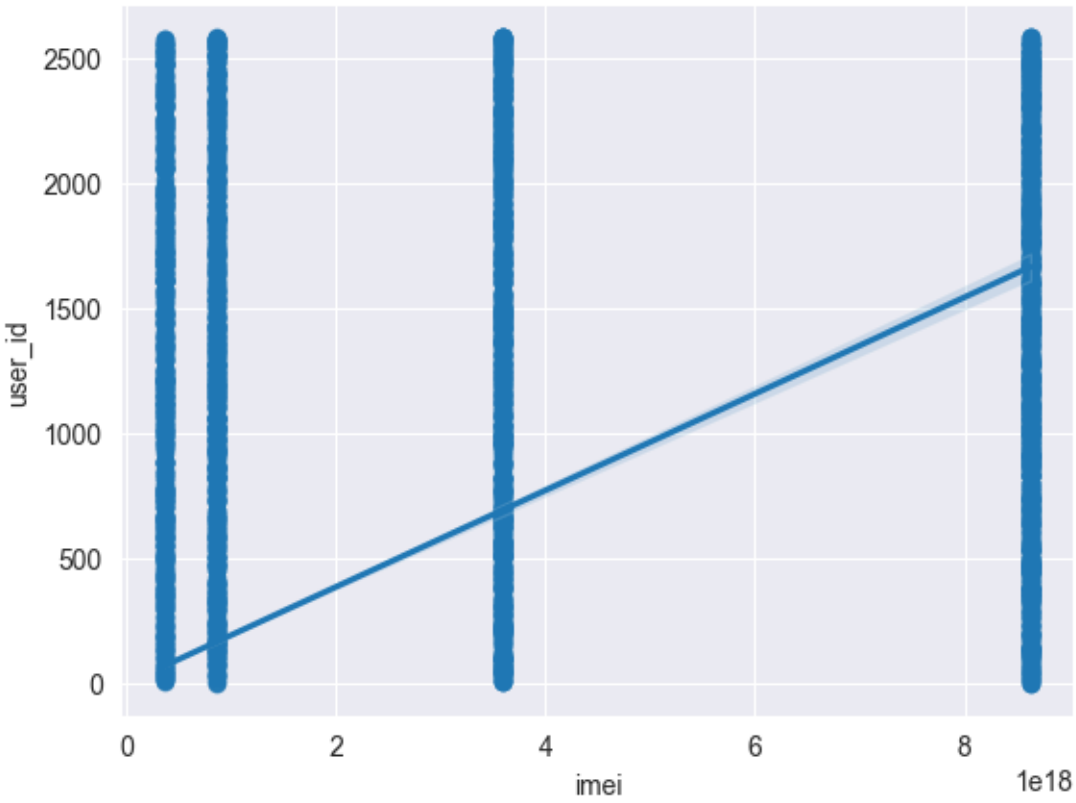
[21]:
```
<Axes: xlabel='imei', ylabel='user_id'>
```



Correlation - how strong linear relationship is of the two values

[22]:
```
sns.regplot(x="imei", y="user_id", data=profiles)
print("Pearson correlation: %.3f" % profiles['imei'].corr(profiles['user_id']))
```

```
Pearson correlation: 0.010
```



Correlation it the table, summary:

[23]:
```
profiles.corr(numeric_only=True)
```

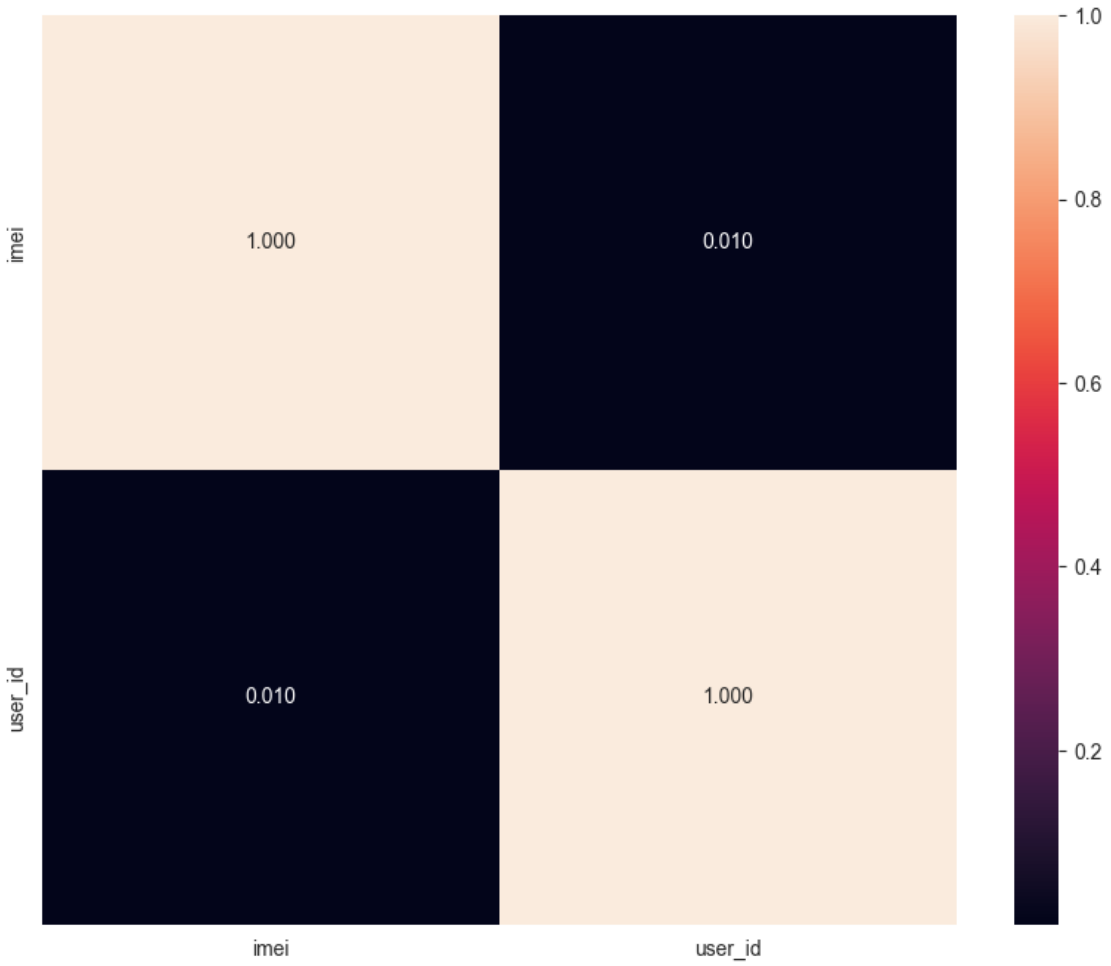
[23]:

	imei	user_id
imei	1.000000	0.009667
user_id	0.009667	1.000000

Show correlations through heatmap to visualize numerical values between columns:

[24]:
```
fig, ax = plt.subplots(figsize=(10,8))
sns.heatmap(profiles.corr(numeric_only=True), ax=ax, annot=True, fmt=".3f")
```

[24]: <Axes: >



2.2.1 1.2 A Problem identification

Duplicates:

[25]:
```
profiles.duplicated().sum()

# zero... damn
```

[25]:
```
np.int64(0)
```

Missing values:

[26]:
```
profiles.isna().sum()
```

[26]:
```
username           0
ssn                0
```

```
mail              0
residence      1682
birthdate      1164
imei              0
user_id           0
registration      0
job            1811
company           0
address         388
name              0
dtype: int64
```

We can see that there are large quantities of missing values in column residence, birthdate and job. Dropping them would shrink the dataset significantly. We can fill them with some default values.

[27]:
```
profiles.fillna({'job': 'Unknown'}, inplace=True)
profiles.fillna({'residence': 'Unknown'}, inplace=True)
profiles.fillna({'birthdate': 'Unknown'}, inplace=True)

profiles.isna().sum()
```

[27]:
```
username          0
ssn               0
mail              0
residence         0
birthdate         0
imei              0
user_id           0
registration      0
job               0
company           0
address         388
name              0
dtype: int64
```