Project_IAU

October 20, 2025

1 Import libraries

[195]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
```

[196]:
```
#warnings fix
warnings.filterwarnings("ignore", category=UserWarning, module="IPython")
plt.rcParams['font.family'] = 'DejaVu Sans'  # Change the font globally
plt.tight_layout()  # Ensure layout adjustments
```

```
<Figure size 640x480 with 0 Axes>
```

2 Load datasets

[197]:
```
patient_df = pd.read_csv('132/patient.csv', sep = '\t')
station_df = pd.read_csv('132/station.csv', sep= '\t')
observation_df = pd.read_csv('132/observation.csv', sep = '\t')
```

[198]:
```
patient_df.head()
```

[198]:
```
                         company  \
0
1  Gray, Cunningham and Morales
2                        Walter
3
4                 Munari s.r.l.

                                   current_location              ssn  \
0      (Decimal('32.168477'), Decimal('9.804478'))      302-73-9054
1     (Decimal('-80.289857'), Decimal('2.308813'))      499-92-6793
2   (Decimal('63.5169555'), Decimal('-48.876252'))      925-81-9055
3   (Decimal('-71.015803'), Decimal('140.978474'))      175-19-6965
4  (Decimal('-58.1525245'), Decimal('-120.099037'))  FRSMRL26C50L167Z
```

```
                  name                   mail  user_id  residence  \
0                       takuma47@gmail.com     1770        NaN
1  Nicholas Campbell     curtis06@yahoo.com      946        NaN
2        Raissa Rose   junkgisbert@yahoo.de     2010        NaN
3                     morimomoko@gmail.com     1100        NaN
4  Eraldo Anguillara   bpergolesi@poste.it     1247        NaN

              username   birthdate  \
0            snakajima  1937-03-07
1           benjamin02  1993-05-15
2             jkoehler         NaN
3        yamaguchikana  2000-08-20
4  guglielmomicheletti         NaN

                                          address          registration  \
0                                     27 2 7            2020-08-15
1  0515 Angela Run\r\nPort Thomasberg, GU 35535            2018-11-30
2     Louise-Stey-Platz 79\r\n88420 Bremervörde  05/09/2025, 00:00:00
3                    26 5 4     985  10/21/2022, 00:00:00
4                                             NaN            2024-11-26

  blood_group  station_ID
0          B-         464
1          A-         109
2          O-         462
3          B-         191
4          B+         588
```

[199]:
```
station_df.head()
```

[199]:
```
           QoS code  latitude              revision  longitude      station
0         good   JP  36.00000            2020-11-25  139.55722      Okegawa
1    excellent   IN  11.93381  05/24/2021, 00:00:00   79.82979   Puducherry
2  maintenance   DE  52.21099            2022-05-10    7.02238       Gronau
3    excellent   CN  41.09822            2018-01-23  120.74792      Nanpiao
4         good   US  33.54428           08 Jun 2024  -84.23381  Stockbridge
```

[200]:
```
observation_df.head()
```

[200]:
```
         SpO          HR          PI          RR        EtCO         FiO  \
0  97.538229  87.194745  11.225419  14.812012  42.113735  33.852538
1  97.933271  80.787303  11.730935  14.964972  39.537692  65.326035
2  98.209983  79.733895  12.839449  14.840668  39.758706  53.925230
3  98.202790  86.156903  11.204152  14.523288  43.448577  35.227704
4  97.951933  78.966258  13.691758  14.992054  39.722280  35.005559

         PRV         BP  Skin Temperature  Motion/Activity index  …  \
```

```
0  144.504405  100.455727         35.961920             10.302567  …
1  110.615787  102.133386         36.274352              8.975704  …
2  107.208040  104.036654         35.583851              7.653790  …
3  143.282224  105.723603         36.463180              8.795732  …
4  118.524021   98.996494         35.080937              8.388887  …

         CO  Blood Flow Index  PPG waveform features  Signal Quality Index  \
0  4.022852         58.317397              49.143701             42.399816
1  4.002043         70.127865              15.557799             46.078137
2  4.001451         76.139163              53.879956             41.525607
3  4.015162         49.461570              58.701159             36.535021
4  4.001110         47.065823              52.338305             29.506008

   Respiratory effort  O  extraction ratio        SNR  oximetry   latitude  \
0           46.497869                0.289012  39.334620       1.0  49.183239
1           53.351208                0.290879  26.006709       0.0  33.544280
2           52.124182                0.263171  31.890829       1.0 -27.505780
3           50.342830                0.256780  30.721375       1.0  37.656390
4           39.480811                0.276094  38.214856       0.0  51.202190

    longitude
0   15.454273
1  -84.233810
2  153.102360
3  126.835000
4    7.360270

[5 rows x 23 columns]
```

lets have a look at the attributes

[201]:
```
observation_df.columns
```

[201]:
```
Index(['SpO ', 'HR', 'PI', 'RR', 'EtCO ', 'FiO ', 'PRV', 'BP',
       'Skin Temperature', 'Motion/Activity index', 'PVI', 'Hb level', 'SV',
       'CO', 'Blood Flow Index', 'PPG waveform features',
       'Signal Quality Index', 'Respiratory effort', 'O  extraction ratio',
       'SNR', 'oximetry', 'latitude', 'longitude'],
      dtype='object')
```

[202]:
```
station_df.columns
```

[202]:
```
Index(['QoS', 'code', 'latitude', 'revision', 'longitude', 'station'],
dtype='object')
```

[203]:
```
patient_df.columns
```

[203]: Index(['company', 'current_location', 'ssn', 'name', 'mail', 'user_id',
       'residence', 'username', 'birthdate', 'address', 'registration',
       'blood_group', 'station_ID'],
      dtype='object')

[204]:
```
for column in observation_df.columns:
    if column in station_df.columns:
        print ('observation_df and station_df share column :' + str(column) )
    if column in patient_df.columns:
        print ('observation_df and patient_df share column :' + str(column) )

for column in patient_df.columns:
    if column in station_df.columns:
        print ('station_df and patient_df share column :' + str(column) )
```

```
observation_df and station_df share column :latitude
observation_df and station_df share column :longitude
```

we can see that observation and station dataframes might be joinable via coordinates but after manual revision we can see that patient.station_ID will probably map to station ids

before joining the datasets, lets perform some basic EDA

3 1.1 Základný opis dát spolu s ich charakteristikami

3.1 A) Analyza struktur dat ako subory + zaznamy

station_df

[205]:
```
station_df.head()
```

[205]:

	QoS	code	latitude	revision	longitude	station
0	good	JP	36.00000	2020-11-25	139.55722	Okegawa
1	excellent	IN	11.93381	05/24/2021, 00:00:00	79.82979	Puducherry
2	maintenance	DE	52.21099	2022-05-10	7.02238	Gronau
3	excellent	CN	41.09822	2018-01-23	120.74792	Nanpiao
4	good	US	33.54428	08 Jun 2024	-84.23381	Stockbridge

[206]:
```
station_df.shape
```

[206]: (703, 6)

[207]:
```
station_df.columns
```

[207]: Index(['QoS', 'code', 'latitude', 'revision', 'longitude', 'station'],
dtype='object')

[208]:
```
station_df.dtypes
```

[208]:
```
QoS           object
code          object
latitude     float64
revision      object
longitude    float64
station       object
dtype: object
```

[209]:
```
#2 nas, not bad
station_df.isna().sum()
```

[209]:
```
QoS          0
code         2
latitude     0
revision     0
longitude    0
station      0
dtype: int64
```

[210]:
```
station_df[station_df['code'].isna()]
```

[210]:
```
             QoS code  latitude              revision  longitude    station
274  maintenance  NaN -21.98333  02/23/2016, 00:00:00   16.91667  Okahandja
318         good  NaN -21.98333           06 Dec 2021   16.91667  Okahandja
```

[211]:
```
#mby there is valid code for Okahandja?
station_df[station_df['station'] == 'Okahandja']
#there is not :(
```

[211]:
```
             QoS code  latitude              revision  longitude    station
274  maintenance  NaN -21.98333  02/23/2016, 00:00:00   16.91667  Okahandja
318         good  NaN -21.98333           06 Dec 2021   16.91667  Okahandja
```

[212]:
```
station_df.nunique()
```

[212]:
```
QoS            4
code          98
latitude     498
revision     683
longitude    497
station      498
dtype: int64
```

[213]:
```
#prob useless
station_df.describe()
```

[213]:
```
         latitude   longitude
count  703.000000  703.000000
```

```
mean    28.699220   16.946846
std     24.406067   70.122555
min    -44.396720 -156.474320
25%     14.354040  -14.410810
50%     36.650000   13.321270
75%     47.432685   71.552920
max     65.848110  171.253640
```

patient df

[214]:
```
patient_df.head()
```

[214]:
```
                         company  \
0
1  Gray, Cunningham and Morales
2                         Walter
3
4                  Munari s.r.l.

                                     current_location            ssn  \
0       (Decimal('32.168477'), Decimal('9.804478'))       302-73-9054
1      (Decimal('-80.289857'), Decimal('2.308813'))       499-92-6793
2    (Decimal('63.5169555'), Decimal('-48.876252'))       925-81-9055
3    (Decimal('-71.015803'), Decimal('140.978474'))       175-19-6965
4  (Decimal('-58.1525245'), Decimal('-120.099037'))  FRSMRL26C50L167Z

                name                  mail  user_id  residence  \
0                     takuma47@gmail.com     1770        NaN
1  Nicholas Campbell     curtis06@yahoo.com      946        NaN
2        Raissa Rose   junkgisbert@yahoo.de     2010        NaN
3                     morimomoko@gmail.com     1100        NaN
4  Eraldo Anguillara   bpergolesi@poste.it     1247        NaN

              username   birthdate  \
0            snakajima  1937-03-07
1           benjamin02  1993-05-15
2             jkoehler         NaN
3        yamaguchikana  2000-08-20
4  guglielmomicheletti         NaN

                                     address          registration  \
0                                  27 2 7            2020-08-15
1  0515 Angela Run\r\nPort Thomasberg, GU 35535            2018-11-30
2     Louise-Stey-Platz 79\r\n88420 Bremervörde  05/09/2025, 00:00:00
3                        26 5 4     985  10/21/2022, 00:00:00
4                                          NaN            2024-11-26

  blood_group  station_ID
```

```
0          B-         464
1          A-         109
2          O-         462
3          B-         191
4          B+         588
```

[215]:
```
patient_df.shape
```

[215]:
```
(2197, 13)
```

[216]:
```
patient_df.columns
```

[216]:
```
Index(['company', 'current_location', 'ssn', 'name', 'mail', 'user_id',
       'residence', 'username', 'birthdate', 'address', 'registration',
       'blood_group', 'station_ID'],
      dtype='object')
```

[217]:
```
patient_df.dtypes
```

[217]:
```
company              object
current_location     object
ssn                  object
name                 object
mail                 object
user_id               int64
residence           float64
username             object
birthdate            object
address              object
registration         object
blood_group          object
station_ID            int64
dtype: object
```

[218]:
```
patient_df.isna().sum()
#residence is full NaN so it will be dropped, birthdate and address probably␣
 ↪too,
#current location needs to be processed
```

[218]:
```
company                0
current_location     110
ssn                    0
name                   0
mail                   0
user_id                0
residence           2197
username               0
birthdate            989
```

```
address               330
registration            0
blood_group             0
station_ID              0
dtype: int64
```

[219]:
```
patient_df.nunique()
#many unique values
```

[219]:
```
company             1989
current_location    2087
ssn                 2197
name                2131
mail                2192
user_id             1407
residence              0
username            2175
birthdate           1189
address             1867
registration        1987
blood_group            8
station_ID           677
dtype: int64
```

[220]:
```
patient_df.describe()
#also useless
```

[220]:

	user_id	residence	station_ID
count	2197.000000	0.0	2197.000000
mean	1105.308603	NaN	348.892126
std	644.728055	NaN	205.367454
min	0.000000	NaN	0.000000
25%	547.000000	NaN	172.000000
50%	1118.000000	NaN	340.000000
75%	1668.000000	NaN	536.000000
max	2196.000000	NaN	702.000000

observation df

[221]:
```
observation_df.head()
```

[221]:

	SpO	HR	PI	RR	EtCO	FiO	\
0	97.538229	87.194745	11.225419	14.812012	42.113735	33.852538	
1	97.933271	80.787303	11.730935	14.964972	39.537692	65.326035	
2	98.209983	79.733895	12.839449	14.840668	39.758706	53.925230	
3	98.202790	86.156903	11.204152	14.523288	43.448577	35.227704	
4	97.951933	78.966258	13.691758	14.992054	39.722280	35.005559	

```
          PRV          BP  Skin Temperature  Motion/Activity index  …  \
0  144.504405  100.455727         35.961920              10.302567  …
1  110.615787  102.133386         36.274352               8.975704  …
2  107.208040  104.036654         35.583851               7.653790  …
3  143.282224  105.723603         36.463180               8.795732  …
4  118.524021   98.996494         35.080937               8.388887  …

         CO  Blood Flow Index  PPG waveform features  Signal Quality Index  \
0  4.022852         58.317397              49.143701             42.399816
1  4.002043         70.127865              15.557799             46.078137
2  4.001451         76.139163              53.879956             41.525607
3  4.015162         49.461570              58.701159             36.535021
4  4.001110         47.065823              52.338305             29.506008

   Respiratory effort  O  extraction ratio        SNR  oximetry   latitude  \
0           46.497869             0.289012  39.334620       1.0  49.183239
1           53.351208             0.290879  26.006709       0.0  33.544280
2           52.124182             0.263171  31.890829       1.0 -27.505780
3           50.342830             0.256780  30.721375       1.0  37.656390
4           39.480811             0.276094  38.214856       0.0  51.202190

    longitude
0   15.454273
1  -84.233810
2  153.102360
3  126.835000
4    7.360270

[5 rows x 23 columns]
```

[222]:
```python
observation_df.shape
```

[222]:
```
(12177, 23)
```

[223]:
```python
observation_df.columns
```

[223]:
```
Index(['SpO ', 'HR', 'PI', 'RR', 'EtCO ', 'FiO ', 'PRV', 'BP',
       'Skin Temperature', 'Motion/Activity index', 'PVI', 'Hb level', 'SV',
       'CO', 'Blood Flow Index', 'PPG waveform features',
       'Signal Quality Index', 'Respiratory effort', 'O  extraction ratio',
       'SNR', 'oximetry', 'latitude', 'longitude'],
      dtype='object')
```

[224]:
```python
observation_df.dtypes
```

[224]:
```
SpO                      float64
HR                       float64
```

```
PI                        float64
RR                        float64
EtCO                     float64
FiO                      float64
PRV                       float64
BP                        float64
Skin Temperature          float64
Motion/Activity index     float64
PVI                       float64
Hb level                  float64
SV                        float64
CO                        float64
Blood Flow Index          float64
PPG waveform features     float64
Signal Quality Index      float64
Respiratory effort        float64
O  extraction ratio      float64
SNR                       float64
oximetry                  float64
latitude                  float64
longitude                 float64
dtype: object
```

[225]:
```
observation_df.isna().sum()
#great
```

[225]:
```
SpO                      0
HR                        0
PI                        0
RR                        0
EtCO                     0
FiO                      0
PRV                       0
BP                        0
Skin Temperature          0
Motion/Activity index     0
PVI                       0
Hb level                  0
SV                        0
CO                        0
Blood Flow Index          0
PPG waveform features     0
Signal Quality Index      0
Respiratory effort        0
O  extraction ratio      0
SNR                       0
oximetry                  0
```

```
latitude                  0
longitude                 0
dtype: int64
```

[226]:
```
observation_df.nunique()
#makes sense, only oximetry, latitude and longitude are not completely unique
```

[226]:
```
SpO                      11997
HR                       11997
PI                       11997
RR                       11997
EtCO                     11997
FiO                      11997
PRV                      11997
BP                       11997
Skin Temperature         11997
Motion/Activity index    11997
PVI                      11997
Hb level                 11997
SV                       11997
CO                       11997
Blood Flow Index         11997
PPG waveform features    11997
Signal Quality Index     11997
Respiratory effort       11997
O  extraction ratio      11997
SNR                      11997
oximetry                     2
latitude                   498
longitude                  497
dtype: int64
```

[227]:
```
observation_df.describe()
```

[227]:
```
                SpO            HR            PI            RR          EtCO  \
count  12177.000000  12177.000000  12177.000000  12177.000000  12177.000000
mean      97.336001     83.397242     10.360642     16.158948     40.235152
std        0.657577      7.609815      2.417855      1.398210      1.715679
min       95.000000     60.000000      0.200000     12.000000     35.000000
25%       96.895657     77.573676      8.800161     15.044708     38.749846
50%       97.332851     84.323757     10.365288     15.979021     40.531056
75%       97.776356     89.763115     11.893853     17.393451     41.590048
max      100.000000    100.000000     20.000000     20.000000     45.000000

                FiO           PRV            BP  Skin Temperature  \
count  12177.000000  12177.000000  12177.000000      12177.000000
mean      58.821759    117.675964    104.591413         35.426048
```

```
std      12.119443    21.841513    4.088282          0.619283
min      21.000000    20.000000   90.000000         33.000000
25%      49.328258   103.101631  101.857377         35.007462
50%      59.402365   117.695370  104.604980         35.424762
75%      68.437184   132.088845  107.321633         35.845164
max     100.000000   200.000000  120.000000         38.000000

       Motion/Activity index  …           CO  Blood Flow Index  \
count           12177.000000  …  12177.000000      12177.000000
mean                9.436818  …      4.078126         52.640590
std                 0.998907  …      0.222547         13.127091
min                 5.652322  …      4.000000          0.000000
25%                 8.766904  …      4.000762         43.996467
50%                 9.432476  …      4.007395         52.709829
75%                10.105665  …      4.064212         61.440013
max                13.997052  …      8.000000        100.000000

       PPG waveform features  Signal Quality Index  Respiratory effort  \
count           12177.000000          12177.000000        12177.000000
mean               46.734247             47.572094           49.788653
std                13.374194             13.487056           13.006681
min                 0.000000              0.000000            0.000000
25%                37.650597             38.396055           40.992562
50%                46.733166             47.810036           49.742898
75%                55.671672             56.736043           58.630016
max               100.000000            100.000000          100.000000

       O  extraction ratio           SNR      oximetry      latitude  \
count         12177.000000  12177.000000  12177.000000  12177.000000
mean              0.249557     29.994576      0.597602     28.701943
std               0.028947      5.765251      0.490401     24.402668
min               0.200000     20.000000      0.000000    -44.396720
25%               0.224466     24.978691      0.000000     14.082300
50%               0.249072     30.094799      1.000000     36.650000
75%               0.274733     34.961220      1.000000     47.484440
max               0.300000     40.000000      1.000000     65.848110

          longitude
count  12177.000000
mean      17.028640
std       70.059129
min     -156.474320
25%      -13.235600
50%       13.321270
75%       71.577370
max      171.253640
```

```
[8 rows x 23 columns]
```

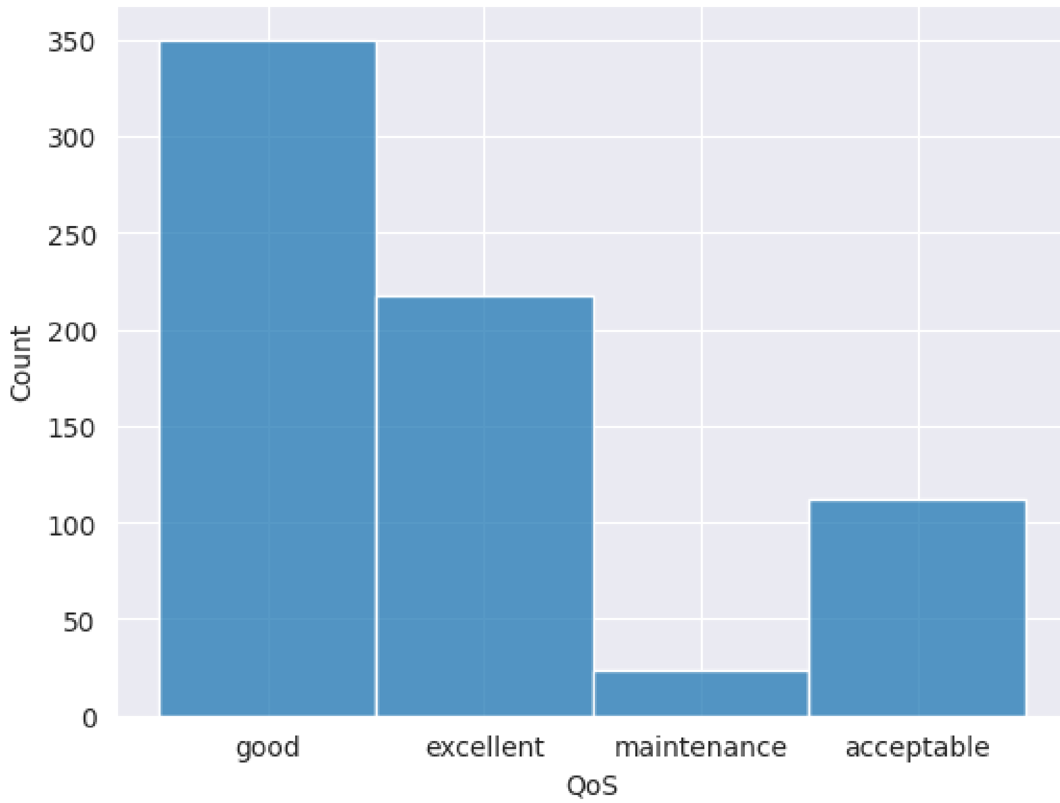
3.2 B) Analýza jednotlivých atribútov

[228]:
```
#Qos(station_df)
station_df['QoS'].unique().tolist()
```

[228]:
```
['good', 'excellent', 'maintenance', 'acceptable']
```

[229]:
```
sns.histplot(station_df['QoS'])
```

[229]:
```
<Axes: xlabel='QoS', ylabel='Count'>
```



[230]:
```
#for exact nums
station_df['QoS'].value_counts()
```

[230]:
```
QoS
good           350
excellent      217
acceptable     112
maintenance     24
```

Name: count, dtype: int64

[231]:
```python
station_df['QoS'].isna().sum()
```

[231]: np.int64(0)

[232]:
```python
#latitude (from both station_df and observation_df)
print (station_df['latitude'].min(), observation_df['latitude'].min())
print (station_df['latitude'].max(), observation_df['latitude'].max())
#the min and the max are both realistic values, from -180 to 180

set(station_df['latitude']) == set(observation_df['latitude'])
#every station is included in observation by latitude
```

```
-44.39672 -44.39672
65.84811 65.84811
```

[232]: True

[233]:
```python
#longitude (from both station_df and observation_df)
print (station_df['longitude'].min(), observation_df['longitude'].min())
print (station_df['longitude'].max(), observation_df['longitude'].max())
#the min and the max are both realistic values, from -180 to 180

set(station_df['longitude']) == set(observation_df['longitude'])
#every station is included by longitude also

#this could be checked with patient -> current location but that needs␣
↪preprocessing in further steps
```

```
-156.47432 -156.47432
171.25364 171.25364
```

[233]: True

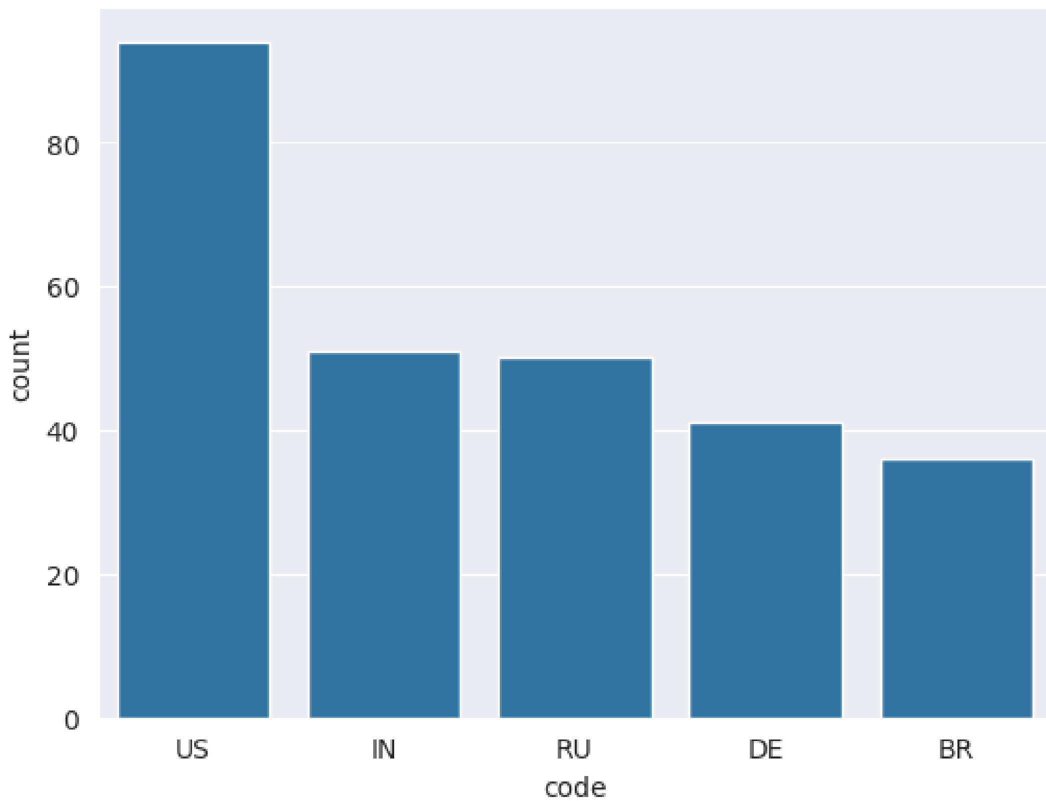
[234]:
```python
#Code (station_df)
station_df['code'].value_counts()
```

[234]:
```
code
US    94
IN    51
RU    50
DE    41
BR    36
      ..
DK     1
CU     1
GH     1
AF     1
```

```
AT      1
Name: count, Length: 98, dtype: int64
```

[235]:
```
#top 5
top5 = station_df['code'].value_counts().head(5).reset_index()
top5.columns = ['code', 'count']

sns.barplot(x='code', y='count', data=top5)
```

[235]: <Axes: xlabel='code', ylabel='count'>

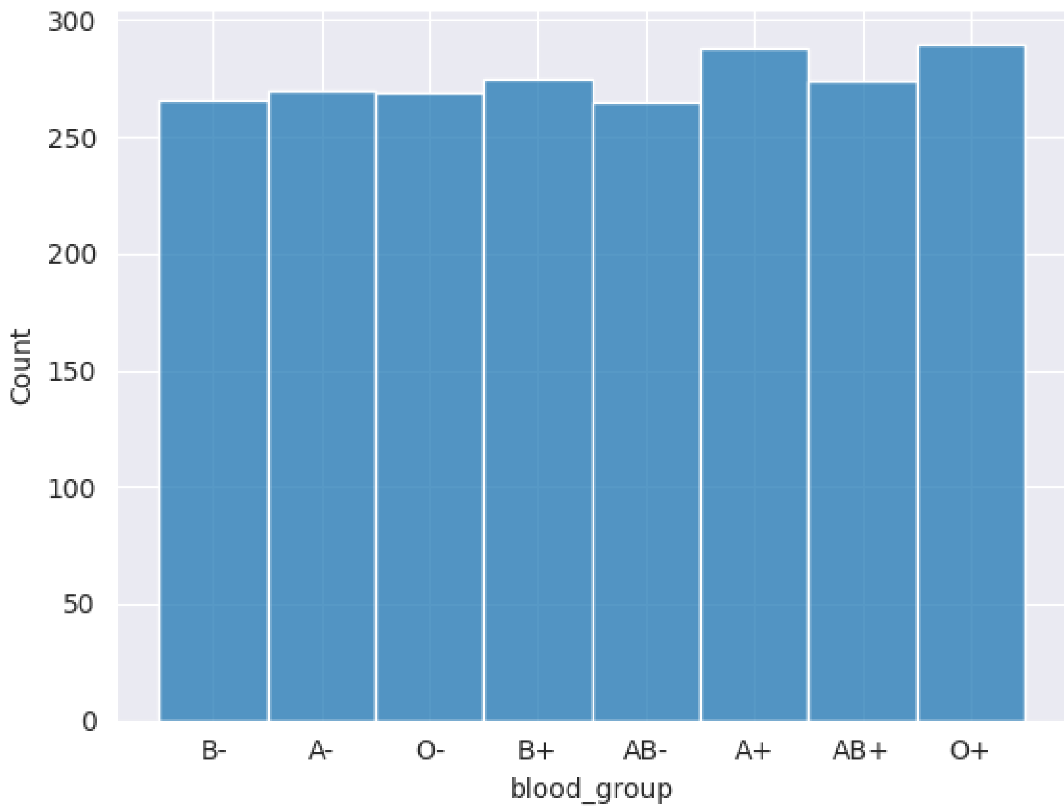


[236]:
```
#Blood group (patient_df)
patient_df['blood_group'].unique().tolist()
#All blood groups are represented
```

[236]: ['B-', 'A-', 'O-', 'B+', 'AB-', 'A+', 'AB+', 'O+']

[237]:
```
sns.histplot(patient_df['blood_group'])
#not that big of a range
```

[237]: <Axes: xlabel='blood_group', ylabel='Count'>



[238]:
```
patient_df['blood_group'].value_counts()
```

[238]:
```
blood_group
O+     290
A+     288
B+     275
AB+    274
A-     270
O-     269
B-     266
AB-    265
Name: count, dtype: int64
```

[239]:
```
patient_df['blood_group'].isna().sum()
#great
```

[239]:
```
np.int64(0)
```

[240]:
```
# SpO2
sns.boxplot(y = observation_df['SpO '])
```

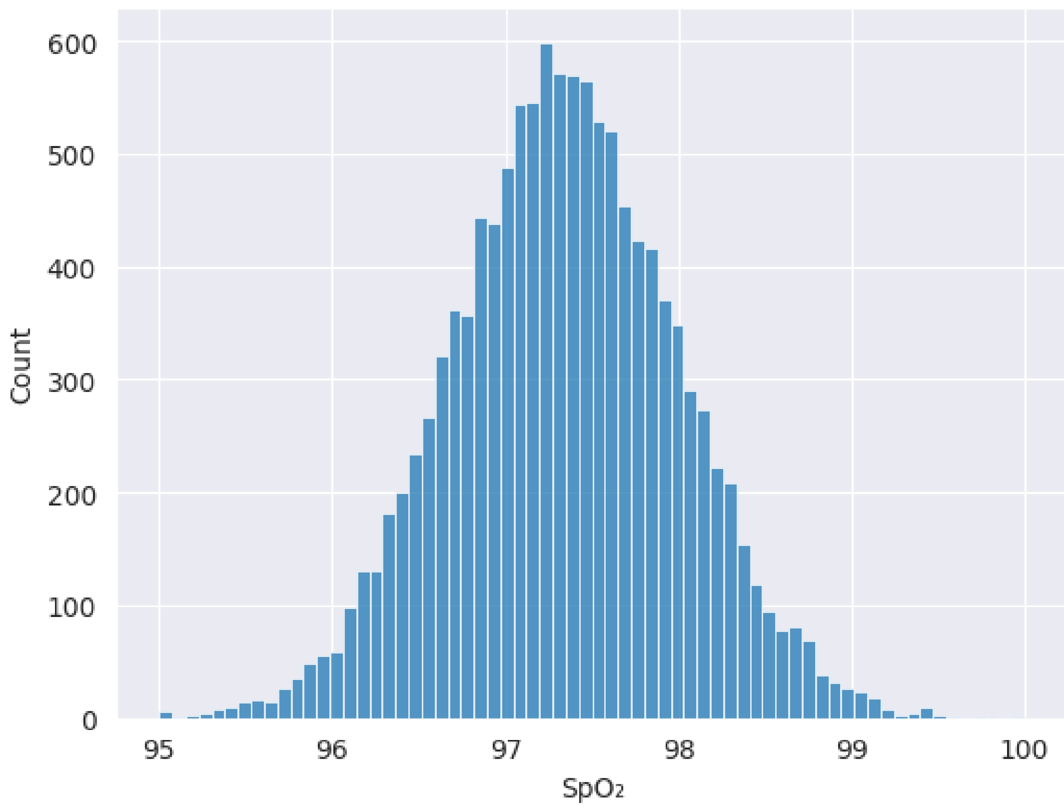
[240]: <Axes: ylabel='SpO '>

[241]:
```
#just to have the exact range
observation_df['SpO '].min(), observation_df['SpO '].max()
#realistic values
```

[241]: (np.float64(95.0), np.float64(100.0))

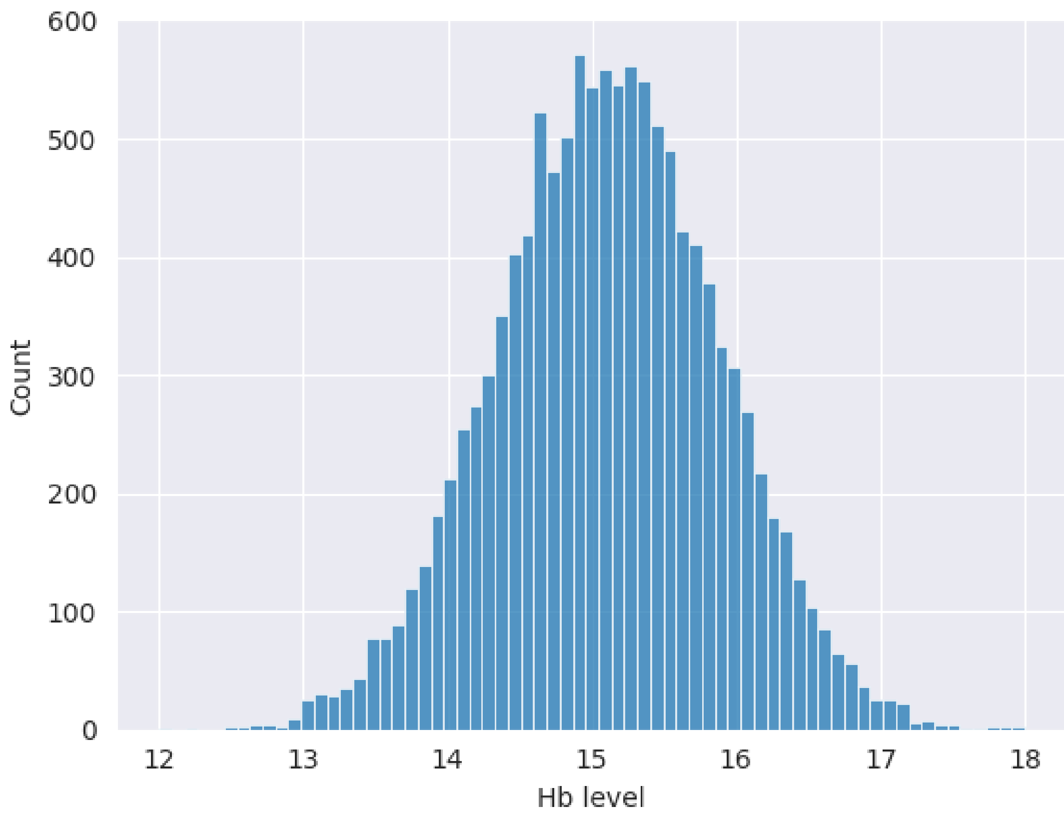
[242]:
```
sns.histplot(observation_df['SpO '])
#normal distribution
```

[242]: <Axes: xlabel='SpO ', ylabel='Count'>



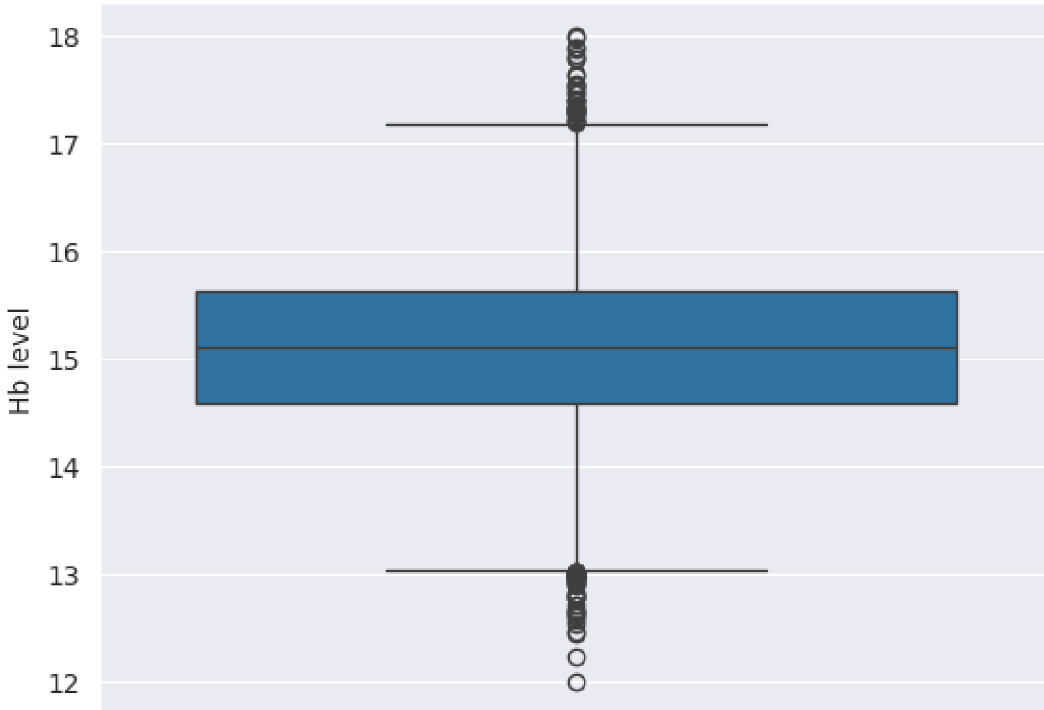
[243]:
```
#Hemoglobin (Observation_df)
sns.histplot(observation_df['Hb level'])
#pretty normal, realistic values
```

[243]: <Axes: xlabel='Hb level', ylabel='Count'>



[244]:
```
sns.boxplot(y = observation_df['Hb level'])
```

[244]: <Axes: ylabel='Hb level'>

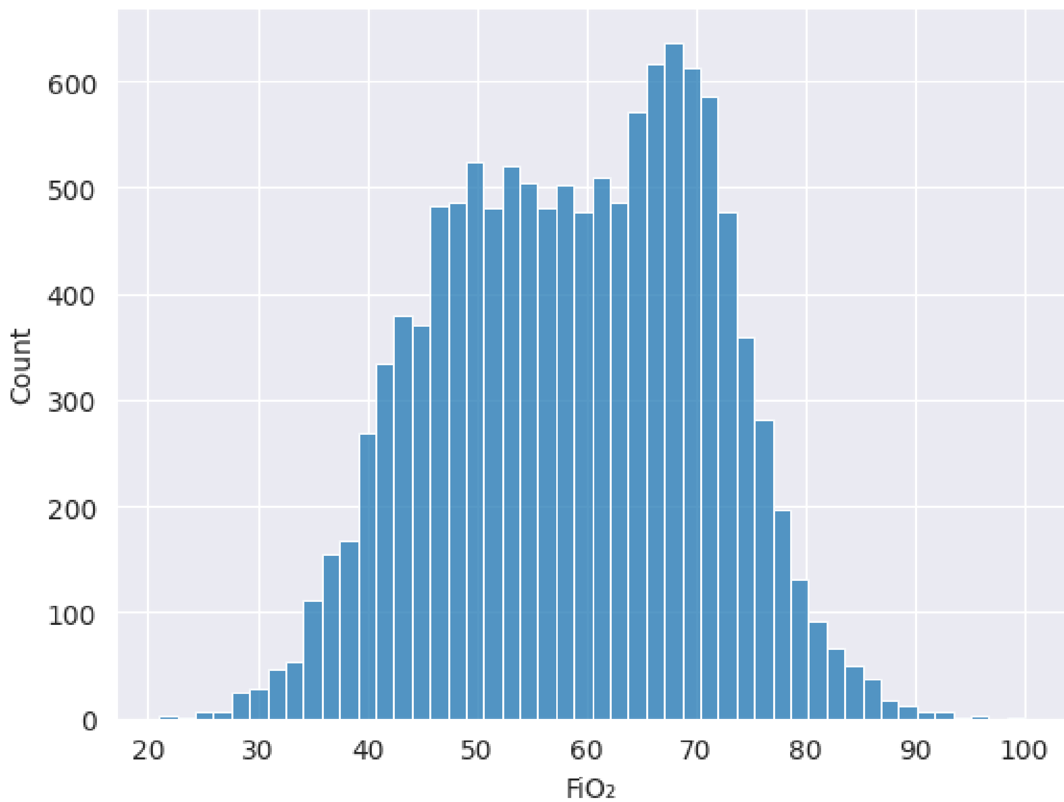


[245]:
```
#exact range
observation_df['Hb level'].min(), observation_df['Hb level'].max()
```

[245]:
```
(np.float64(12.0), np.float64(18.0))
```

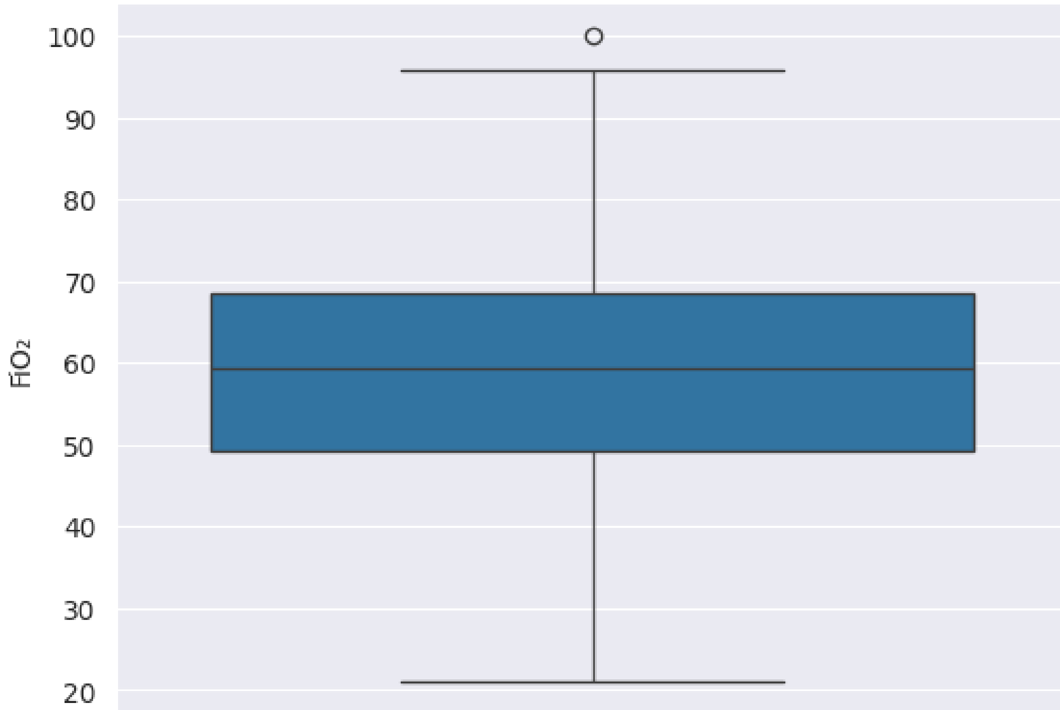
[246]:
```
#Fio (Observation_df)
sns.histplot(observation_df['FiO '])
#not that normal, slight skew to the right at the peak
#also values are not realistic for common people, probably on oxygen therapy or␣
↪something
```

[246]:
```
<Axes: xlabel='FiO ', ylabel='Count'>
```



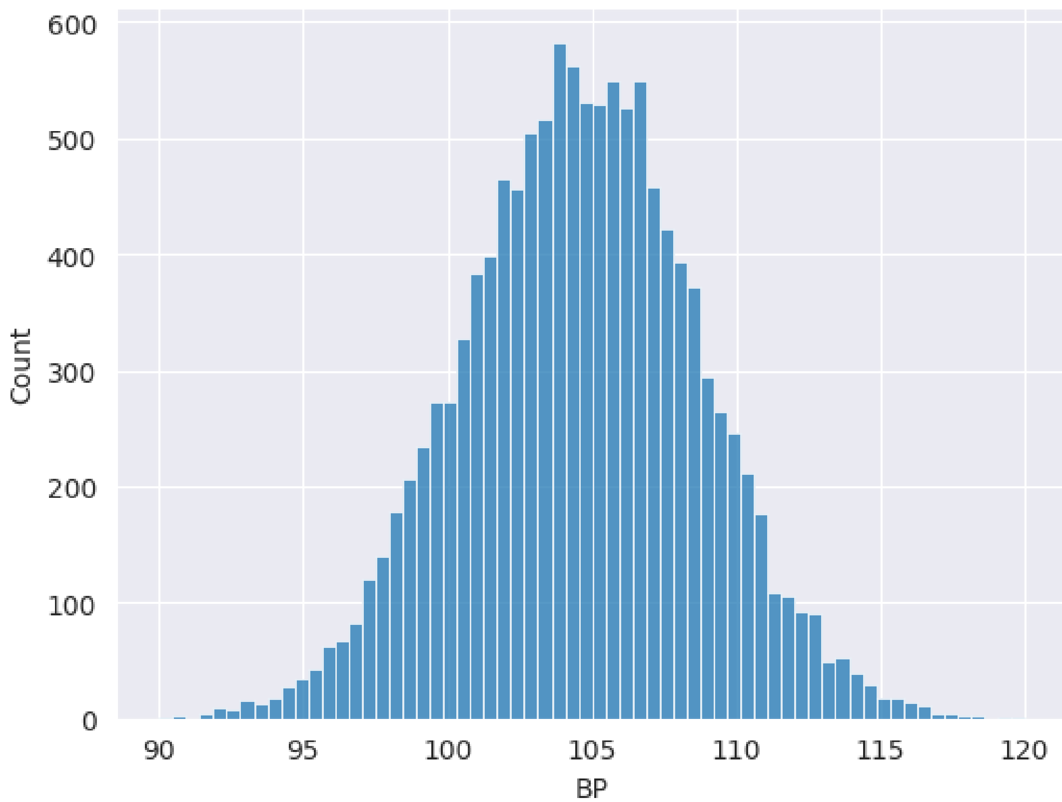

```
[247]: sns.boxplot(observation_df['FiO '])
```

```
[247]: <Axes: ylabel='FiO '>
```



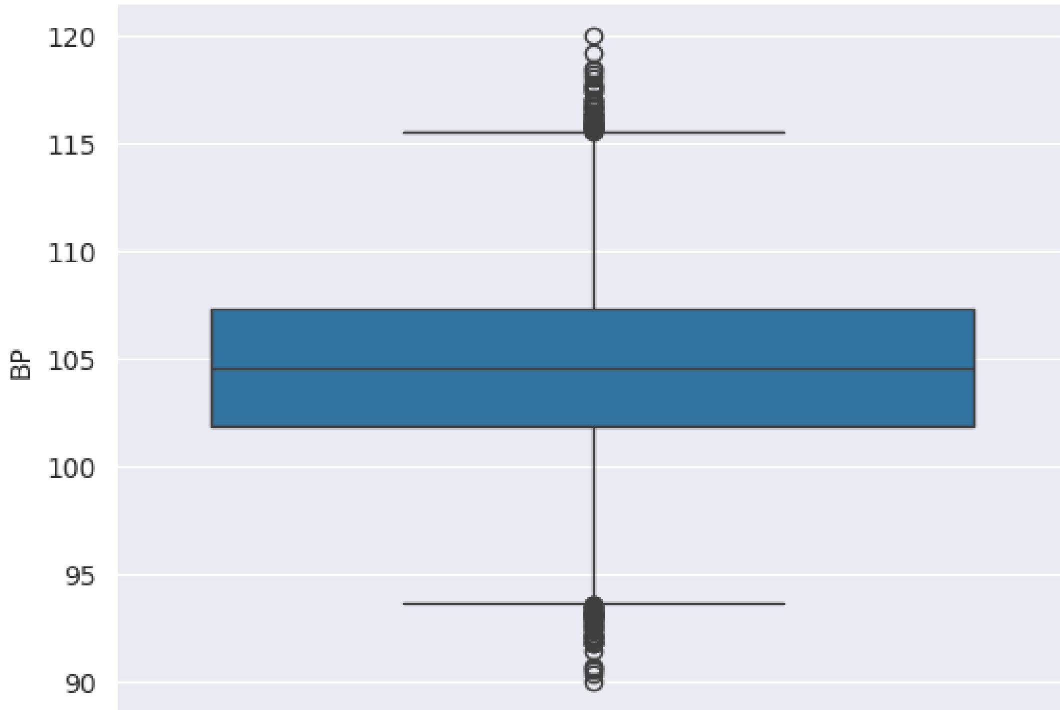
[248]:
```
#blood pressure (observation_df)
sns.histplot(observation_df['BP'])
#normal distribution with usual values
```

[248]: `<Axes: xlabel='BP', ylabel='Count'>`



[249]:
```
sns.boxplot(y = observation_df['BP'])
```

[249]: <Axes: ylabel='BP'>

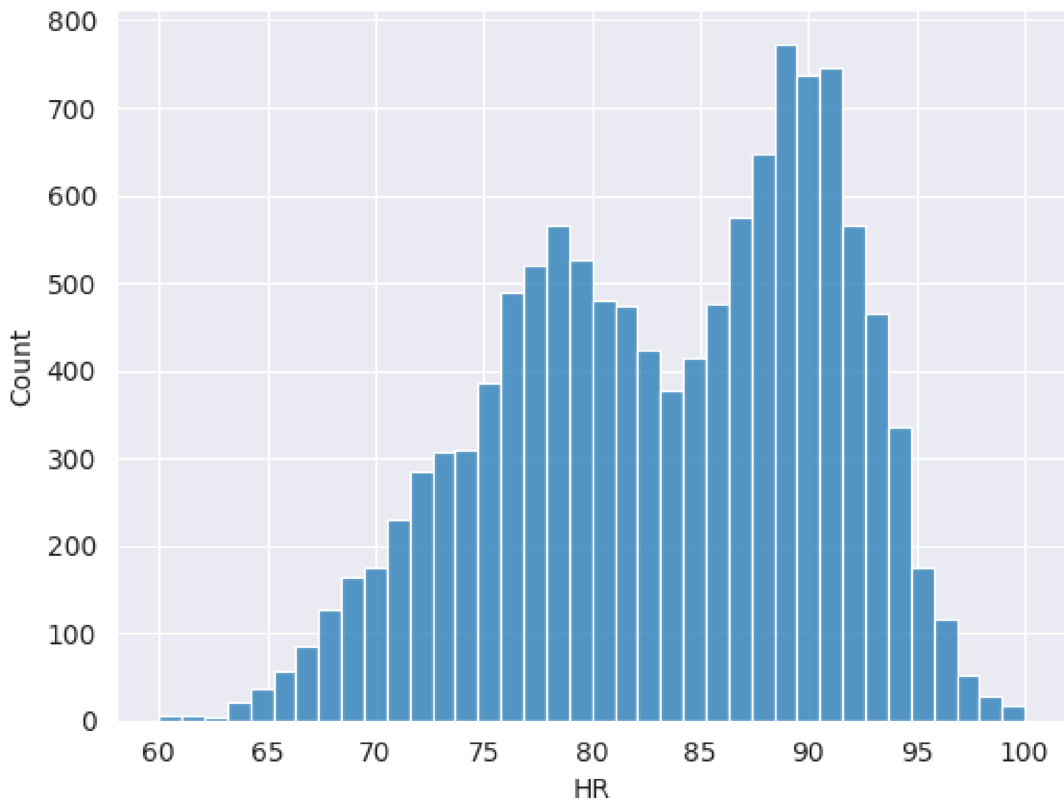



```
[250]: observation_df['BP'].min(), observation_df['BP'].max()
```

```
[250]: (np.float64(90.0), np.float64(120.0))
```

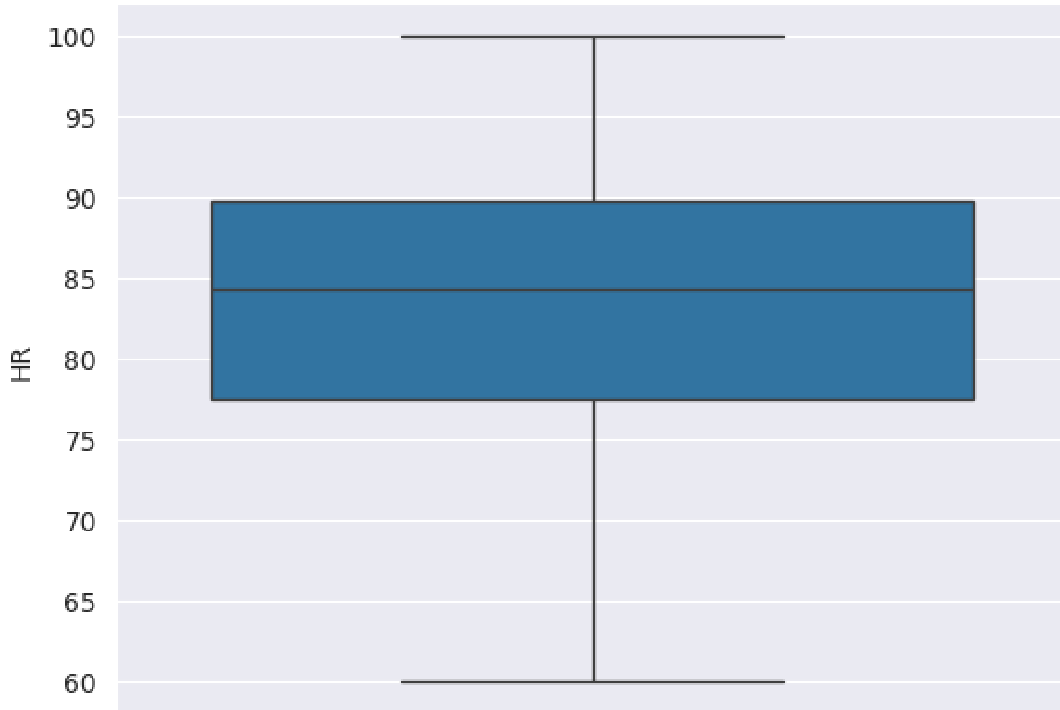
```
[251]: #Heart rate
sns.histplot(observation_df['HR'])
#pretty bimodal, but realistic values
```

```
[251]: <Axes: xlabel='HR', ylabel='Count'>
```



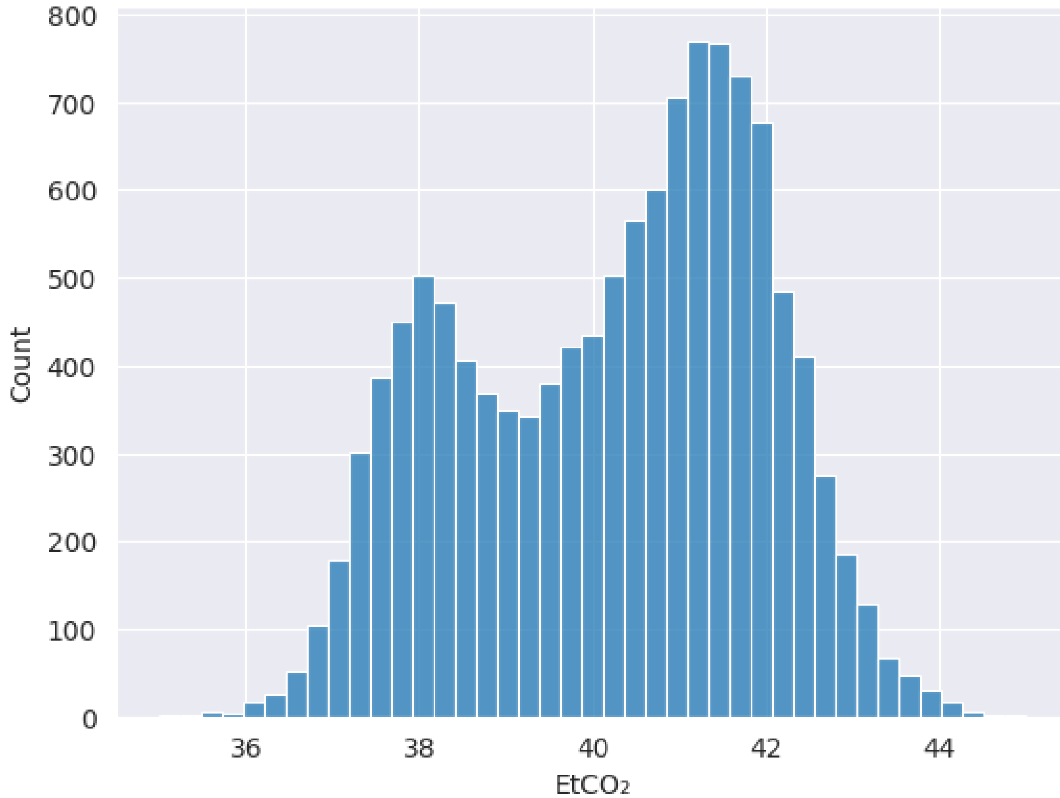
[252]:
```
sns.boxplot(y = observation_df['HR'])
```

[252]: <Axes: ylabel='HR'>



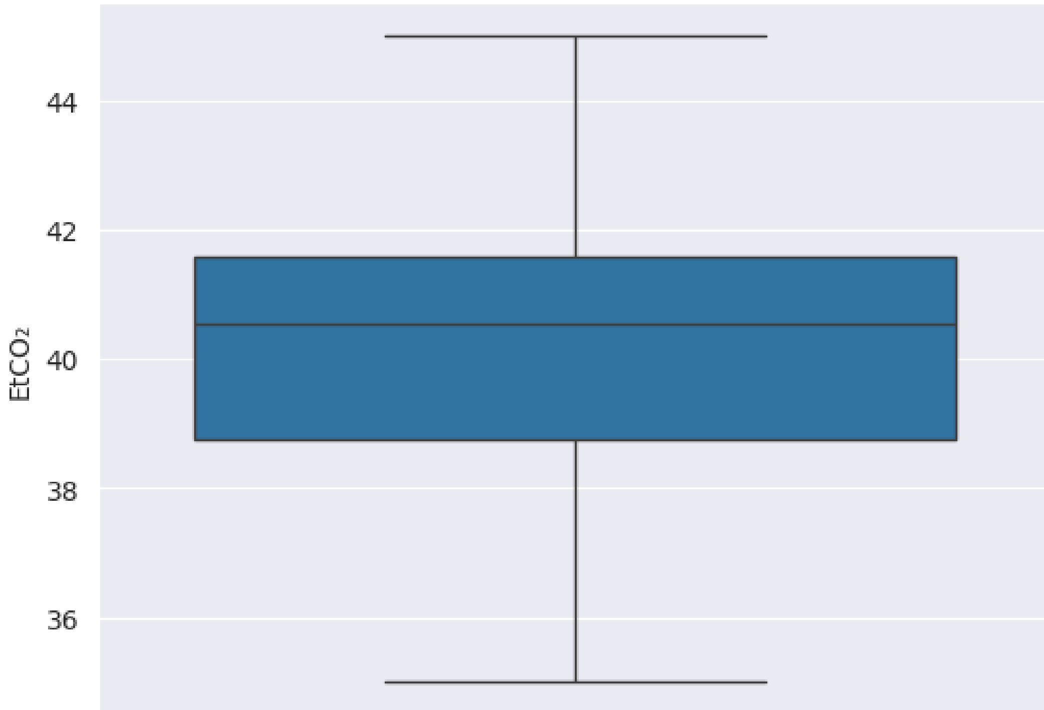

```
[253]: #ETCO2
sns.histplot(observation_df['EtCO '])
#also bimodal
```

```
[253]: <Axes: xlabel='EtCO ', ylabel='Count'>
```



[254]:
```
sns.boxplot(observation_df['EtCO '])
```

[254]: <Axes: ylabel='EtCO '>



[255]:
```
observation_df['EtCO '].min(), observation_df['EtCO '].max()
#also usual values
```

[255]: (np.float64(35.0), np.float64(45.0))

[256]:
```
#just checking if this is relevant
patient_df['company'].nunique(), patient_df.shape
#too many unique vals, irrelevant
```

[256]: (1989, (2197, 13))

3.3 C) Párová analýza dát: Identifikujte vzťahy a závislostí medzi dvojicami atribútov.

But first lets create a dataset with merged tables with only the attributes we consider as necessary

3.3.1 Creation of the joined df

[257]:
```
#This is a list of attributes that will make up the new merged dataset

#QoS might be of significance with Fio and other atts,
#some codes may also have a touch (for example US station are healthier... idk)
#latitude and longitude will be kept for potentional current location␣
↪(patient_df) relation
```

```python
station_attributes = ['QoS', 'code', 'latitude', 'longitude', ]

#current location has been mentioned in station atts, may be dropped -> later comment : it was dropped
#blood group may be relevant, may be not we will see
#user id just so we have some unique user identifier since we are dropping name and everything
patient_attributes = ['user_id', 'blood_group', 'station_ID']

#everything apart from longitude and latitude since that is already kept from station_attributes and we mentioned before that all stations are mentioned in obs_df
observation_attributes = ['SpO ', 'HR', 'PI', 'RR', 'EtCO ', 'FiO ', 'PRV', 'BP',
       'Skin Temperature', 'Motion/Activity index', 'PVI', 'Hb level', 'SV',
       'CO', 'Blood Flow Index', 'PPG waveform features',
       'Signal Quality Index', 'Respiratory effort', 'O  extraction ratio',
       'SNR', 'oximetry']

#for check, lets count the atts
len(station_attributes) + len(patient_attributes) + len(observation_attributes)
```

[257]: 28

[258]:
```python
station_df2 = station_df[station_attributes].copy()
patient_df2 = patient_df[patient_attributes].copy()
observation_df2 = observation_df[observation_attributes + ['latitude', 'longitude']].copy()
#this is needed to convert the ID into a normal attribute
station_df2 = station_df2.reset_index().rename(columns={'index': 'station_ID'})
```

[259]:
```python
observation_df.shape
```

[259]: (12177, 23)

[260]:
```python
df_obs_stat = observation_df2.merge(
    station_df2,
    left_on=['latitude', 'longitude'],   # from merged patiens
    right_on=['latitude', 'longitude'],  # from observation_df
    how='inner'
)
```

[261]:
```python
#the shape should be the same as observation_df.shape
df_obs_stat.shape
#it is not because in the following cell we can see that some stations have the same coordinates
```

[261]: (21385, 26)

[262]:
```
station_df2[['latitude', 'longitude']].duplicated().sum()
```

[262]:
```
np.int64(205)
```

[263]:
```
df_obs_stat.head(5)
```

[263]:
```
         SpO         HR         PI         RR       EtCO        FiO  \
0  97.538229  87.194745  11.225419  14.812012  42.113735  33.852538
1  97.933271  80.787303  11.730935  14.964972  39.537692  65.326035
2  97.933271  80.787303  11.730935  14.964972  39.537692  65.326035
3  98.209983  79.733895  12.839449  14.840668  39.758706  53.925230
4  98.202790  86.156903  11.204152  14.523288  43.448577  35.227704

          PRV          BP  Skin Temperature  Motion/Activity index  …  \
0  144.504405  100.455727         35.961920              10.302567  …
1  110.615787  102.133386         36.274352               8.975704  …
2  110.615787  102.133386         36.274352               8.975704  …
3  107.208040  104.036654         35.583851               7.653790  …
4  143.282224  105.723603         36.463180               8.795732  …

   Signal Quality Index  Respiratory effort  O extraction ratio        SNR  \
0             42.399816           46.497869            0.289012  39.334620
1             46.078137           53.351208            0.290879  26.006709
2             46.078137           53.351208            0.290879  26.006709
3             41.525607           52.124182            0.263171  31.890829
4             36.535021           50.342830            0.256780  30.721375

   oximetry   latitude   longitude  station_ID        QoS  code
0       1.0  49.183239   15.454273         403       good    CZ
1       0.0  33.544280  -84.233810           4       good    US
2       0.0  33.544280  -84.233810         426  excellent    US
3       1.0 -27.505780  153.102360         219  excellent    AU
4       1.0  37.656390  126.835000         319  excellent    KR

[5 rows x 26 columns]
```

[264]:
```
#now we need to join the df_obs_stat with patient_df
df = df_obs_stat.merge(
    patient_df2,
    left_on=['station_ID'],
    right_on=['station_ID'],
    how='inner'
)
```

[265]:
```
df.shape
#now we have many more entries since there are many patients sharing the same␣
↪station as we can see in the next cell
```

[265]: (66973, 28)

[305]:
```
patient_df2[['station_ID']].duplicated().sum()
```

[305]: np.int64(1520)

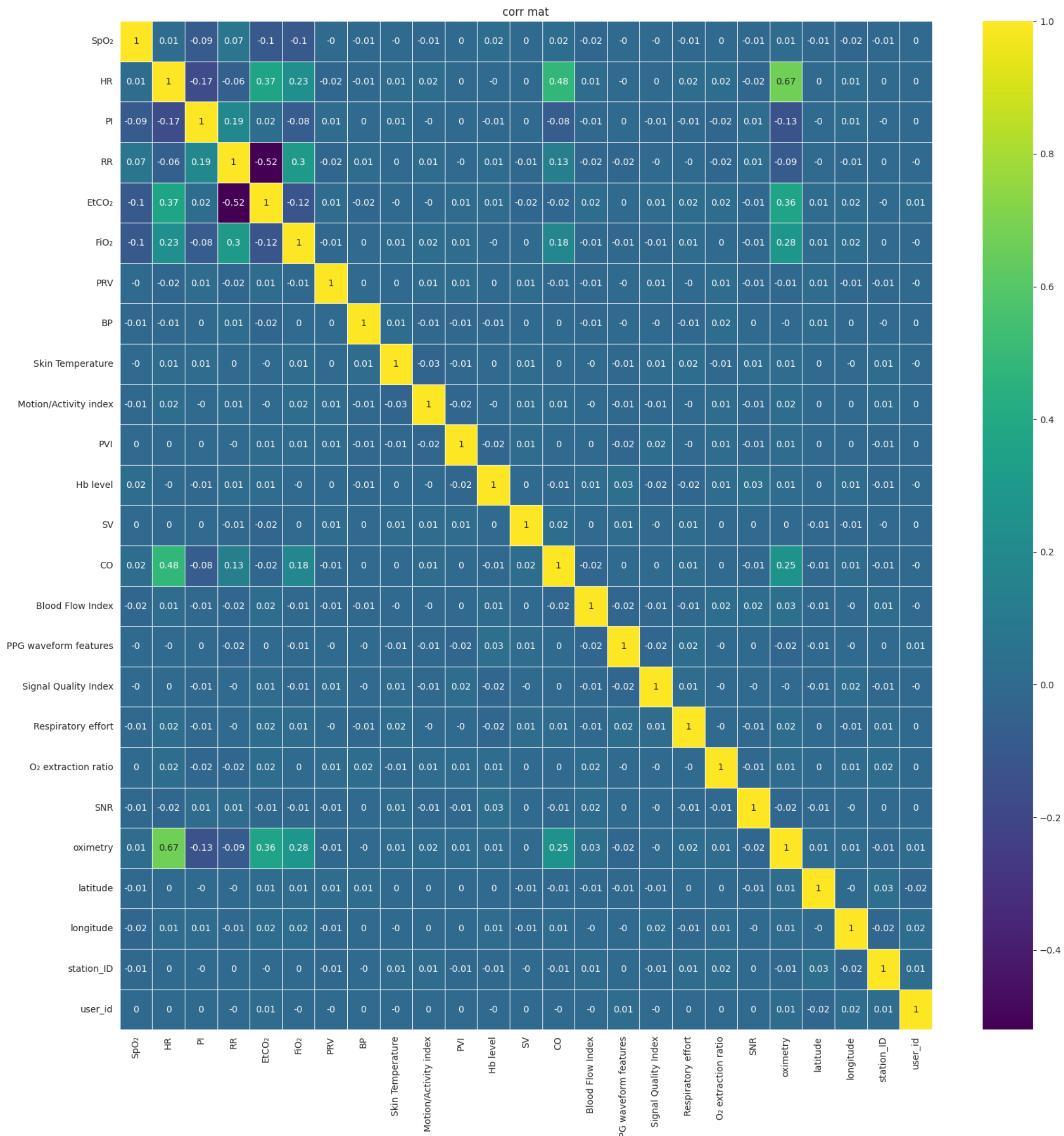
[267]:
```
df.columns
```

[267]:
```
Index(['SpO ', 'HR', 'PI', 'RR', 'EtCO ', 'FiO ', 'PRV', 'BP',
       'Skin Temperature', 'Motion/Activity index', 'PVI', 'Hb level', 'SV',
       'CO', 'Blood Flow Index', 'PPG waveform features',
       'Signal Quality Index', 'Respiratory effort', 'O  extraction ratio',
       'SNR', 'oximetry', 'latitude', 'longitude', 'station_ID', 'QoS', 'code',
       'user_id', 'blood_group'],
      dtype='object')
```

3.3.2 correlation matrix with heatmap

[268]:
```
numeric_df = df.select_dtypes(include=['float64', 'int64'])

corr_matrix = numeric_df.corr().round(2)
```

[269]:
```
plt.figure(figsize=(20, 20))
sns.heatmap(corr_matrix, cmap='viridis', annot=True,linewidths=0.5)
plt.title("corr mat")
plt.show()
```



[270]:

```
#based on this heatmap lets write out all the correlations so we can take a␣
↪look at them later
#I only included minimum 0,2 corr

#GROUP 1 (WITHOUT OXIMETRY):              -- used in this step

#HR - EtCO2(weak), FiO2(weak), CO(mid)
#RR - Etco2(mid), FiO2(weak),

#Group 2(OXIMETRY):
```

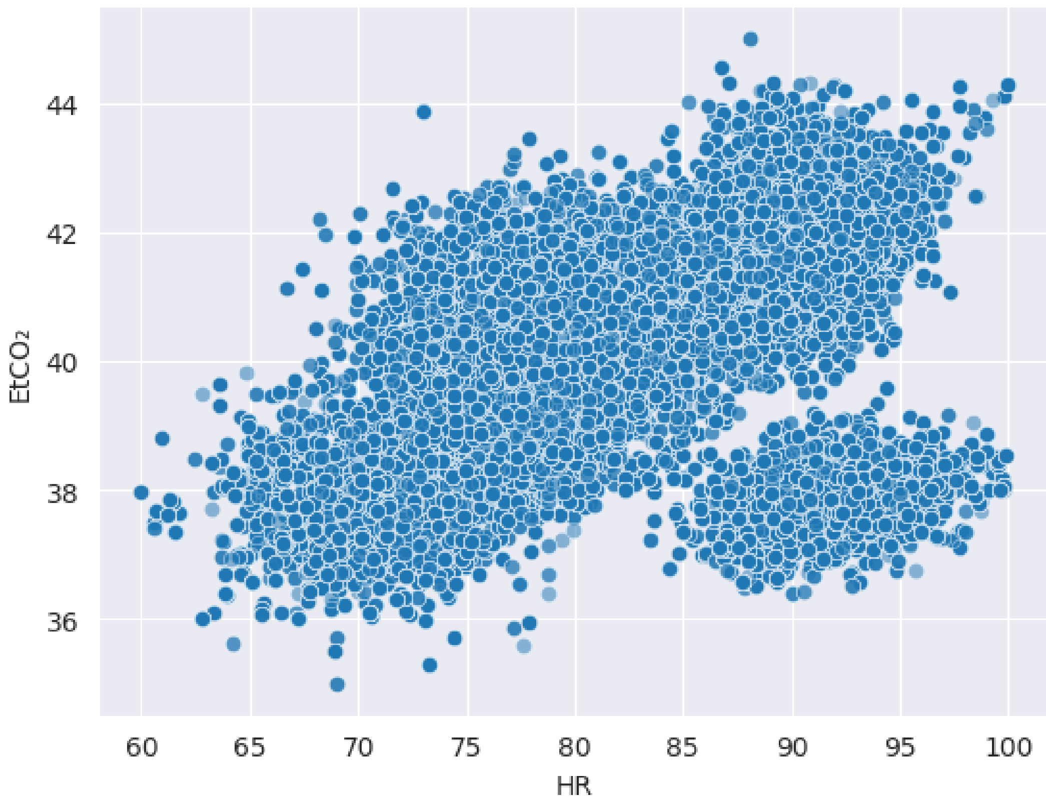
```
#OX - HR(strong),EtCO2(weak) ,FiO2(weak), CO(weak)      -- used in 1D
```

3.3.3 solution for 1.1C)

here are the comparisons between the attributes we got from the correlation heatmap

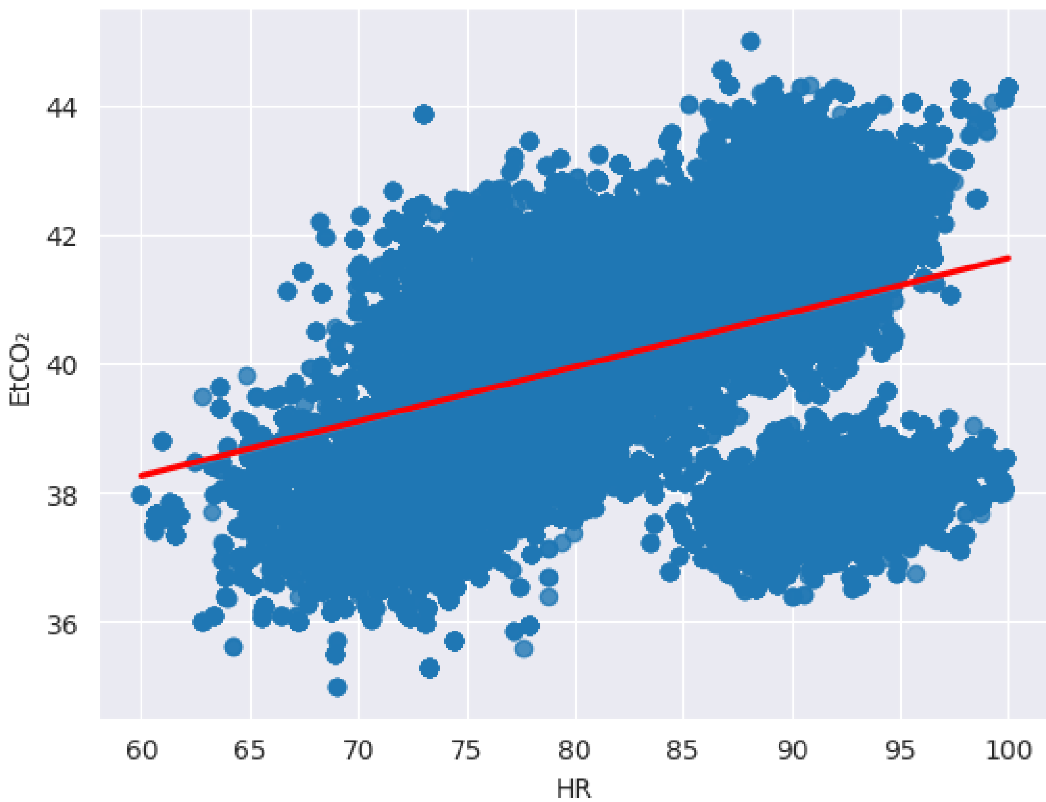
[271]:
```
#HR-EtCO2
sns.scatterplot(data=df, x='HR', y='EtCO ', alpha=0.5)
```

[271]: <Axes: xlabel='HR', ylabel='EtCO '>



[272]:
```
sns.regplot(data=df, x='HR', y='EtCO ', line_kws={'color':'red'})
```

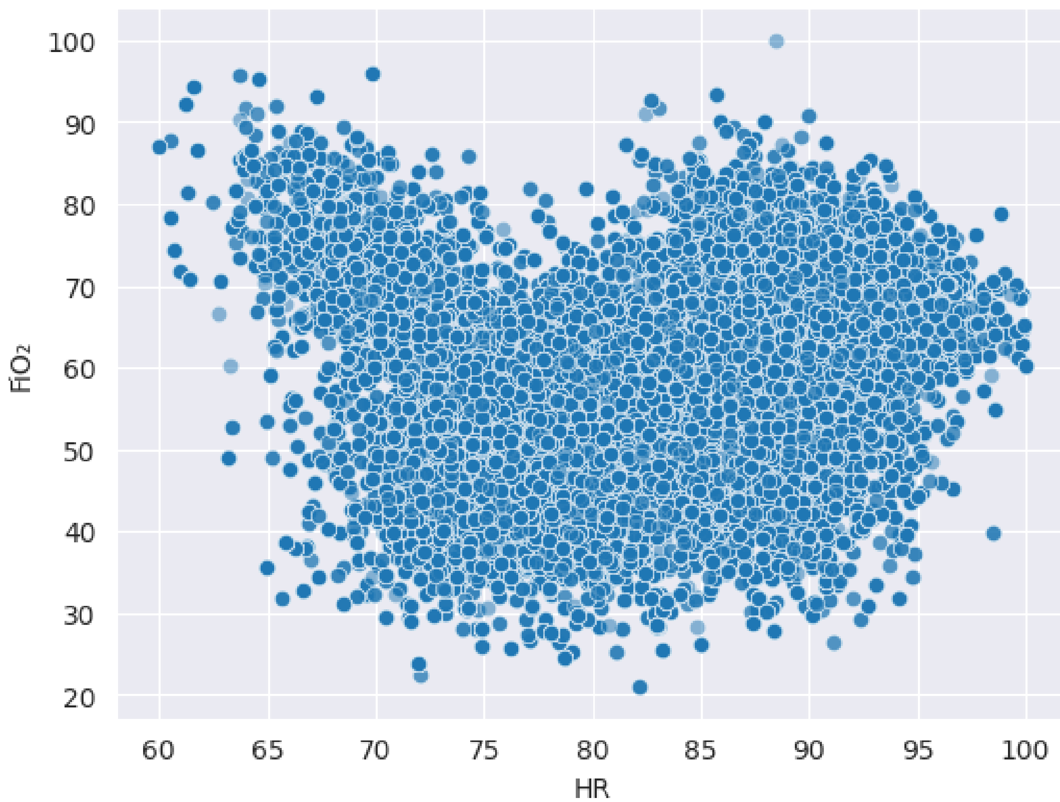
[272]: <Axes: xlabel='HR', ylabel='EtCO '>



[273]:
```
#this is not enough to determine that these two variables correlate, maybe␣
↪without the cluster around [92.5 , 38]
```

[274]:
```
#HR-FiO2
sns.scatterplot(data=df, x='HR', y='FiO ', alpha=0.5)
```

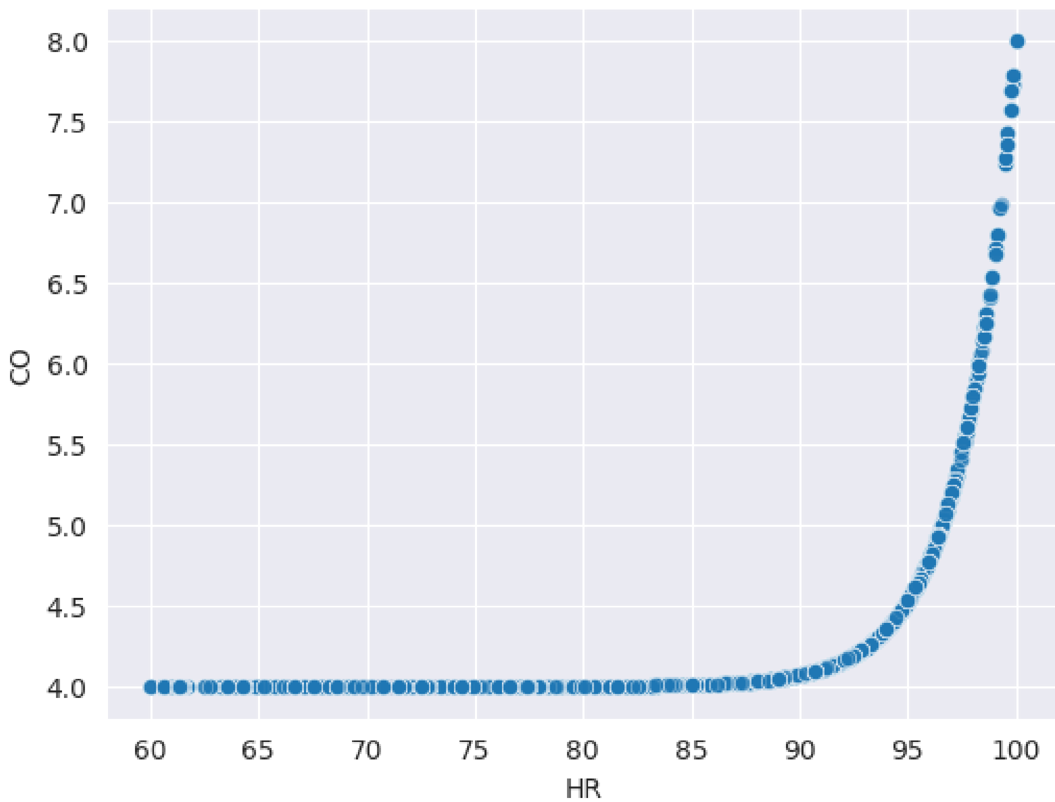
[274]: <Axes: xlabel='HR', ylabel='FiO '>



[275]:
```
#way too spread out, insignificant
```

[276]:
```
#HR-CO
sns.scatterplot(data=df, x='HR', y='CO', alpha=0.5)
```

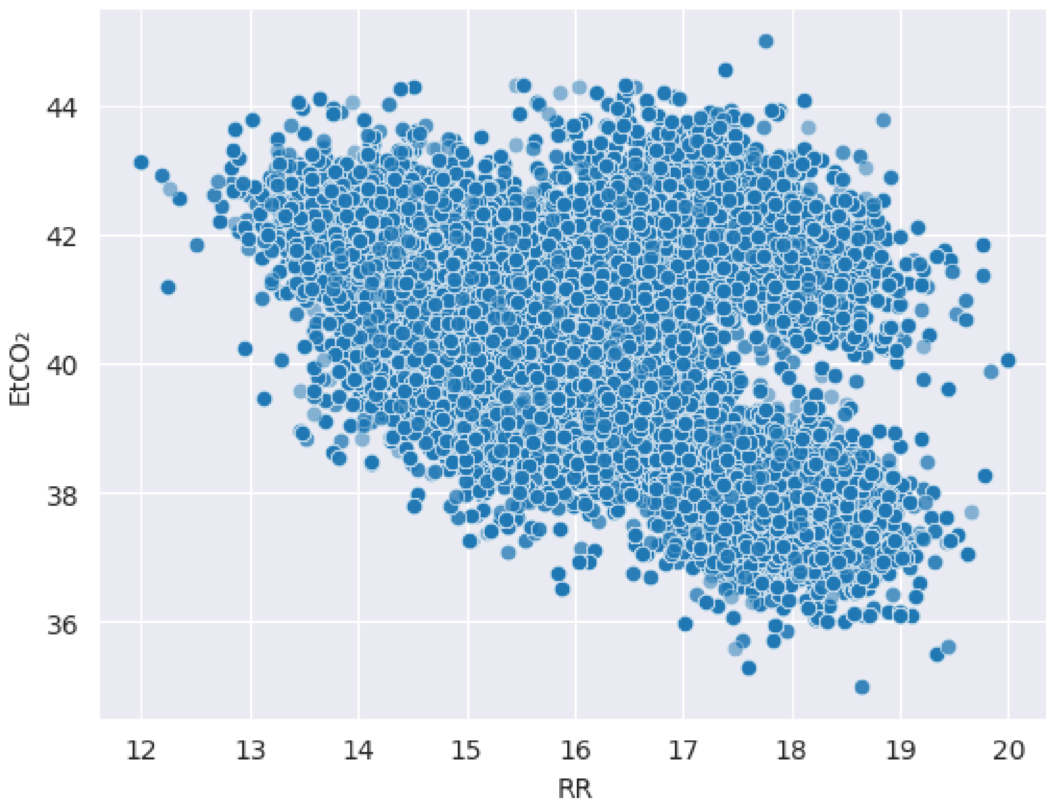
[276]: <Axes: xlabel='HR', ylabel='CO'>



[277]:
```
#This is definitely a significant correlation, but not linear since CO values␣
↪are cut off at 4.0
```

[278]:
```
#RR - Etco2
sns.scatterplot(data=df, x='RR', y='EtCO ', alpha=0.5)
```

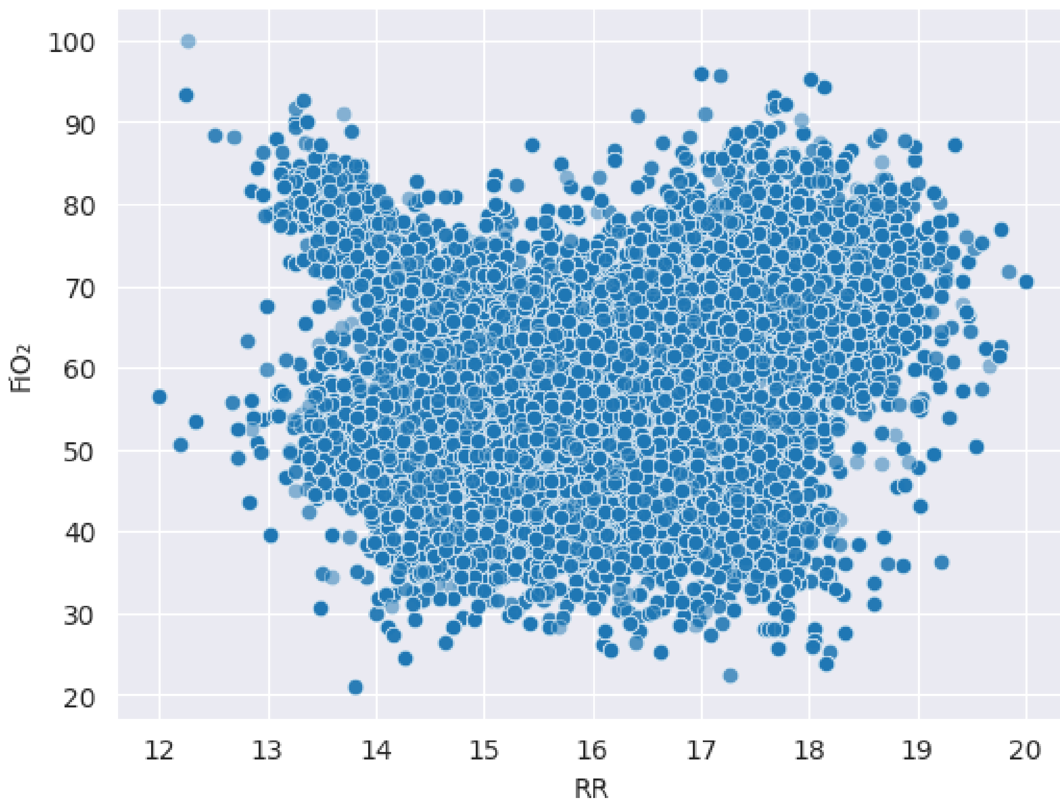
[278]: <Axes: xlabel='RR', ylabel='EtCO '>



[279]:
```
#slight show of negative correlation, but ruined by the upper right cluster
```

[280]:
```
#RR FiO2
sns.scatterplot(data=df, x='RR', y='FiO ', alpha=0.5)
```

[280]:
```
<Axes: xlabel='RR', ylabel='FiO '>
```



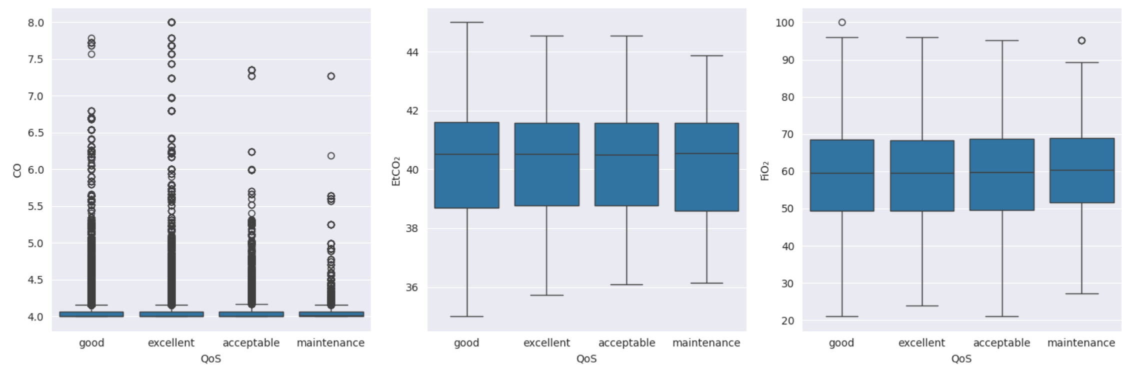
[281]:
```
#no correlation
```

these were only the numeric attributes we took from the heatmap, now lets take at look at non numeric correlations aswell

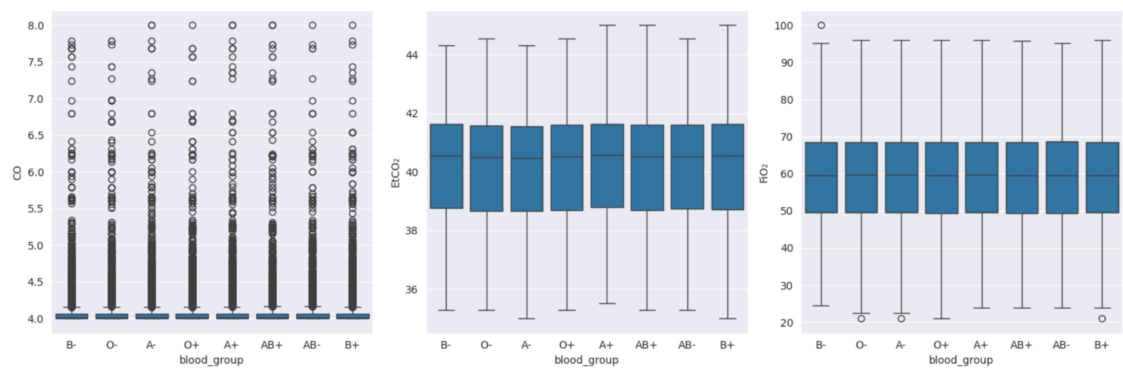
[282]:
```
non_numeric = df.select_dtypes(exclude=['float64', 'int64']).columns
print(non_numeric)
#these will be tested for correlation with CO,FIO2 and EtCO2 since these seem␣
↪to be the most important attributes
```

```
Index(['QoS', 'code', 'blood_group'], dtype='object')
```

[283]:
```
#QoS
fig, axes = plt.subplots(1, 3, figsize=(15, 5))

sns.boxplot(data=df, x='QoS', y='CO', ax=axes[0])
sns.boxplot(data=df, x='QoS', y='EtCO ', ax=axes[1])
sns.boxplot(data=df, x='QoS', y='FiO ', ax=axes[2])

plt.tight_layout()
plt.show()
#no real significance
```

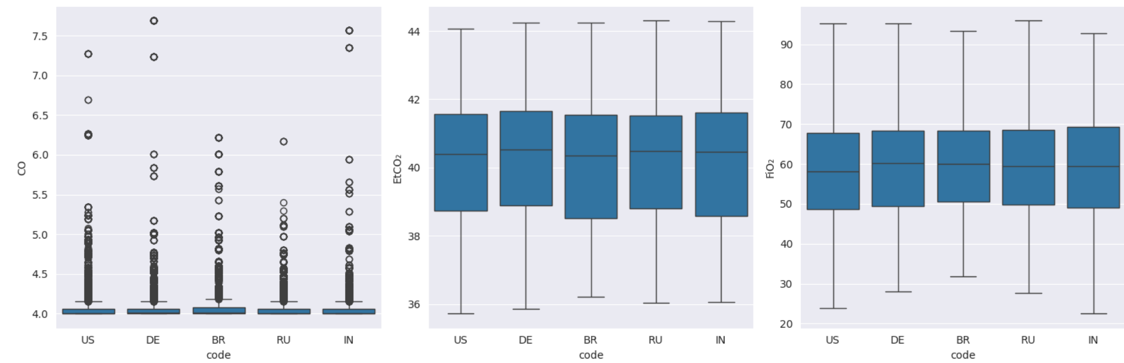


[284]:
```
#QoS
fig, axes = plt.subplots(1, 3, figsize=(15, 5))

sns.boxplot(data=df, x='blood_group', y='CO', ax=axes[0])
sns.boxplot(data=df, x='blood_group', y='EtCO ', ax=axes[1])
sns.boxplot(data=df, x='blood_group', y='FiO ', ax=axes[2])

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



[285]:
```
#code - top 5
top_codes = df['code'].value_counts().head(5).index
filtered_df = df[df['code'].isin(top_codes)]

fig, axes = plt.subplots(1, 3, figsize=(15, 5))

sns.boxplot(data=filtered_df, x='code', y='CO', ax=axes[0])
sns.boxplot(data=filtered_df, x='code', y='EtCO ', ax=axes[1])
sns.boxplot(data=filtered_df, x='code', y='FiO ', ax=axes[2])
```

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

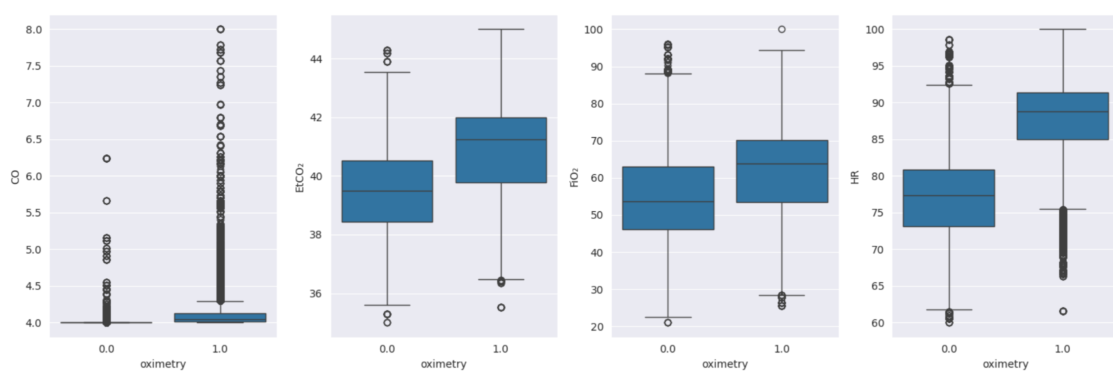


3.4 D)Párová analýza dát: Identifikujte závislosti medzi predikovanou premennou a ostatnými premennými (potenciálnymi prediktormi)

from the correlation heatmap we have these attributes that seem to have some correlation with the predicted attribute

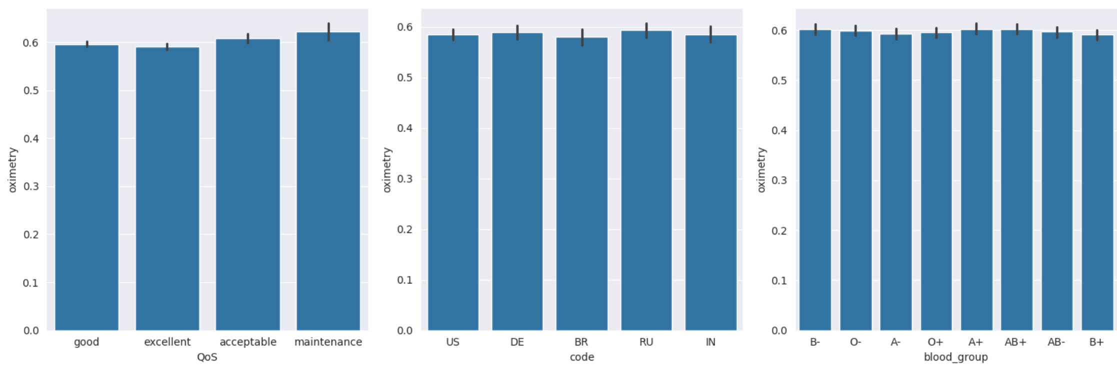
HR(strong),EtCO2(weak) ,FiO2(weak), CO(weak)

[286]:
```
fig, axes = plt.subplots(1, 4, figsize=(15, 5))

sns.boxplot(data=df, x='oximetry', y='CO', ax=axes[0])
sns.boxplot(data=df, x='oximetry', y='EtCO ', ax=axes[1])
sns.boxplot(data=df, x='oximetry', y='FiO ', ax=axes[2])
sns.boxplot(data=df, x='oximetry', y='HR', ax=axes[3])

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



[287]:
```
#CO is hard to determine, the IQRs are close but the outliers may mean a lot␣
↪(will probably see more after normalization), the EtCO2 and FiO2 seem to be␣
↪higher when oximetry is set to 1, HR seems to have a really significant␣
↪correlation with oximetry, as suggested by the correlation heatmap
```

[288]:
```
#correlation with non numerical values
fig, axes = plt.subplots(1, 3, figsize=(15, 5))

# QoS
sns.barplot(ax=axes[0], x=df['QoS'], y=df['oximetry'])
# code (top 5)
top5 = df['code'].value_counts().head(5).index
sns.barplot(ax=axes[1], x=df[df['code'].isin(top5)]['code'], y=df[df['code'].
↪isin(top5)]['oximetry'])
# blood_group
sns.barplot(ax=axes[2], x=df['blood_group'], y=df['oximetry'])

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



[289]:
```
#this graphs show the oximetry 1/0 ratio per each QoS, Code and blood group. So␣
↪rather than correlation, this shows that the data are evenly distributed
```

3.5 E) Findings, thought ect.

all of our thoughs and findings are thoroughly documented throughout the operations, but here is the summary:

Firstly we analysed each table and gave a quick look for every attribute, for dataset station, we used the coordinations to merge it with observation_df, we ended up not using the station name as it is insignificant as well as revision. The only missing values were 2 codes which will probably be filled later with a newly created code. The patient dataframe is full of insignificant attributes, we end up only using the blood group and station ID for merge with station df. From observation we keep every value, although some seem to be more signicicant than others, for example FiO2, CO,RR, EtCO2. We take some time to look at individual distributions of attributes in B), but we dont really find any abnormalities that would pose a threat to our models precition, most distributions are either normal or bimodal or uniform. In C) we needed to take a look at correlation between the attributes, so we needed to create a correlation heatmap which was created from a joint dataframe consisting of all three datasets, but only the attributes that we deemed as important. We discarded the useless features and we will not work with them from that point onward. However we find out minor correlation between HR and EtCO2 as well as a negative one between HR and FiO2 and a RR and EtCO2. HR and CO seem to have a strong positive correlation but since CO is capped at 4.0, it is not shown that nicely. In D) we see that the predicted value oximetry strongly correlates with HR, and FiO2 and EtCO2 also show medium signs of correlation. Correlation with CO is harder to determine without normalising the data first.

4 1.2 Identifikácia problémov, integrácia a čistenie dát

4.1 A) nevhodná štruktúra, nejednotne formaty ,duplikáty, chýbajúce hodnoty, vychýlene hodnoty, abnormalne hodnoty, nelogické vzťahy

[292]:
```
df.columns
```

[292]:
```
Index(['SpO ', 'HR', 'PI', 'RR', 'EtCO ', 'FiO ', 'PRV', 'BP',
       'Skin Temperature', 'Motion/Activity index', 'PVI', 'Hb level', 'SV',
       'CO', 'Blood Flow Index', 'PPG waveform features',
       'Signal Quality Index', 'Respiratory effort', 'O  extraction ratio',
       'SNR', 'oximetry', 'latitude', 'longitude', 'station_ID', 'QoS', 'code',
       'user_id', 'blood_group'],
      dtype='object')
```

[316]:
```
# I see many collumn name that are corectly name but it is so anoying to copy␣
↪the same ` ` symbol over and over again so that's why im gonna rename it just␣
↪for now

rename_map = {
    'SpO ': 'SpO2',
    'EtCO ': 'EtCO2',
    'FiO ': 'FiO2',
    'O  extraction ratio': 'O2 extraction ratio',
}
df = df.rename(columns={k:v for k,v in rename_map.items() if k in df.columns})
```

[320]:
```
for c in ['QoS','code','blood_group']:
    if c in df.columns:
        df[c] = df[c].astype('category')
```

[300]:
```
#checking if every value in oximetry col is type int so it does not make any␣
↪issues in the future but most of this is already done in previous cells in 1.
↪1
```

```python
if 'oximetry' in df.columns:
    df['oximetry'] = df['oximetry'].astype(int)
```

[323]:
```python
df.dtypes.value_counts()
```

[323]:
```
float64     22
int64        3
category     1
category     1
category     1
Name: count, dtype: int64
```

[308]:
```python
df.duplicated().sum()
```

[308]:
```
np.int64(0)
```

[328]:
```python
df.duplicated(subset=['HR','RR','BP']).sum()
```

[328]:
```
np.int64(55244)
```

[336]:
```python
duplicates_output = df[df.duplicated(subset=[col for col in df.columns if col !
 ↪= 'user_id'], keep=False)]
duplicates_output
```

[336]:
```
            SpO2         HR         PI         RR       EtCO2       FiO2  \
6      97.933271  80.787303  11.730935  14.964972  39.537692  65.326035
7      97.933271  80.787303  11.730935  14.964972  39.537692  65.326035
9      97.933271  80.787303  11.730935  14.964972  39.537692  65.326035
30     96.851933  91.008225  11.323571  15.148349  42.056062  58.615353
32     96.851933  91.008225  11.323571  15.148349  42.056062  58.615353
…            …          …          …          …          …          …
66965  96.698981  71.499309  13.427977  15.751513  41.201634  55.177636
66966  96.698981  71.499309  13.427977  15.751513  41.201634  55.177636
66967  96.698981  71.499309  13.427977  15.751513  41.201634  55.177636
66968  96.698981  71.499309  13.427977  15.751513  41.201634  55.177636
66970  96.698981  71.499309  13.427977  15.751513  41.201634  55.177636

              PRV          BP  Skin Temperature  Motion/Activity index  …  \
6      110.615787  102.133386         36.274352               8.975704  …
7      110.615787  102.133386         36.274352               8.975704  …
9      110.615787  102.133386         36.274352               8.975704  …
30      77.660061  102.695264         35.163246              11.823341  …
32      77.660061  102.695264         35.163246              11.823341  …
…             …           …                 …                    …  …
66965  103.386006  102.693239         34.895596              11.429244  …
66966  103.386006  102.693239         34.895596              11.429244  …
66967  103.386006  102.693239         34.895596              11.429244  …
66968  103.386006  102.693239         34.895596              11.429244  …
```

```
66970  103.386006  102.693239        34.895596            11.429244  …

       O2 extraction ratio        SNR  oximetry  latitude  longitude  \
6                 0.290879  26.006709         0  33.54428  -84.23381
7                 0.290879  26.006709         0  33.54428  -84.23381
9                 0.290879  26.006709         0  33.54428  -84.23381
30                0.285955  32.768112         1  10.29085  105.75635
32                0.285955  32.768112         1  10.29085  105.75635
…                      …         …        …         …         …
66965             0.283565  28.887543         1  51.04962   12.13690
66966             0.283565  28.887543         1  51.04962   12.13690
66967             0.283565  28.887543         1  51.04962   12.13690
66968             0.283565  28.887543         1  51.04962   12.13690
66970             0.283565  28.887543         1  51.04962   12.13690

       station_ID        QoS  code  user_id  blood_group
6             426  excellent    US      398           O+
7             426  excellent    US      988           O+
9             426  excellent    US     2033           O+
30            663  excellent    VN     1225           A-
32            663  excellent    VN     1454           A-
…           …        …   …      …         …
66965         223  excellent    DE      575           B-
66966         223  excellent    DE      468           O+
66967         223  excellent    DE     1212           B-
66968         223  excellent    DE      928           B-
66970         223  excellent    DE       51           B-

[21334 rows x 28 columns]
```

[337]:
```python
duplicates_output['user_id'].nunique(), len(duplicates_output)
```

[337]:
```
(599, 21334)
```

[341]:
```python
duplicates_output['station_ID'].value_counts().head()
```

[341]:
```
station_ID
186    468
8      414
350    352
208    318
223    318
Name: count, dtype: int64
```

After checking for duplicates, we found that some patients shared identical measurements. It is normal but not if we found out that 21 336 row that are similar. This was caused by merging patient_df and observation_df using station_ID.

[351]:
```
na_count = df.isna().sum().sort_values(ascending=False)
na_count
```

[351]:
```
code                      170
SpO2                        0
PI                          0
HR                          0
EtCO2                       0
FiO2                        0
PRV                         0
BP                          0
Skin Temperature            0
Motion/Activity index       0
PVI                         0
RR                          0
Hb level                    0
SV                          0
Blood Flow Index            0
CO                          0
Signal Quality Index        0
Respiratory effort          0
O2 extraction ratio         0
PPG waveform features       0
SNR                         0
oximetry                    0
longitude                   0
latitude                    0
station_ID                  0
QoS                         0
user_id                     0
blood_group                 0
dtype: int64
```

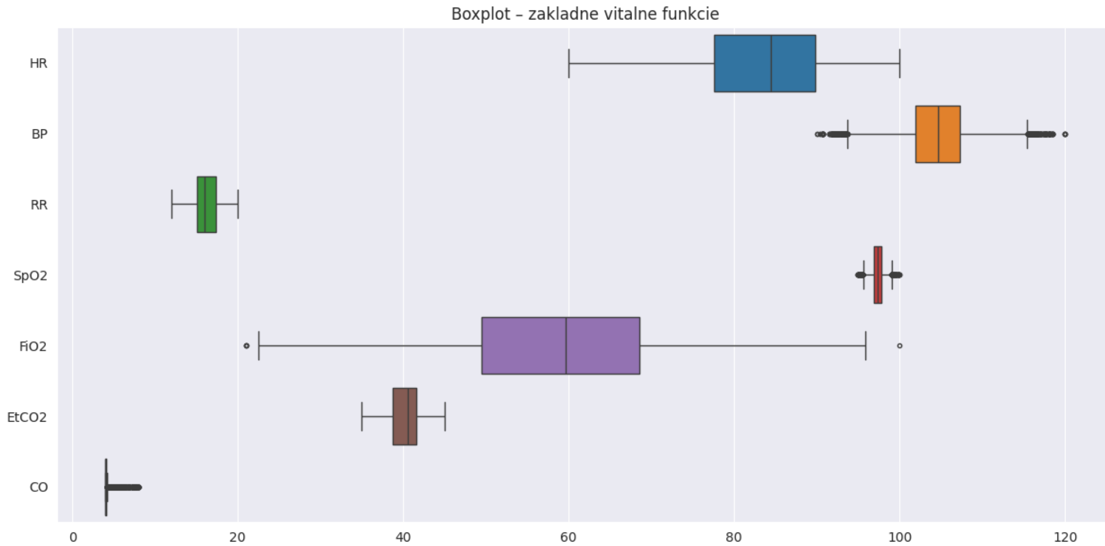
4.2 B) Kontrola správnosť v dátach

Most of the analyzing the atributes was done in part 1.1 that includes all the correlation between columns and also checking if the dataset involve some useless data connection that was stated in last blog. There are some duplicates that will be removed in future. Cause of these duplicates were connecting datasets with many to many multiplicity

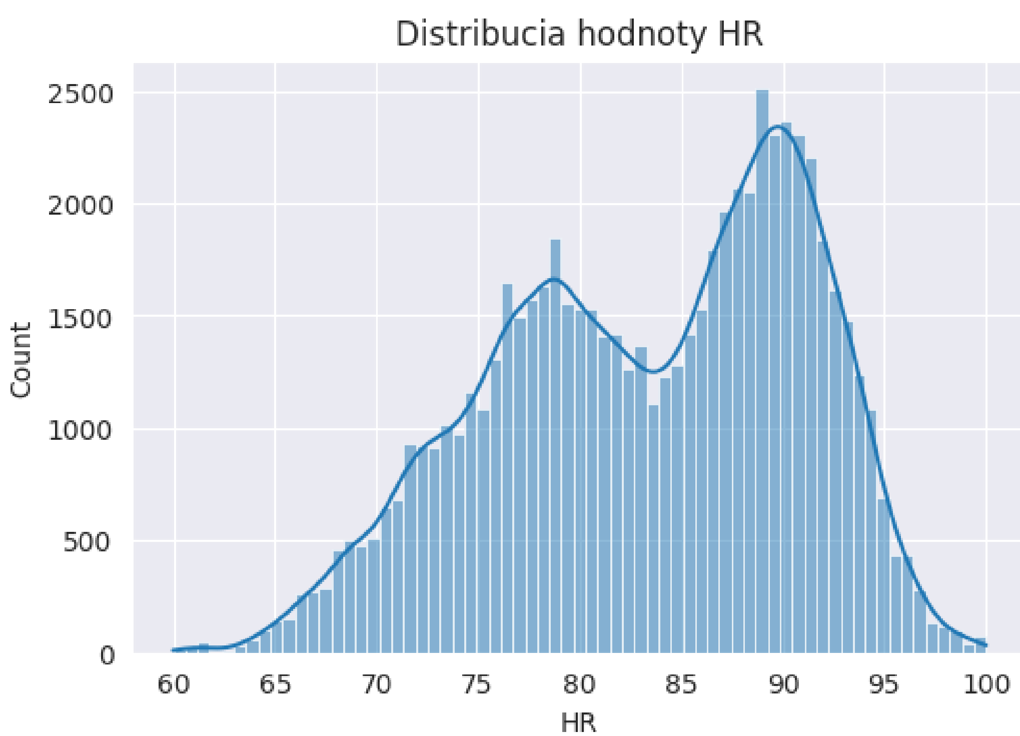
4.3 C) Outlier detection

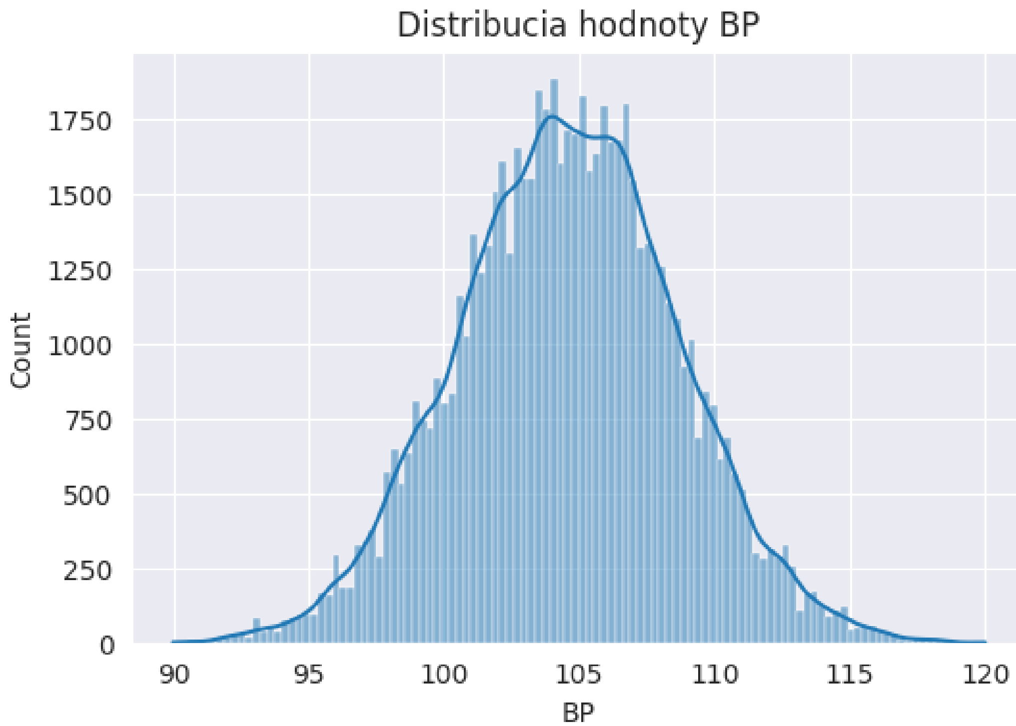
[374]:
```
cols1 = ['HR', 'BP', 'RR', 'SpO2', 'FiO2', 'EtCO2',"CO"]
plt.figure(figsize=(12, 6))
sns.boxplot(data=df[cols1], orient='h', fliersize=3)
plt.title('Boxplot - zakladne vitalne funkcie')
plt.tight_layout()
```

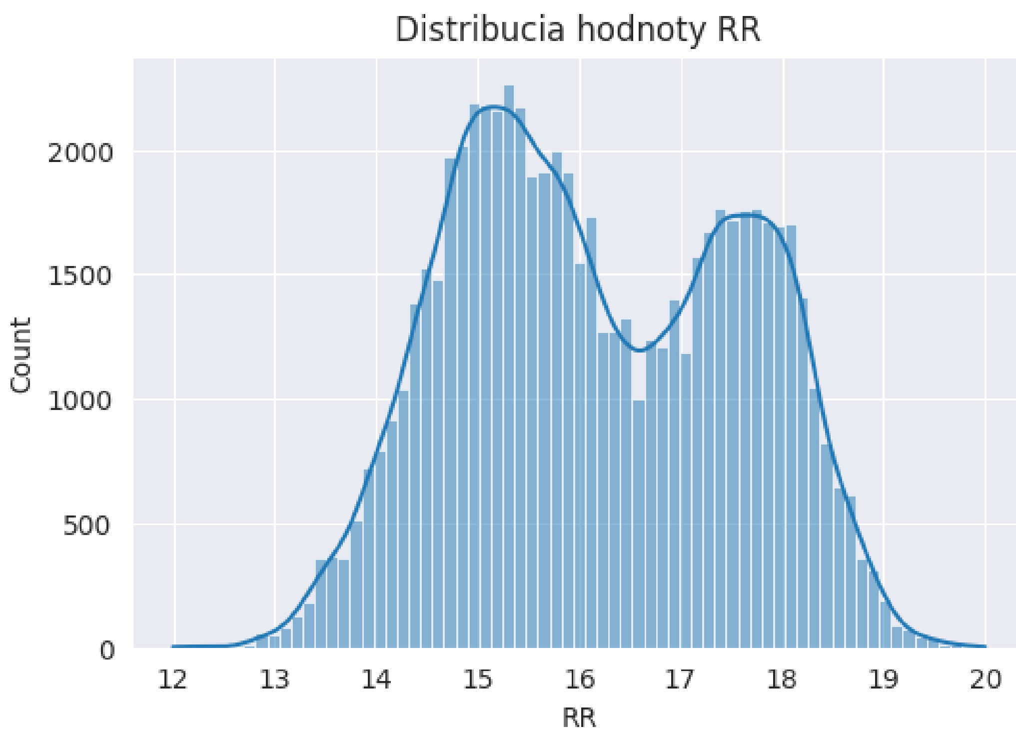
```
plt.show()
```

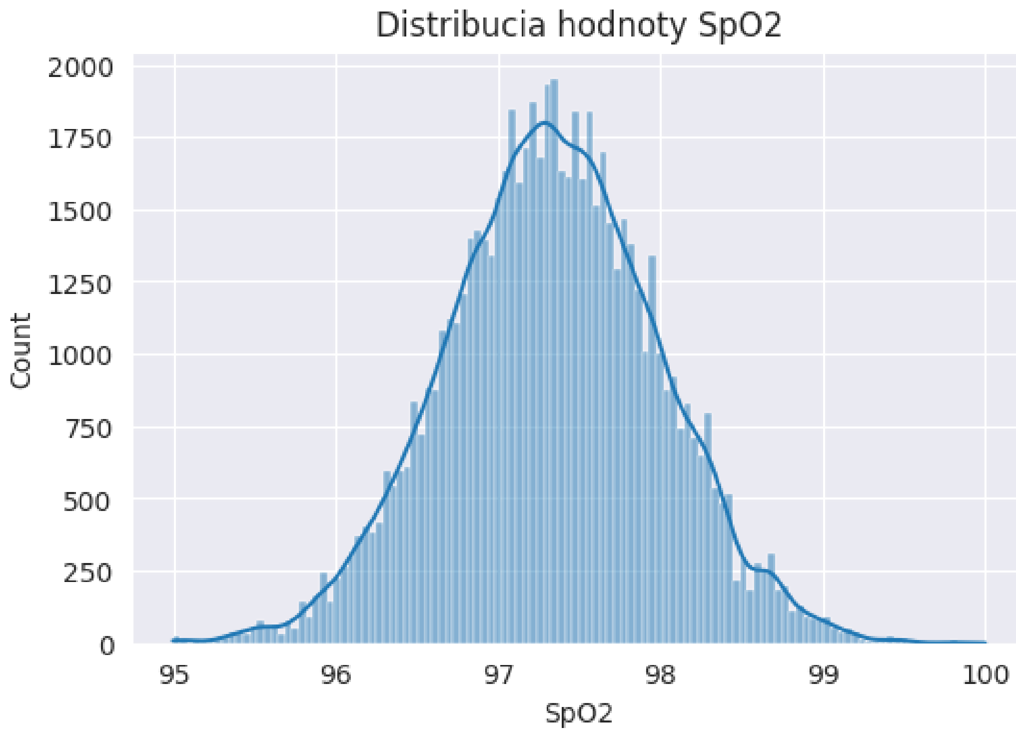


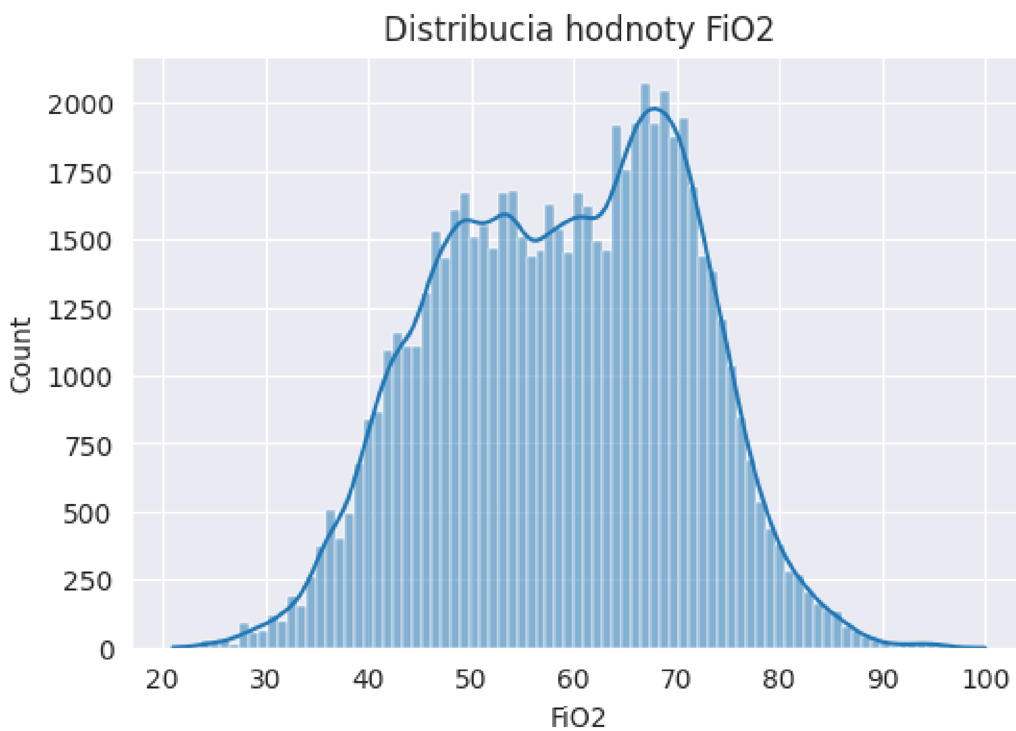
[375]:
```
for col in cols1:
    plt.figure(figsize=(6, 4))
    sns.histplot(df[col], kde=True)
    plt.title(f'Distribucia hodnoty {col}')
    plt.show()
```

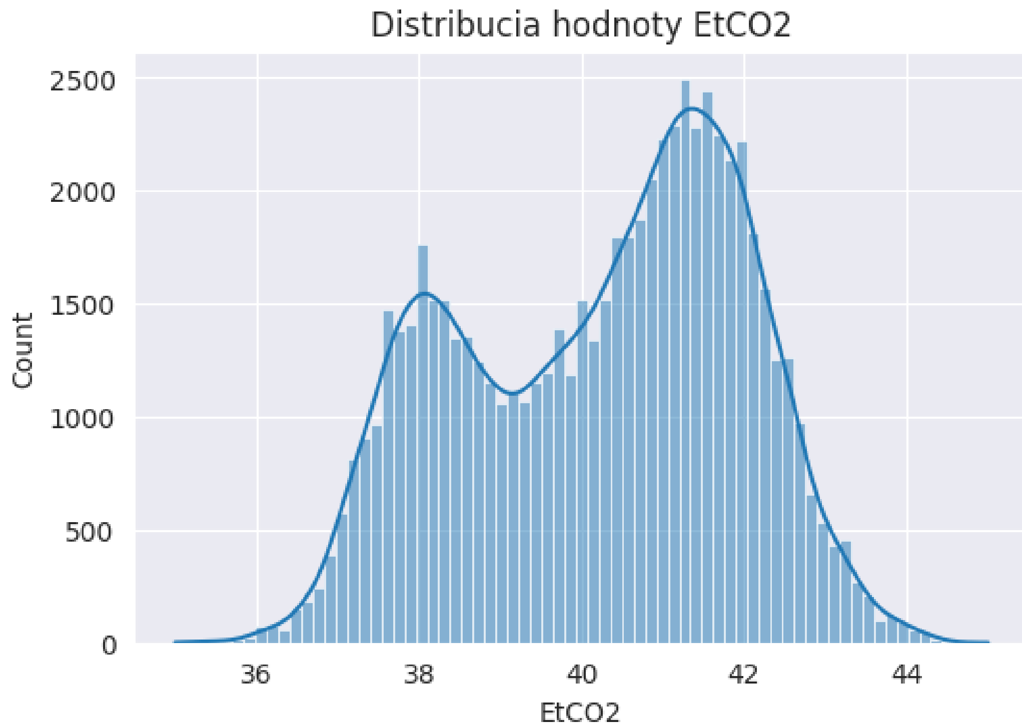


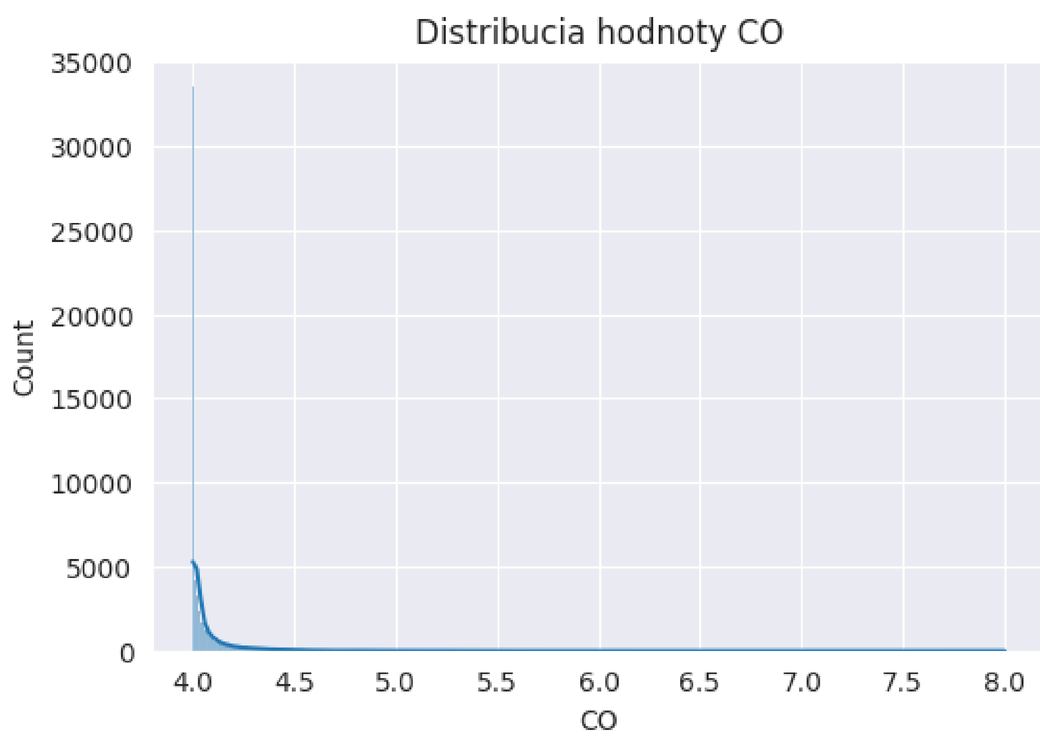

Distribucia hodnoty BP
Count
BP
0
250
500
750
1000
1250
1500
1750
90
95
100
105
110
115
120


Distribucia hodnoty RR
Count
RR
0
500
1000
1500
2000
12
13
14
15
16
17
18
19
20


Distribucia hodnoty SpO2
Count
SpO2
0
250
500
750
1000
1250
1500
1750
2000
95
96
97
98
99
100

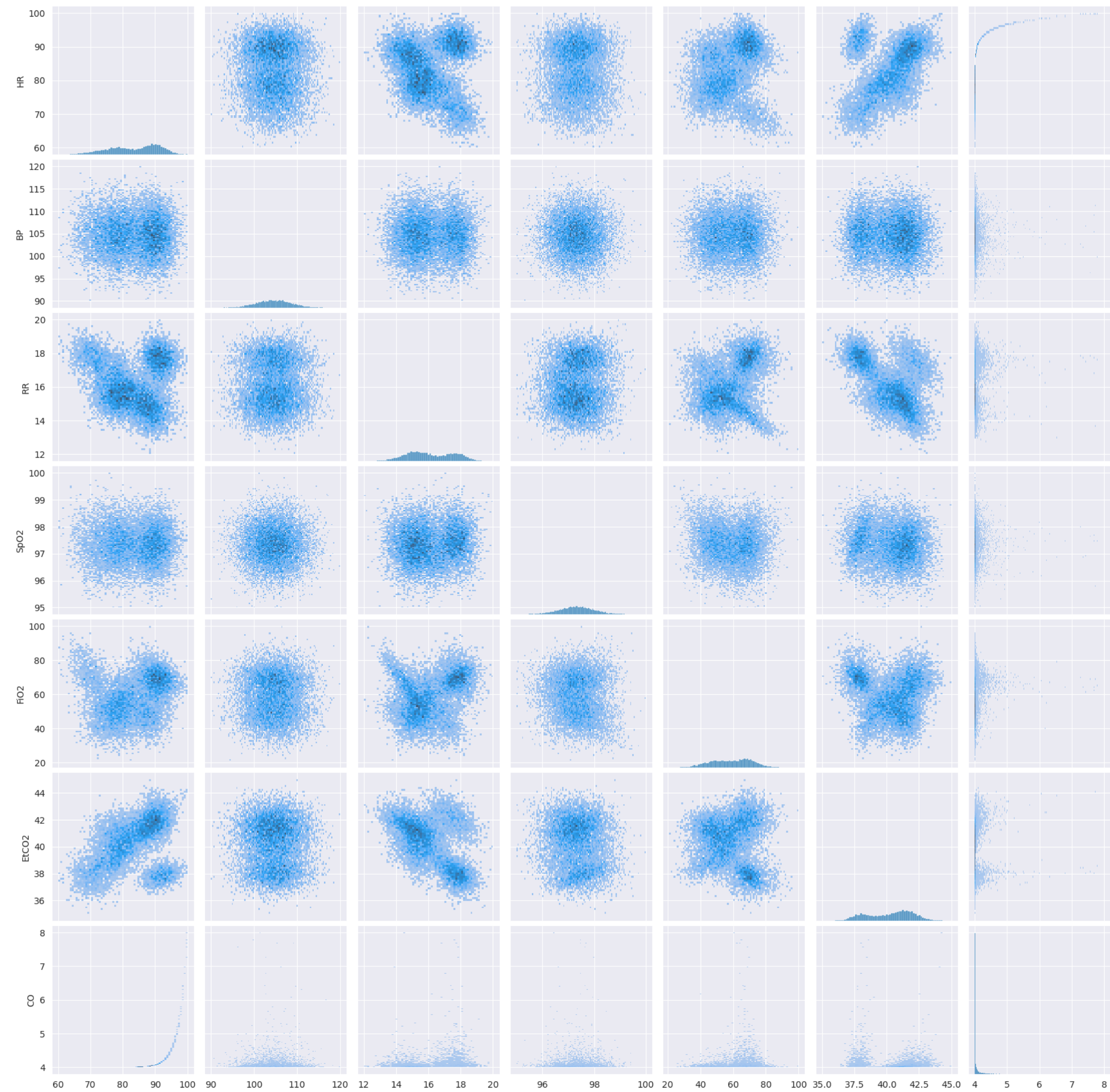

Distribucia hodnoty FiO2
Count
FiO2
0
250
500
750
1000
1250
1500
1750
2000
20
30
40
50
60
70
80
90
100


Distribucia hodnoty EtCO2
Count
EtCO2
0
500
1000
1500
2000
2500
36
38
40
42
44

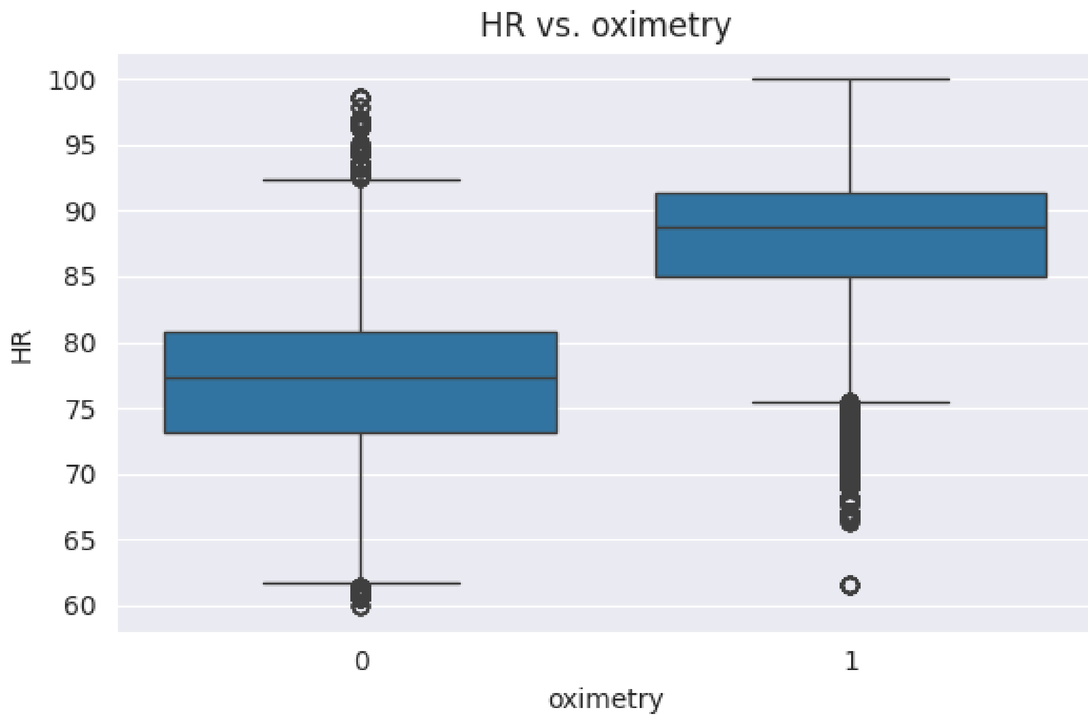

Distribucia hodnoty CO
Count
CO
0
5000
10000
15000
20000
25000
30000
35000
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0

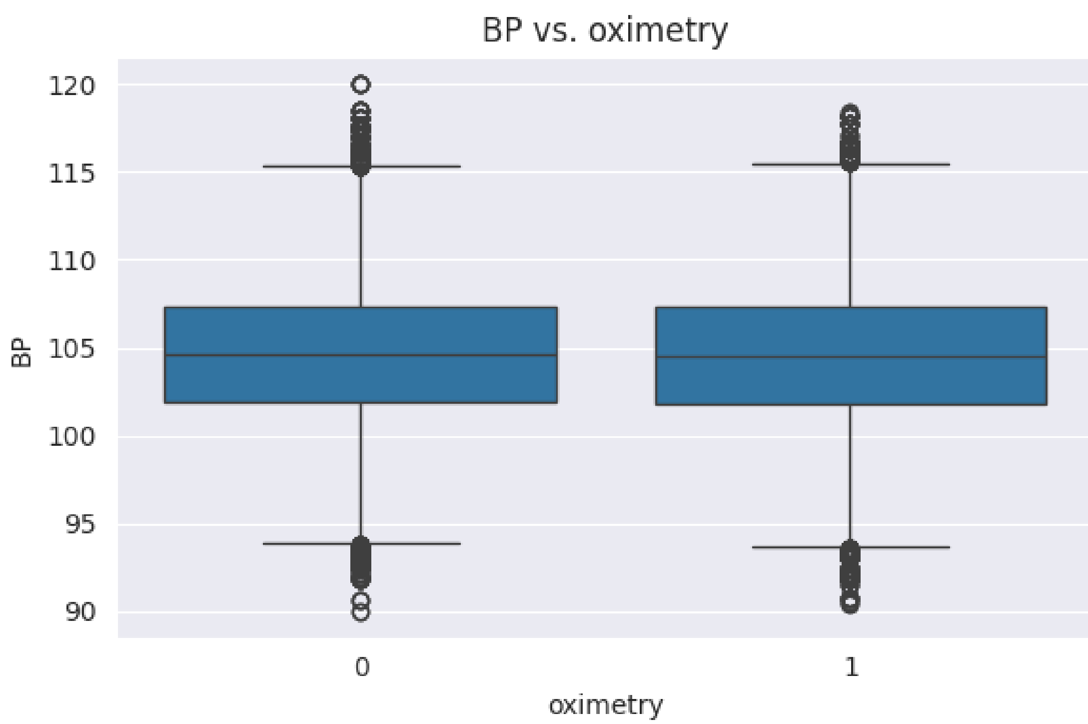
[376]:
```
sns.pairplot(data=df[cols1], kind='hist')
```

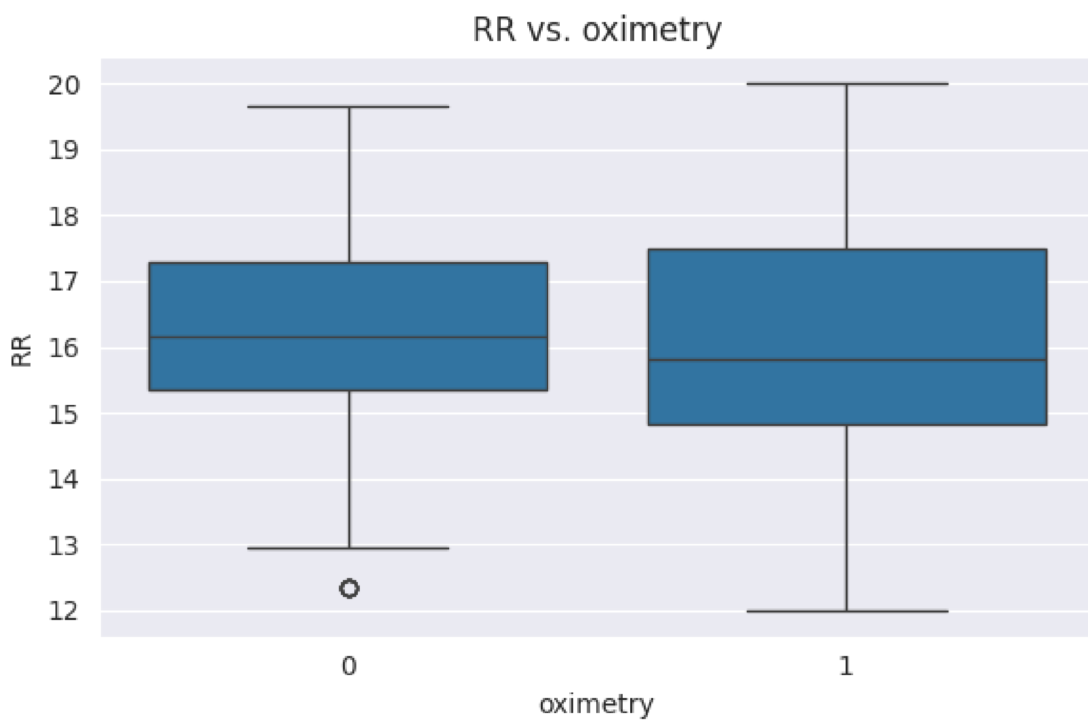
[376]: <seaborn.axisgrid.PairGrid at 0x1c06e87d6d0>



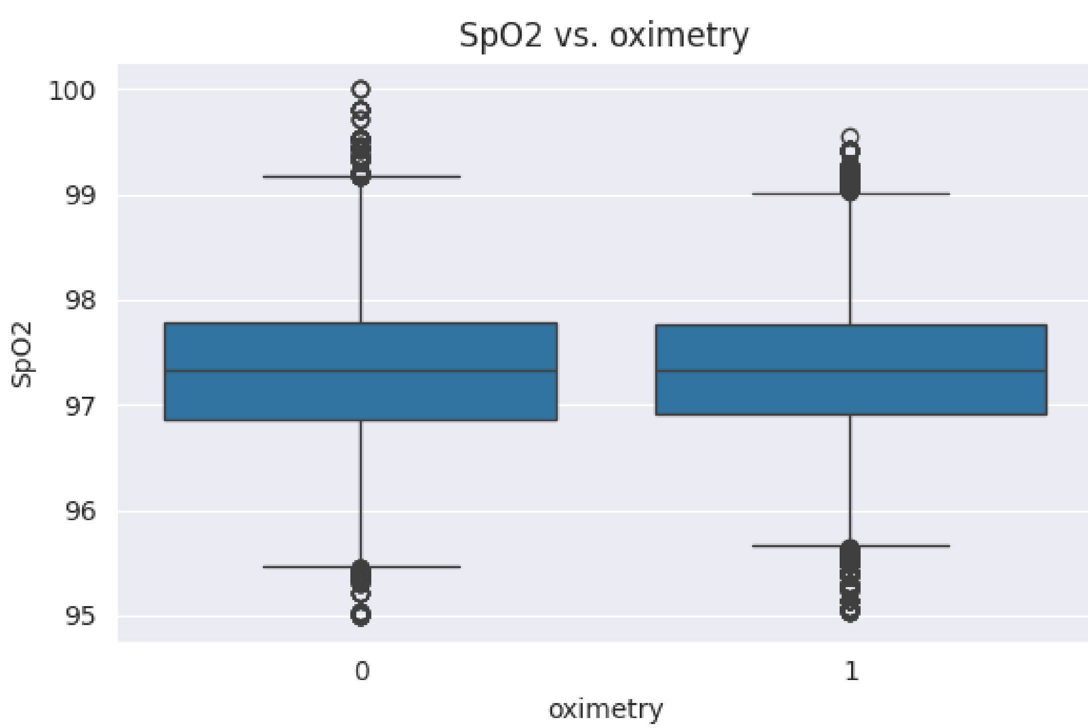
[377]:
```
for col in cols1:
    plt.figure(figsize=(6,4))
    sns.boxplot(data=df, x='oximetry', y=col)
    plt.title(f'{col} vs. oximetry')
    plt.tight_layout()
    plt.show()
```


HR vs. oximetry
100
95
90
85
80
75
70
65
60
HR
0
1
oximetry

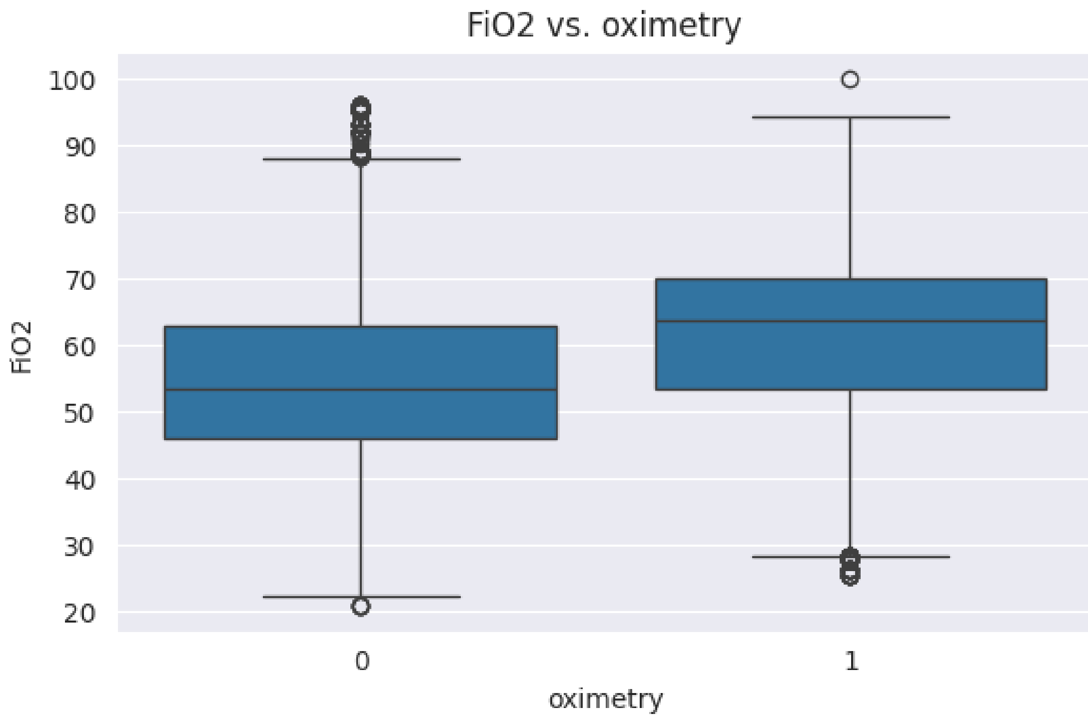

BP vs. oximetry
120
115
110
105
100
95
90
BP
0
1
oximetry



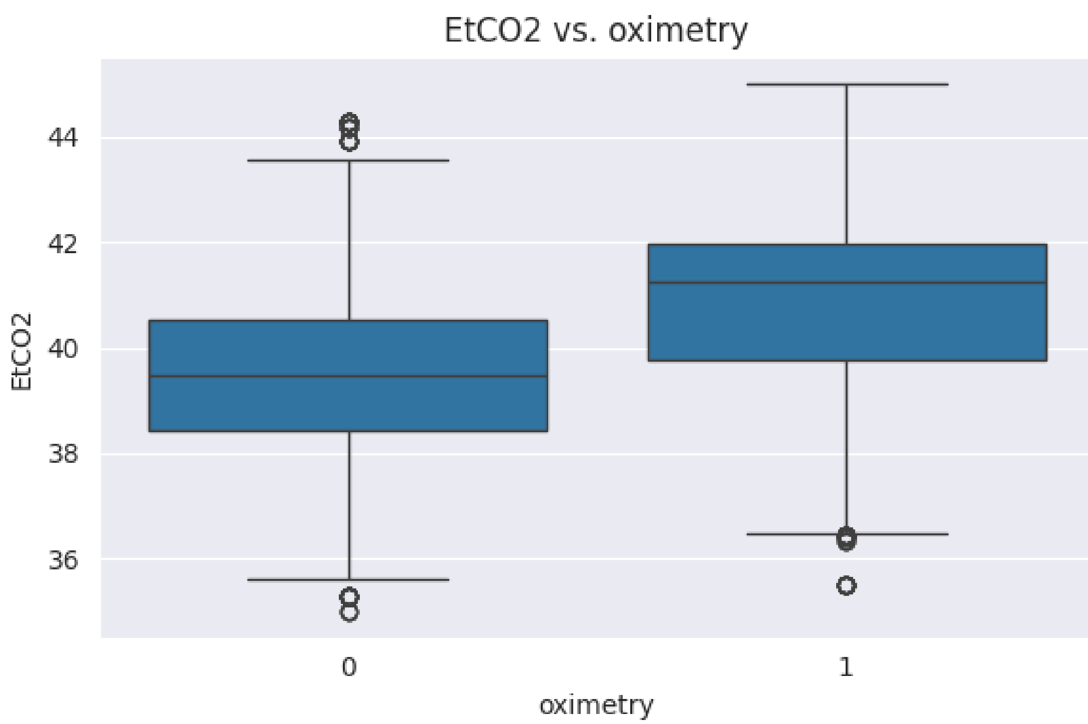
RR vs. oximetry
20
19
18
17
16
15
14
13
12
RR
0
1
oximetry



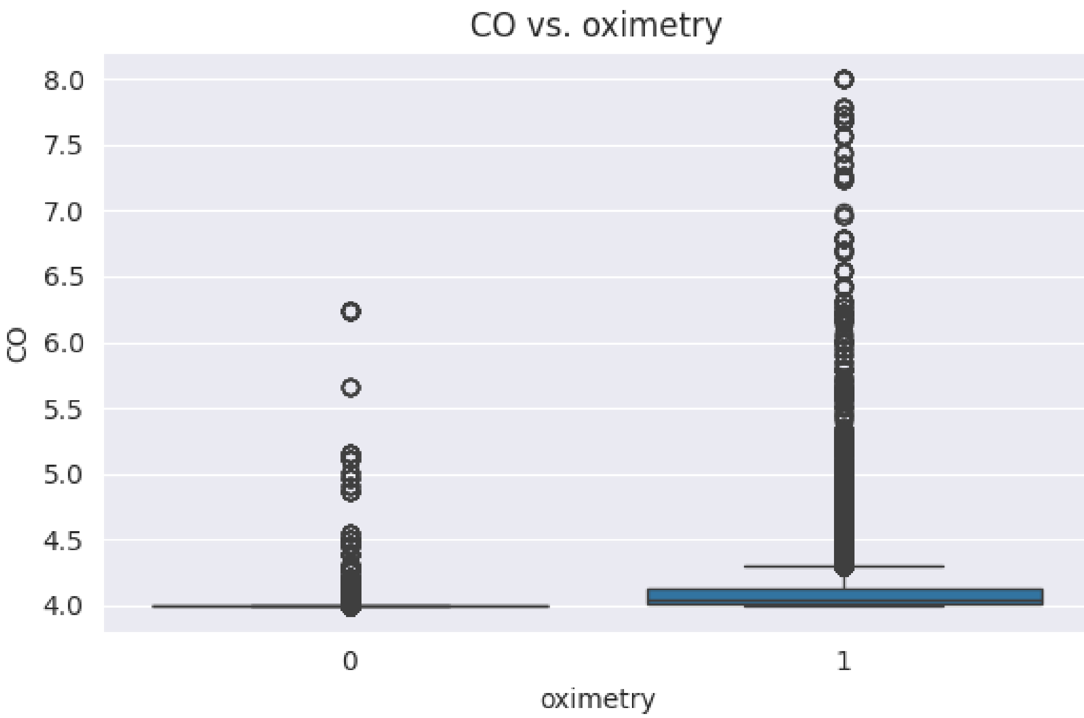
SpO2 vs. oximetry
100
99
98
97
96
95
SpO2
0
1
oximetry



FiO2 vs. oximetry
FiO2
100
90
80
70
60
50
40
30
20
0
1
oximetry



EtCO2 vs. oximetry
EtCO2
44
42
40
38
36
0
1
oximetry



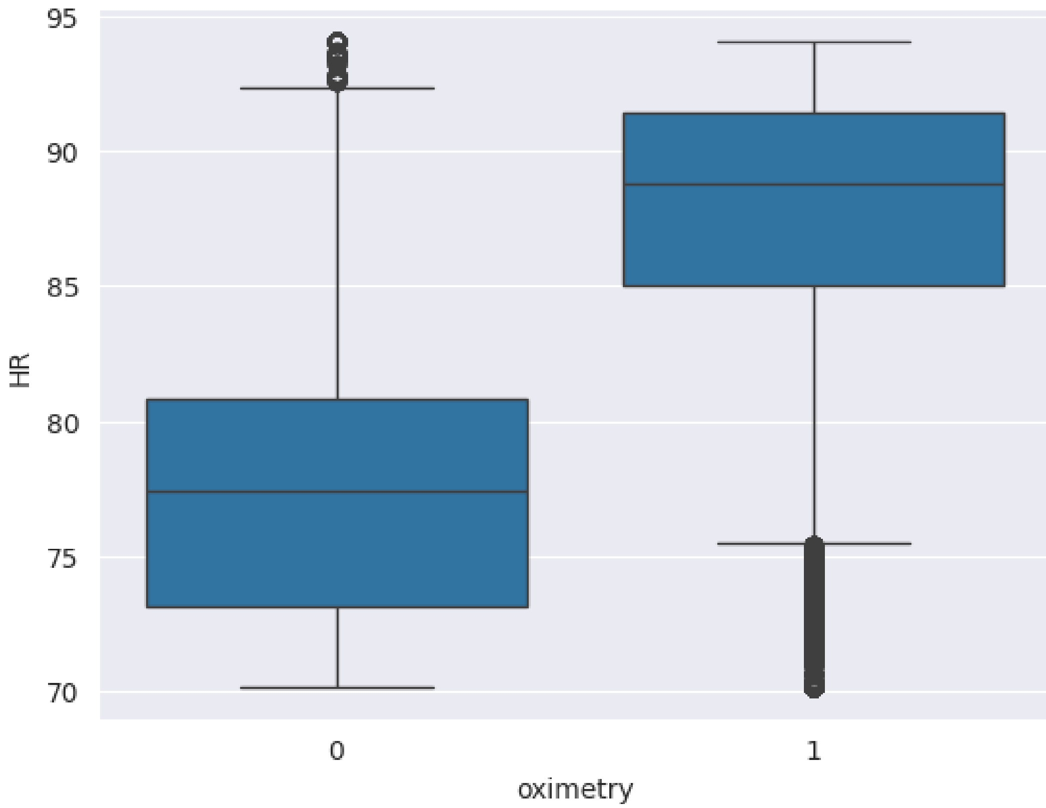
[ ]:
```
# HR (Heart Rate) showed several extreme values .Very high for oximetry = 0 and␣
↪very low for oximetry = 1.
#These outliers can affect model accuracy, so in the next phase we will clean␣
↪them using the IQR method and winsorization (5th-95th percentile) to keep␣
↪only realistic heart rate values.

#The CO variable shows a highly right-skewed distribution with a long upper␣
↪tail, indicating a large number of potential outliers compared to other␣
↪attributes.
#These extreme values are likely to distort the model, so CO will require␣
↪normalization or outlier treatment (IQR filtering or winsorization).
```

[393]:
```
# Calculate 5th and 95th percentiles
low, high = df['HR'].quantile([0.05, 0.95])

# before winsorization
before = ((df['HR'] < low) | (df['HR'] > high)).sum()

# apply wins.
df_win = df.copy()
df_win['HR'] = df_win['HR'].clip(lower=low, upper=high)

# after winsorization
```

```
after = ((df_win['HR'] < low) | (df_win['HR'] > high)).sum()

print(f"Before winsorization: {before} outliers")
print(f"After winsorization: {after} outliers")

sns.boxplot(data = df_win, x='oximetry', y='HR')
```

```
Pred winsorizáciou: 6696 outlierov
Po winsorizácii: 0 outlierov
```

[393]: <Axes: xlabel='oximetry', ylabel='HR'>



Now I can see that there is not such a values that are higher or lower to quntile range from 0.05 to 0.95 that's why we did not get any after winsortization value on the other hand we caught some outliers with the IQR method on df["CO"] atribute because he included many abnormal values

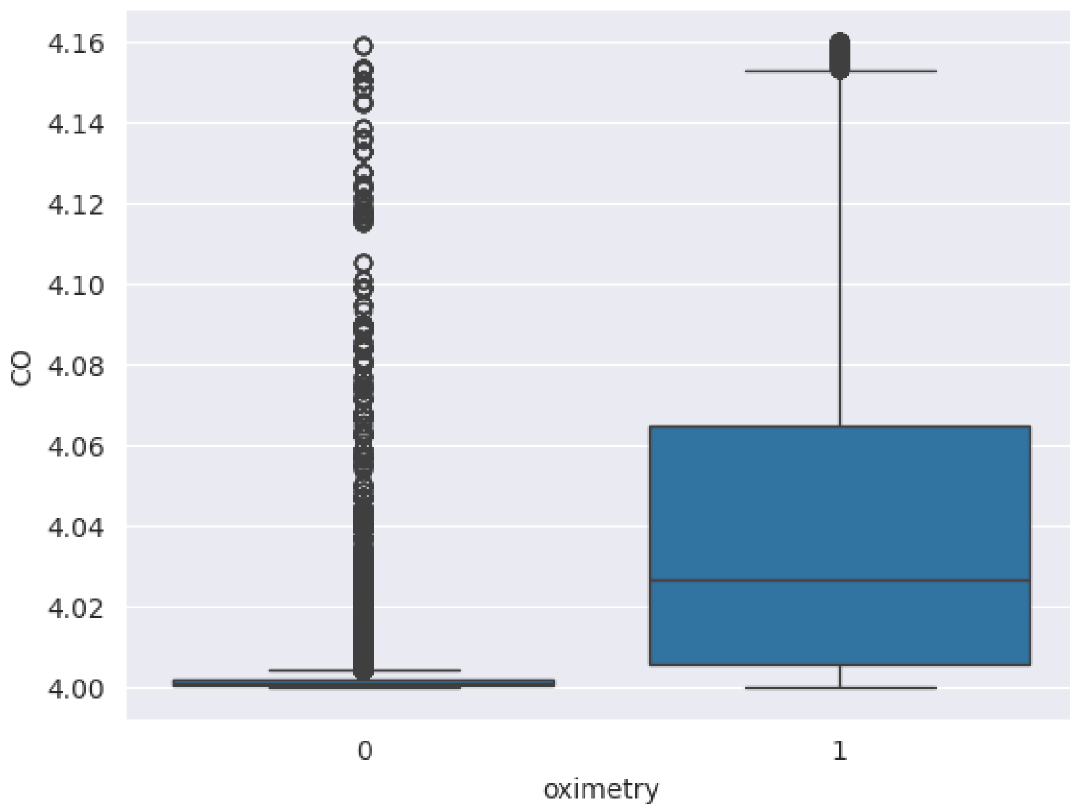
[398]:
```
# IQR bounders
Q1 = df['CO'].quantile(0.25)
Q3 = df['CO'].quantile(0.75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
```

```
upper_bound = Q3 + 1.5 * IQR

print(f"CO bounds (IQR): {lower_bound:.2f} - {upper_bound:.2f}")

# delete values out of range
df_iqr = df[(df['CO'] >= lower_bound) & (df['CO'] <= upper_bound)]

print(f"Removed {len(df) - len(df_iqr)} rows")
print(len(df_iqr))
```

```
CO bounds (IQR): 3.91 - 4.16
Removed 8460 rows
58513
```

[399]:
```
sns.boxplot(data=df_iqr, x='oximetry', y='CO')
```

[399]: <Axes: xlabel='oximetry', ylabel='CO'>



On the graph above we can see extreme difference on outlier values because the previous graph had values close to 9 and this has maximum values 4.16

[401]:
```python
# all number collumns exept oximetry
num_cols = df.select_dtypes(include='number').columns.drop('oximetry',␣
 ↪errors='ignore')

#add collumns if in range of upper or lower bounder
iqr_summary = []
for col in num_cols:
    q1, q3 = df[col].quantile([0.25, 0.75])
    iqr = q3 - q1
    lower, upper = q1 - 1.5 * iqr, q3 + 1.5 * iqr
    outliers = ((df[col] < lower) | (df[col] > upper)).sum()
    iqr_summary.append([col, outliers, round(outliers / len(df) * 100, 2)])

# Display as a small DataFrame
iqr_df = pd.DataFrame(iqr_summary, columns=['Column', 'Outliers', 'Percent'])
iqr_df.sort_values('Percent', ascending=False, inplace=True)
iqr_df
```

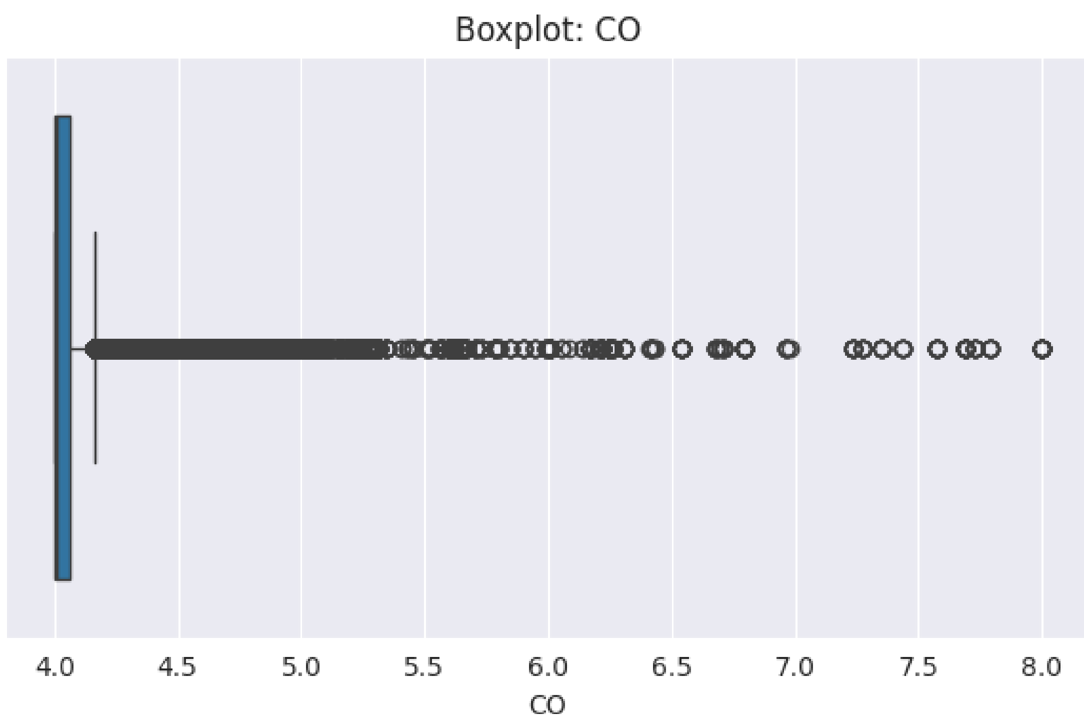
[401]:

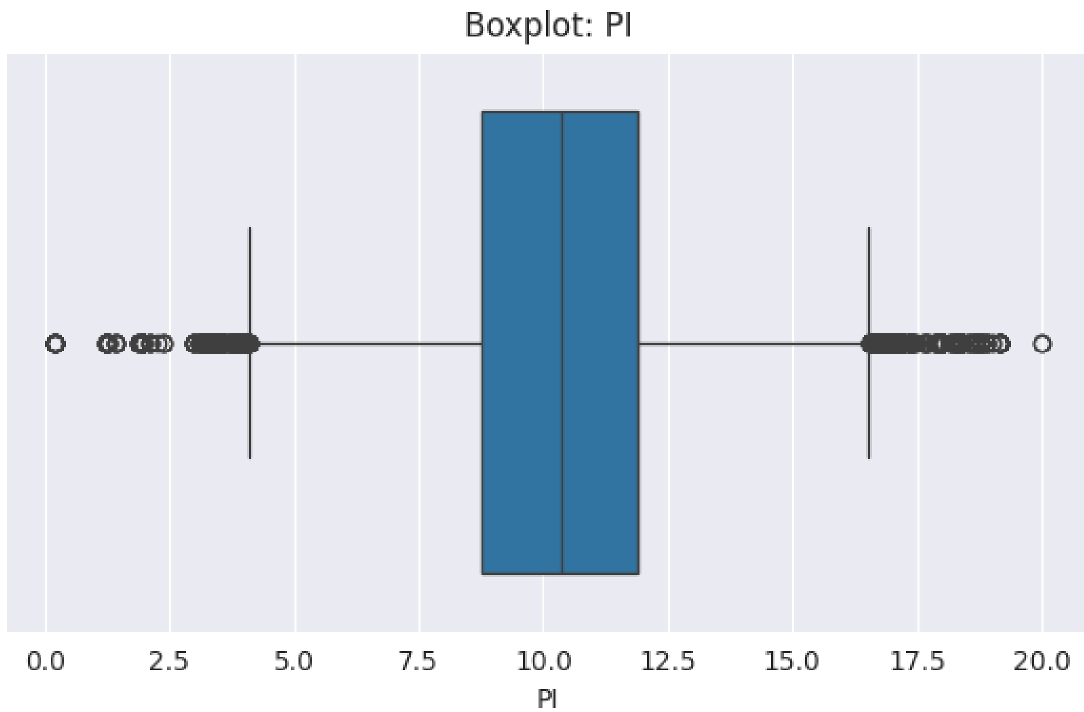
	Column	Outliers	Percent
13	CO	8460	12.63
2	PI	785	1.17
6	PRV	633	0.95
20	latitude	627	0.94
14	Blood Flow Index	602	0.90
0	SpO2	587	0.88
7	BP	560	0.84
15	PPG waveform features	549	0.82
21	longitude	544	0.81
17	Respiratory effort	468	0.70
9	Motion/Activity index	472	0.70
12	SV	439	0.66
8	Skin Temperature	432	0.65
11	Hb level	352	0.53
16	Signal Quality Index	351	0.52
10	PVI	344	0.51
5	FiO2	5	0.01
4	EtCO2	0	0.00
1	HR	0	0.00
3	RR	0	0.00
19	SNR	0	0.00
18	O2 extraction ratio	0	0.00
22	station_ID	0	0.00
23	user_id	0	0.00

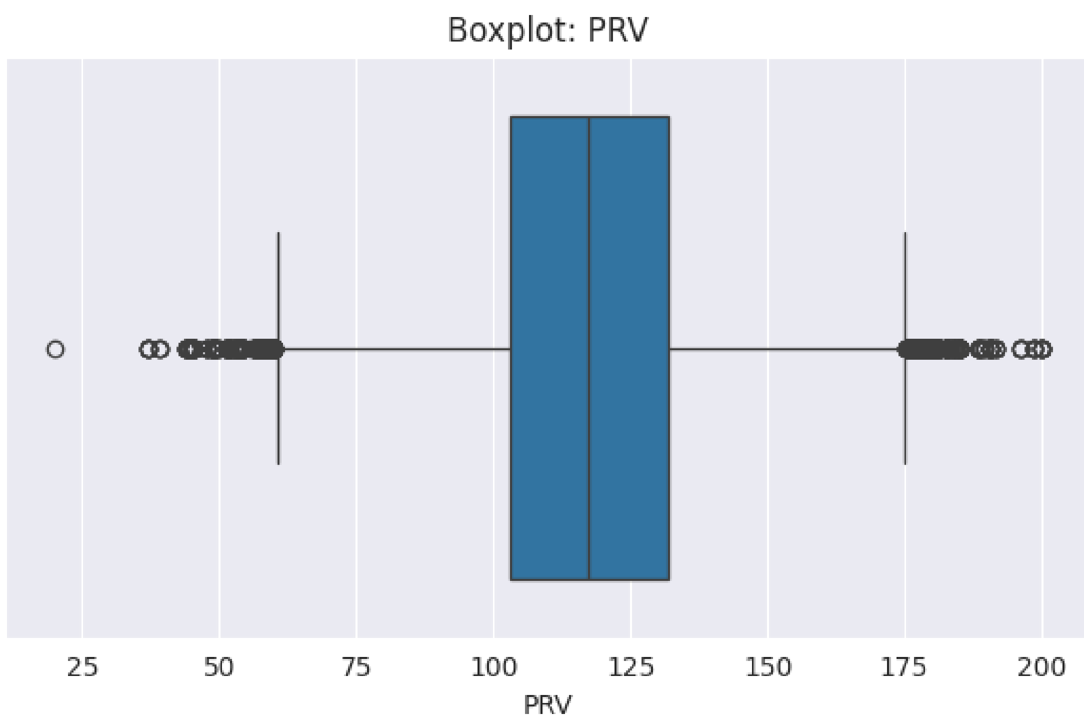
It's bad practice to analyze a atribute one by one so I created a list of numeric values that I can analyze at once. I calculated they're IQR based on dividing they're quantile (0.25 and 0.75) .If any value is less than lower value (<0.25) that it is added to list of outliers this process also include calculating if value is greater that upper bounder (>0.75).

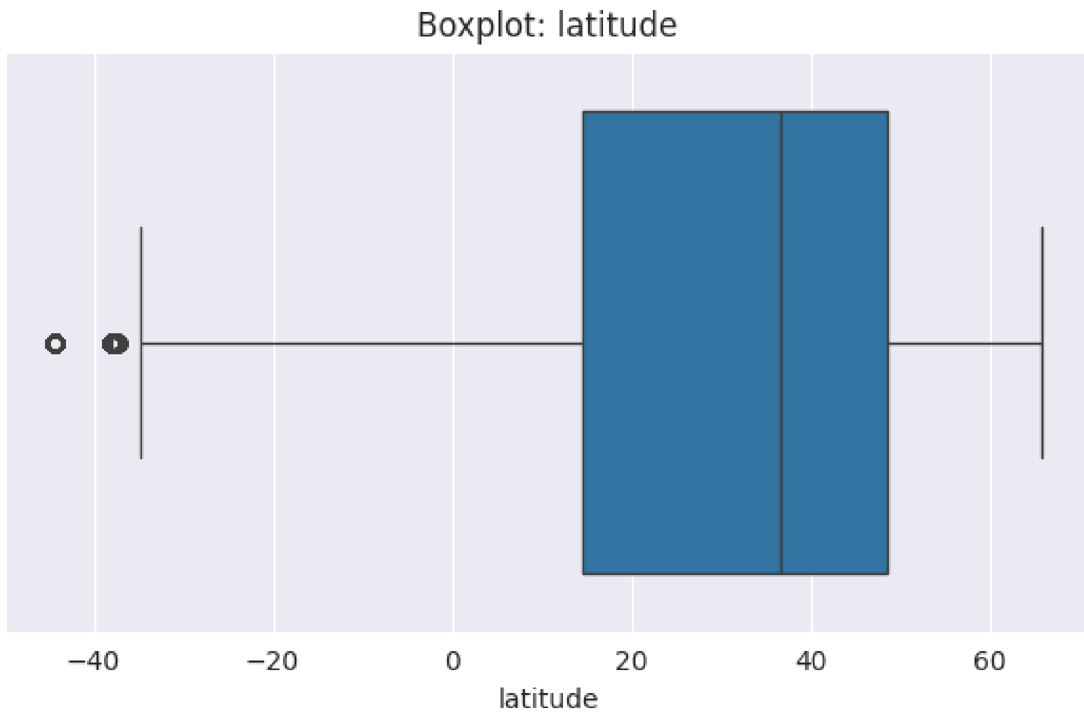
[405]:

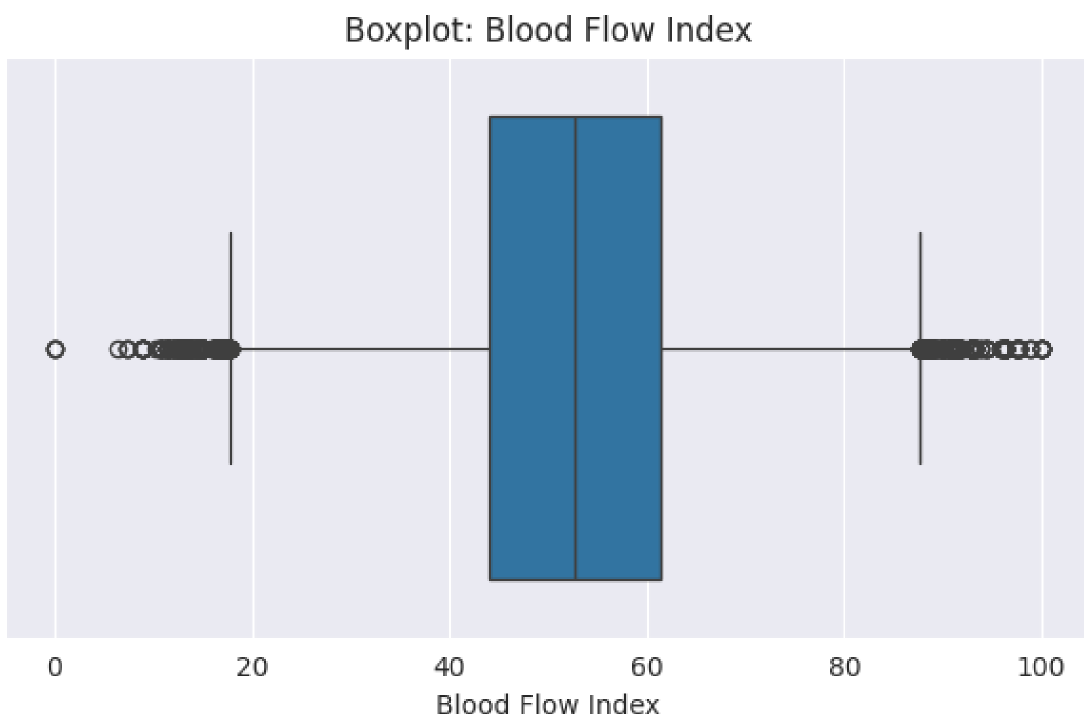
```python
suspects = iqr_df[ iqr_df['Outliers'] > 0 ]['Column'].head(6).tolist()

for col in suspects:
    plt.figure(figsize=(6,4))
    sns.boxplot(x=df[col])
    plt.title(f'Boxplot: {col}')
    plt.tight_layout()
    plt.show()
```

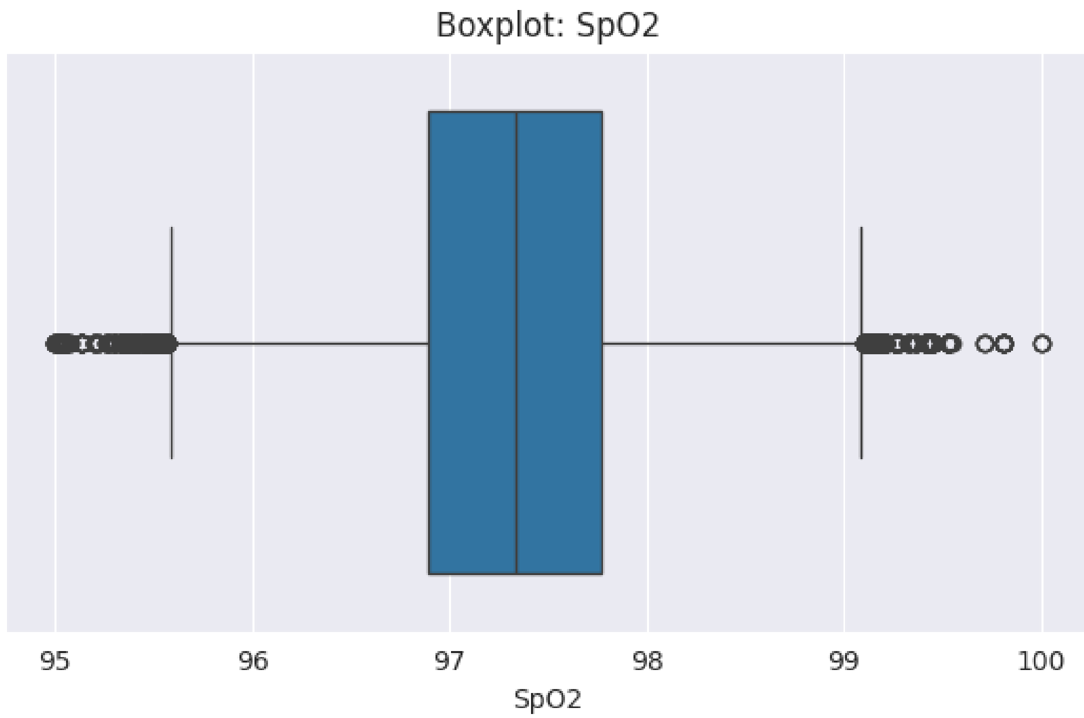



Boxplot: PI
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
PI


Boxplot: PRV
25
50
75
100
125
150
175
200
PRV


Boxplot: latitude
−40
−20
0
20
40
60
latitude


Boxplot: Blood Flow Index
0
20
40
60
80
100
Blood Flow Index


Boxplot: SpO2
95
96
97
98
99
100
SpO2

[ ]: