eda_sounds

March 21, 2021

1 Gender voice recognition - eksploracja danych

Michał Piasecki, Bartosz Siński

[2]:

```
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
import numpy as np
```

[3]:

```
df_voice = pd.read_csv("./src/gender_voice_dataset.csv")
df_attr = pd.read_csv("./src/attributes_gender_voice.csv")
```

2 Podstawowe informacje

[4]:

```
pd.options.display.max_colwidth = 200
df_attr
```

[4]:

```
        name   type  \
0   meanfreq  float
1         sd  float
2     median  float
3        Q25  float
4        Q75  float
5        IQR  float
6       skew  float
7       kurt  float
8     sp.ent  float
9        sfm  float
10      mode  float
11  centroid  float
12   meanfun  float
13    minfun  float
14    maxfun  float
15   meandom  float
16    mindom  float
17    maxdom  float
18   dfrange  float
```

```
19   modindx   float
20     label  string

                                                                     description
0
mean frequency (in kHz)
1
standard deviation of frequency
2
median frequency (in kHz)
3
first quantile (in kHz)
4
third quantile (in kHz)
5
interquantile range (in kHz)
6
skewness (see note in specprop description)
7
kurtosis (see note in specprop description)
8
spectral entropy
9
spectral flatness
10
mode frequency
11
frequency centroid (see specprop)
12
average of fundamental frequency measured across acoustic signal
13
minimum fundamental frequency measured across acoustic signal
14
maximum fundamental frequency measured across acoustic signal
15
average of dominant frequency measured across acoustic signal
16
minimum of dominant frequency measured across acoustic signal
17
maximum of dominant frequency measured across acoustic signal
18
range of dominant frequency measured across acoustic signal
19  modulation index. Calculated as the accumulated absolute difference between
adjacent measurements of fundamental frequencies divided by the frequency range
20
Predictor class, male or female
```

[5]:
```
df_voice.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3168 entries, 0 to 3167
Data columns (total 21 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   meanfreq  3168 non-null   float64
 1   sd        3168 non-null   float64
 2   median    3168 non-null   float64
 3   Q25       3168 non-null   float64
 4   Q75       3168 non-null   float64
 5   IQR       3168 non-null   float64
 6   skew      3168 non-null   float64
 7   kurt      3168 non-null   float64
 8   sp.ent    3168 non-null   float64
 9   sfm       3168 non-null   float64
 10  mode      3168 non-null   float64
 11  centroid  3168 non-null   float64
 12  meanfun   3168 non-null   float64
 13  minfun    3168 non-null   float64
 14  maxfun    3168 non-null   float64
 15  meandom   3168 non-null   float64
 16  mindom    3168 non-null   float64
 17  maxdom    3168 non-null   float64
 18  dfrange   3168 non-null   float64
 19  modindx   3168 non-null   float64
 20  label     3168 non-null   object 
dtypes: float64(20), object(1)
memory usage: 519.9+ KB
```

Jak widzimy, nasz zbiór nie zawiera brakujących danych.

[6]:
```
df_voice.describe()
```

[6]:
```
          meanfreq           sd       median          Q25          Q75  \
count  3168.000000  3168.000000  3168.000000  3168.000000  3168.000000   
mean      0.180907     0.057126     0.185621     0.140456     0.224765   
std       0.029918     0.016652     0.036360     0.048680     0.023639   
min       0.039363     0.018363     0.010975     0.000229     0.042946   
25%       0.163662     0.041954     0.169593     0.111087     0.208747   
50%       0.184838     0.059155     0.190032     0.140286     0.225684   
75%       0.199146     0.067020     0.210618     0.175939     0.243660   
max       0.251124     0.115273     0.261224     0.247347     0.273469   

               IQR         skew         kurt       sp.ent          sfm  \
count  3168.000000  3168.000000  3168.000000  3168.000000  3168.000000   
mean      0.084309     3.140168    36.568461     0.895127     0.408216   
```

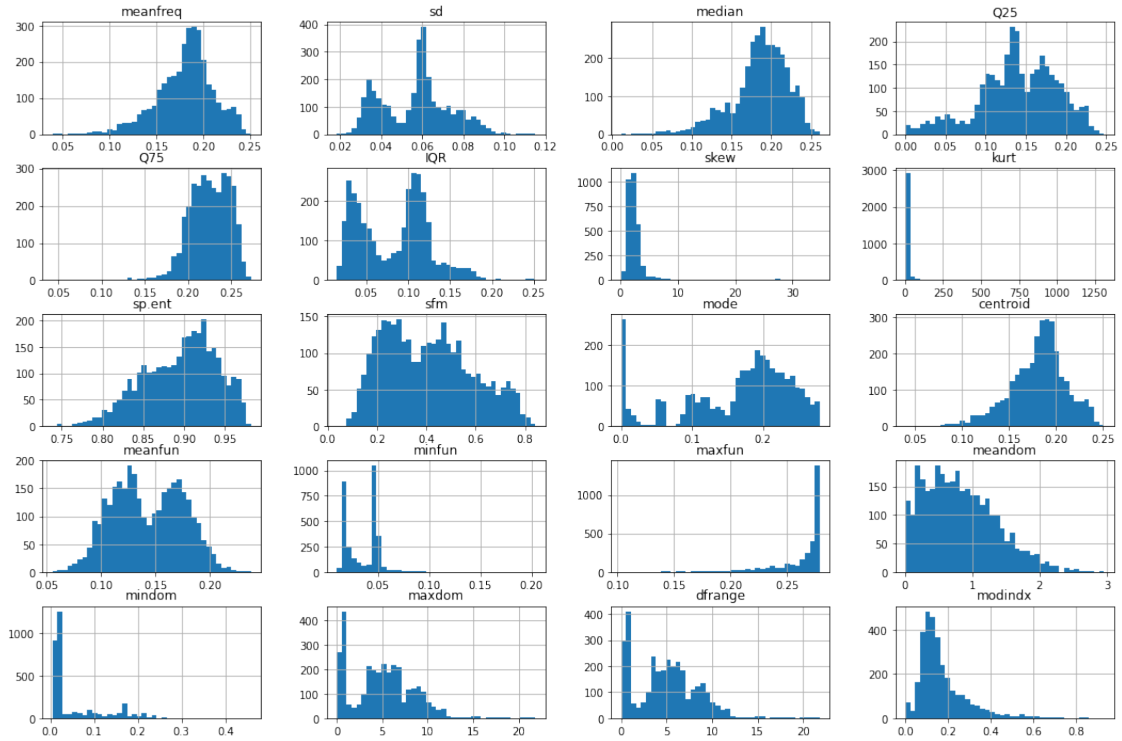
```
std       0.042783     4.240529   134.928661     0.044980     0.177521
min       0.014558     0.141735     2.068455     0.738651     0.036876
25%       0.042560     1.649569     5.669547     0.861811     0.258041
50%       0.094280     2.197101     8.318463     0.901767     0.396335
75%       0.114175     2.931694    13.648905     0.928713     0.533676
max       0.252225    34.725453  1309.612887     0.981997     0.842936

                mode     centroid      meanfun       minfun       maxfun  \
count  3168.000000  3168.000000  3168.000000  3168.000000  3168.000000
mean      0.165282     0.180907     0.142807     0.036802     0.258842
std       0.077203     0.029918     0.032304     0.019220     0.030077
min       0.000000     0.039363     0.055565     0.009775     0.103093
25%       0.118016     0.163662     0.116998     0.018223     0.253968
50%       0.186599     0.184838     0.140519     0.046110     0.271186
75%       0.221104     0.199146     0.169581     0.047904     0.277457
max       0.280000     0.251124     0.237636     0.204082     0.279114

             meandom       mindom       maxdom      dfrange      modindx
count  3168.000000  3168.000000  3168.000000  3168.000000  3168.000000
mean      0.829211     0.052647     5.047277     4.994630     0.173752
std       0.525205     0.063299     3.521157     3.520039     0.119454
min       0.007812     0.004883     0.007812     0.000000     0.000000
25%       0.419828     0.007812     2.070312     2.044922     0.099766
50%       0.765795     0.023438     4.992188     4.945312     0.139357
75%       1.177166     0.070312     7.007812     6.992188     0.209183
max       2.957682     0.458984    21.867188    21.843750     0.932374
```

[7]:

```
df_voice.drop(["label"], axis=1).hist(bins = 40, figsize=(18, 12))
plt.show()
```



[8]:

```
voice_grouped = df_voice.groupby(by="label")
voice_grouped['meanfreq'].count()
```

[8]:

```
label
female    1584
male      1584
Name: meanfreq, dtype: int64
```

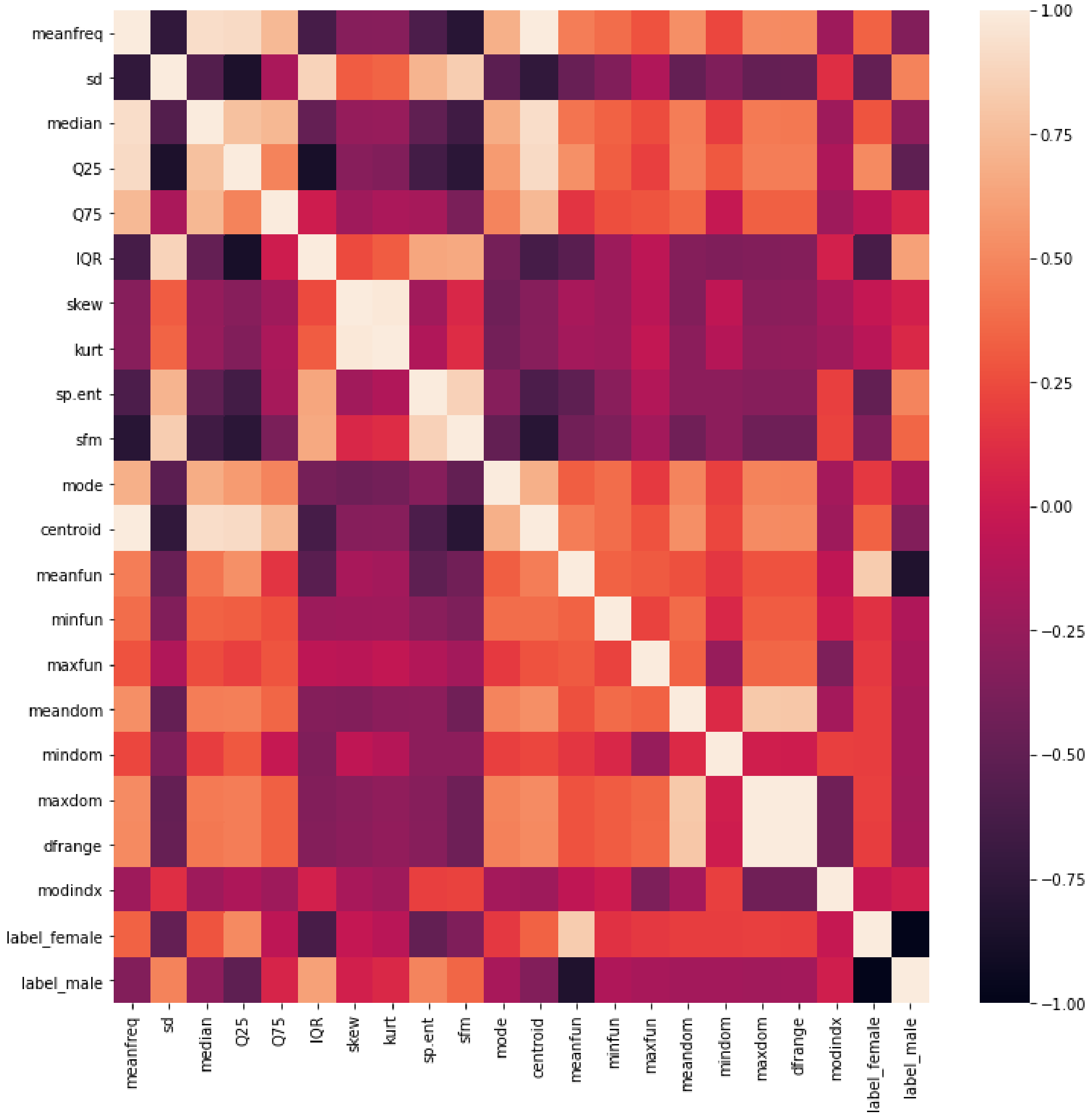
[9]:

```
df_voice=pd.get_dummies(df_voice)
```

3 Korelacje i zależności zmiennych

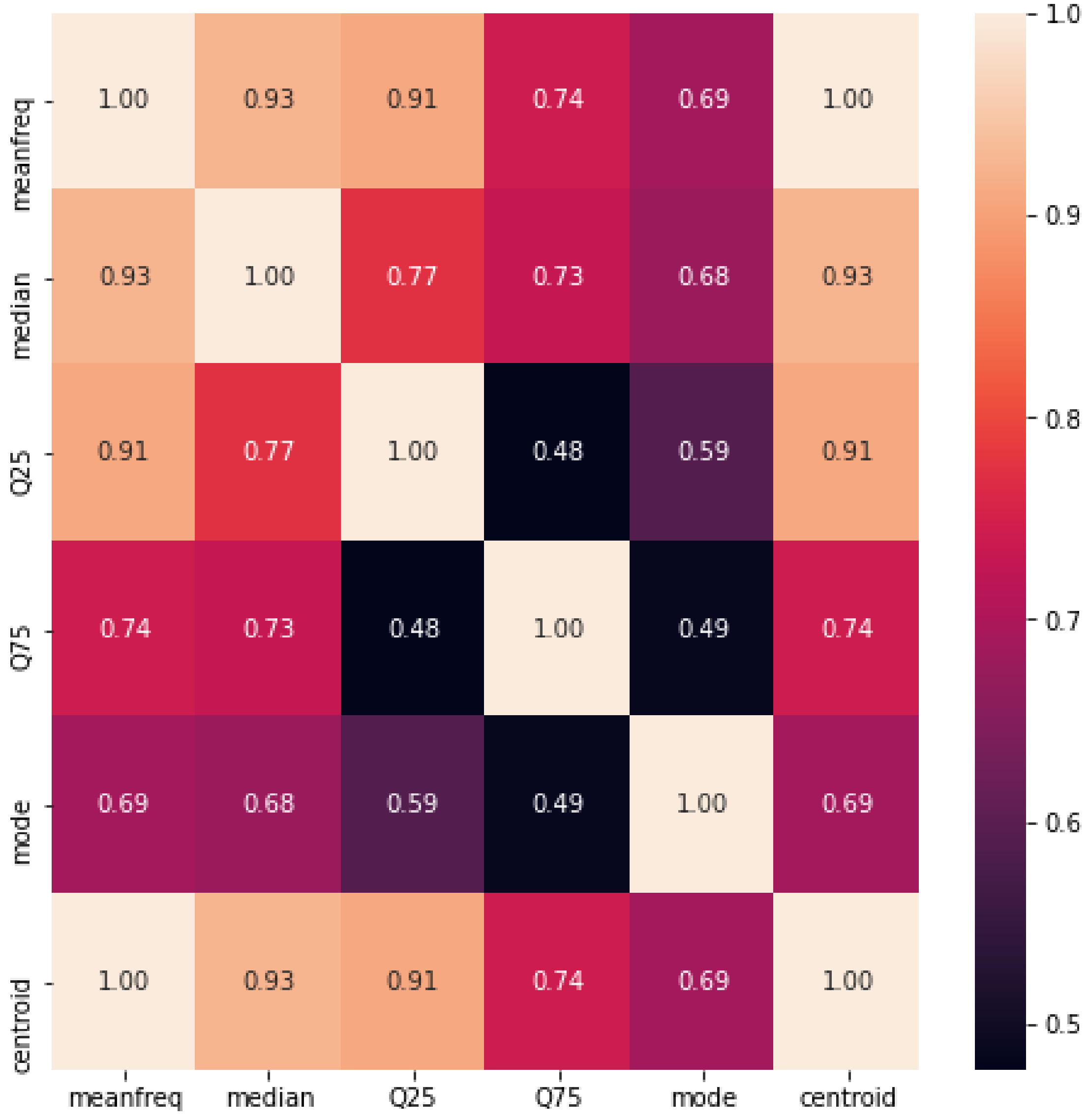
[10]:

```
plt.figure(figsize=(12,12))
sns.heatmap(df_voice.corr())
plt.show()
```



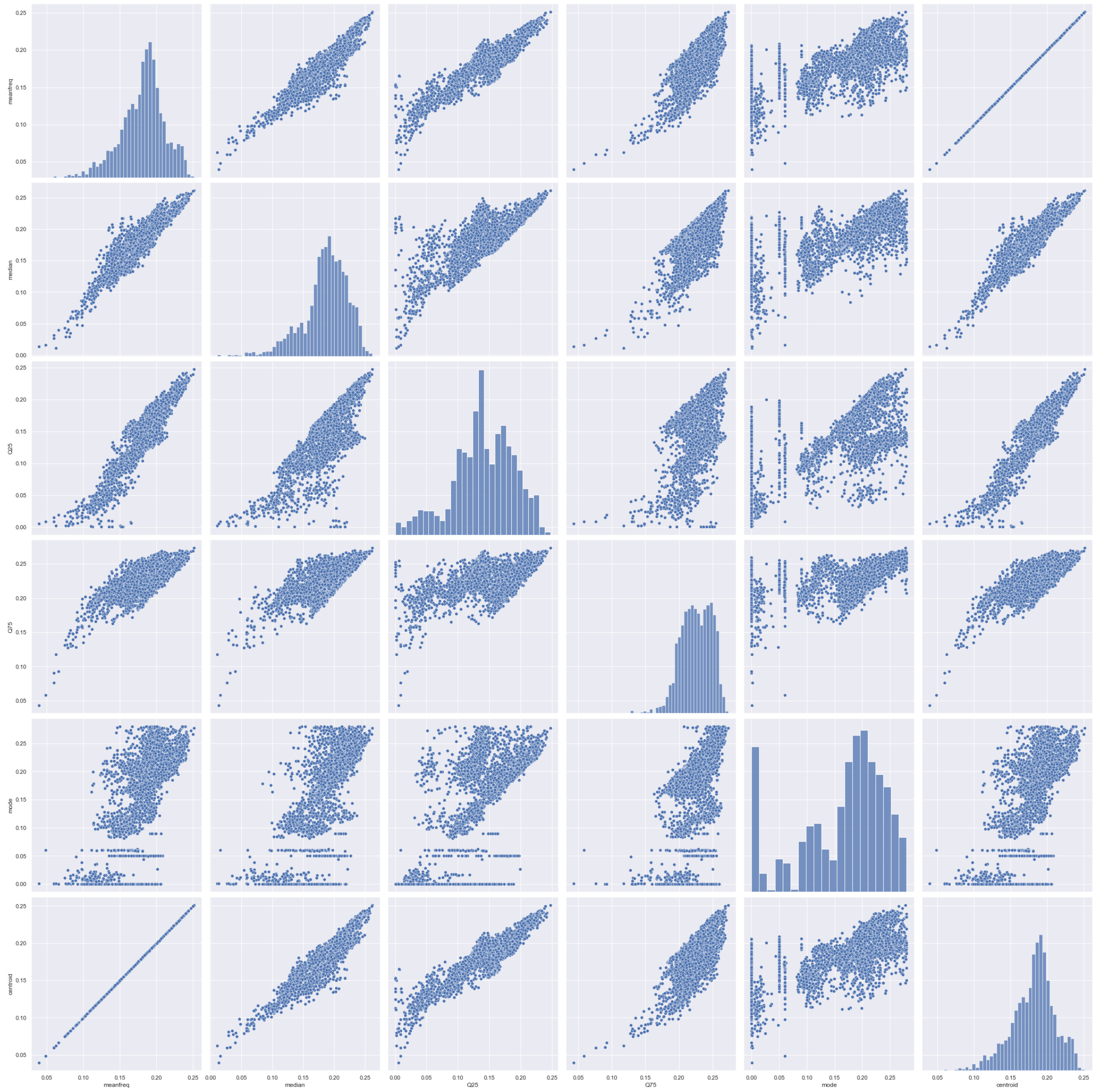
Przyjrzyjmy się bliżej *meanfreq,median,Q25,Q75,mode,centroid*, które wydają się byc ze soba najlepiej skorelowane.

[11]:
```
plt.figure(figsize=(8,8))
sns.heatmap(df_voice[['meanfreq','median','Q25','Q75','mode','centroid']].
↪corr(),annot=True, annot_kws={'size': 10}, fmt='.2f')
plt.show()
```



[27]:

```
sns.set()
sns.pairplot(df_voice[['meanfreq','median','Q25','Q75','mode','centroid']],␣
 ↪height = 5)
plt.show();
```



Już teraz widzimy, że niektórych zmiennych będziemy mogli nie uwzględniać przy budowie naszego modelu.

4 Zmienne najlepiej skorelowane z targetem

[12]:
```
voice_corr = df_voice.corr()[['label_male','label_female']]
voice_corr.iloc[(-voice_corr['label_male'].abs()).argsort()]
```

[12]:
```
              label_male  label_female
label_male      1.000000     -1.000000
label_female   -1.000000      1.000000
meanfun        -0.833921      0.833921
```

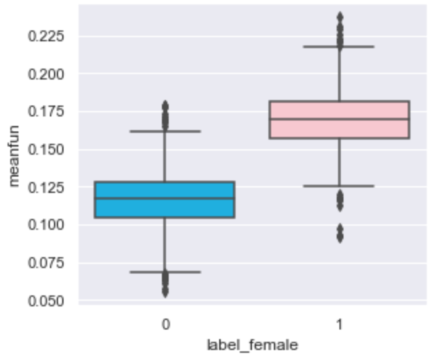
```
IQR          0.618916  -0.618916
Q25         -0.511455   0.511455
sp.ent       0.490552  -0.490552
sd           0.479539  -0.479539
sfm          0.357499  -0.357499
centroid    -0.337415   0.337415
meanfreq    -0.337415   0.337415
median      -0.283919   0.283919
maxdom      -0.195657   0.195657
mindom      -0.194974   0.194974
dfrange     -0.192213   0.192213
meandom     -0.191067   0.191067
mode        -0.171775   0.171775
maxfun      -0.166461   0.166461
minfun      -0.136692   0.136692
kurt         0.087195  -0.087195
Q75          0.066906  -0.066906
skew         0.036627  -0.036627
modindx      0.030801  -0.030801
```

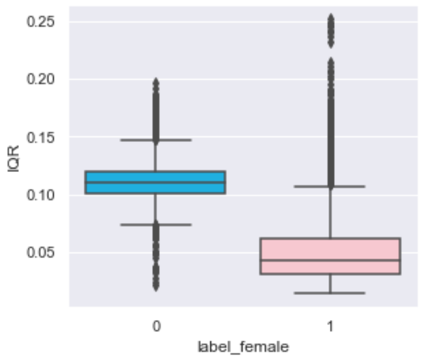
[29]:

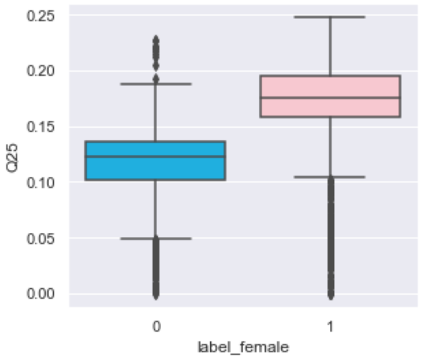
```
fig, axs = plt.subplots(nrows=5,figsize=(5,20))
fig.tight_layout(pad=3.0)
sns.boxplot(data=df_voice,x="label_female",y="meanfun",ax=axs[0], palette =␣
 ↪["deepskyblue","pink"])
sns.boxplot(data=df_voice,x="label_female",y="IQR",ax=axs[1], palette =␣
 ↪["deepskyblue","pink"])
sns.boxplot(data=df_voice,x="label_female",y="Q25",ax=axs[2], palette =␣
 ↪["deepskyblue","pink"])
sns.boxplot(data=df_voice,x="label_female",y="sp.ent",ax=axs[3], palette =␣
 ↪["deepskyblue","pink"])
sns.boxplot(data=df_voice,x="label_female",y="sd",ax=axs[4], palette =␣
 ↪["deepskyblue","pink"])
```

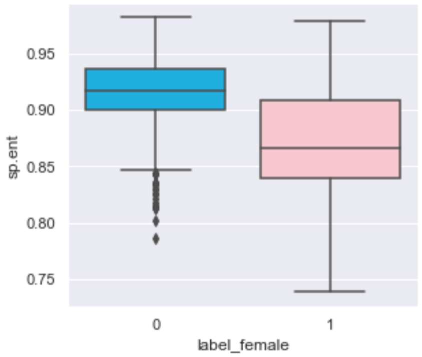
[29]:

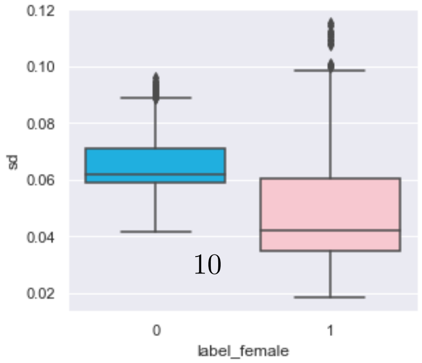
```
<AxesSubplot:xlabel='label_female', ylabel='sd'>
```


meanfun
0.225
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0
1
label_female


IQR
0.25
0.20
0.15
0.10
0.05
0
1
label_female


Q25
0.25
0.20
0.15
0.10
0.05
0.00
0
1
label_female

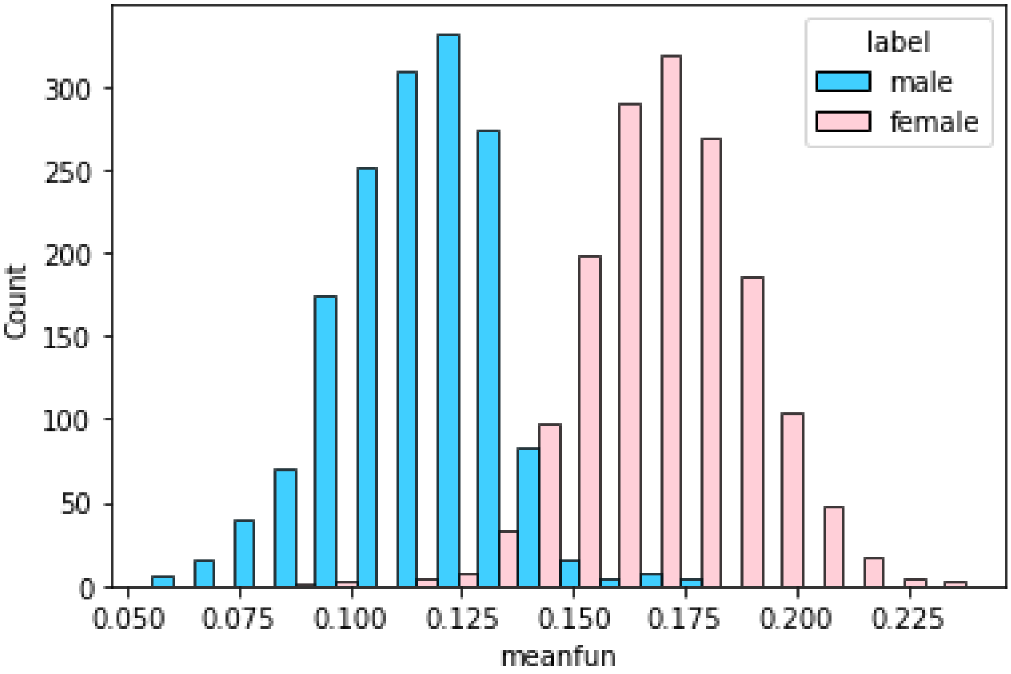

sp.ent
0.95
0.90
0.85
0.80
0.75
0
1
label_female

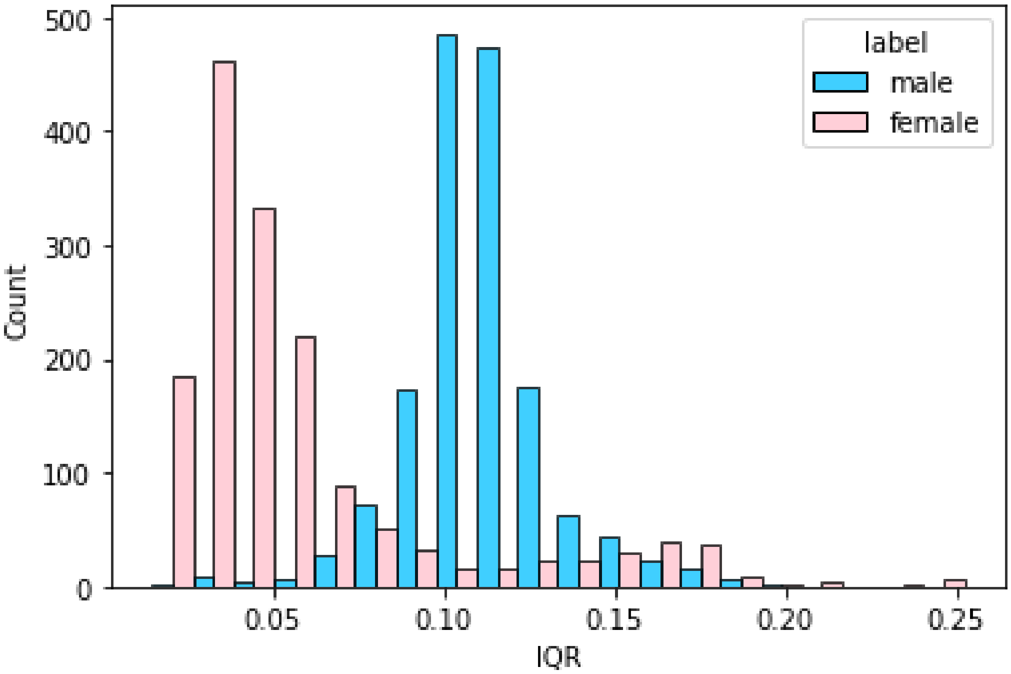

sd
0.12
0.10
0.08
0.06
0.04
0.02
0
1
label_female

5 Różnice pomiędzy kobietami i mężczyznami

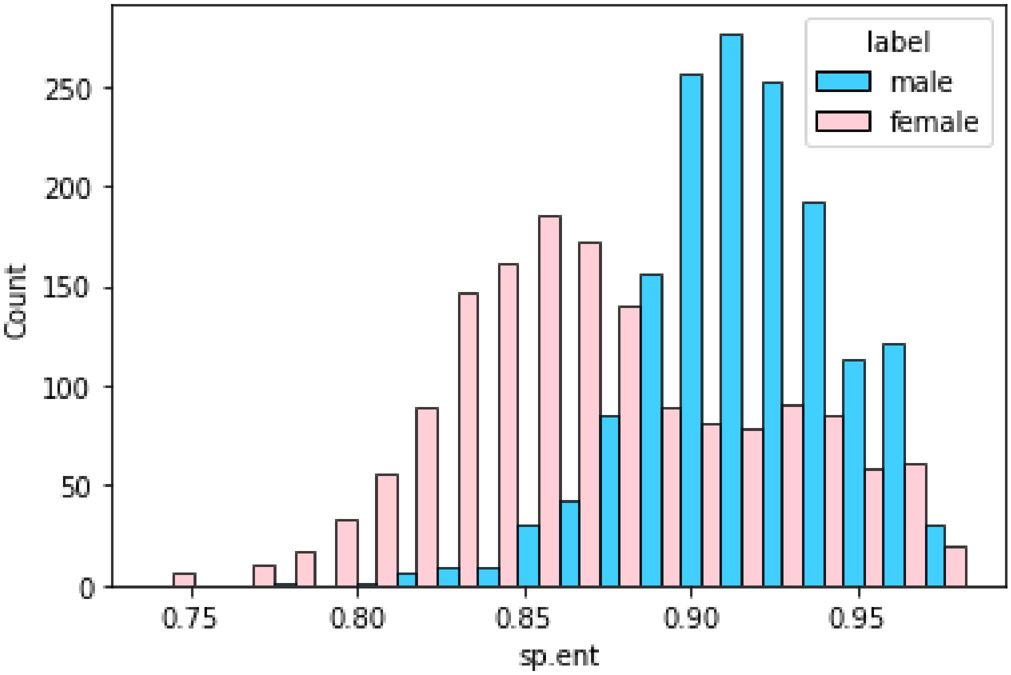
[21]:

```
df_voice1 = pd.read_csv("./src/gender_voice_dataset.csv")
females = df_voice1[df_voice1.label == "female"]
males = df_voice1[df_voice1.label == "male"]
columns = df_voice1[['meanfun','IQR','sp.ent','Q25','sd']].columns
columns = columns.tolist()
for column in columns:
    sns.histplot(data = df_voice1, x = column, hue = "label", bins = 20,␣
 ↪multiple = "dodge", palette = ["deepskyblue","pink"])
    plt.show()
```

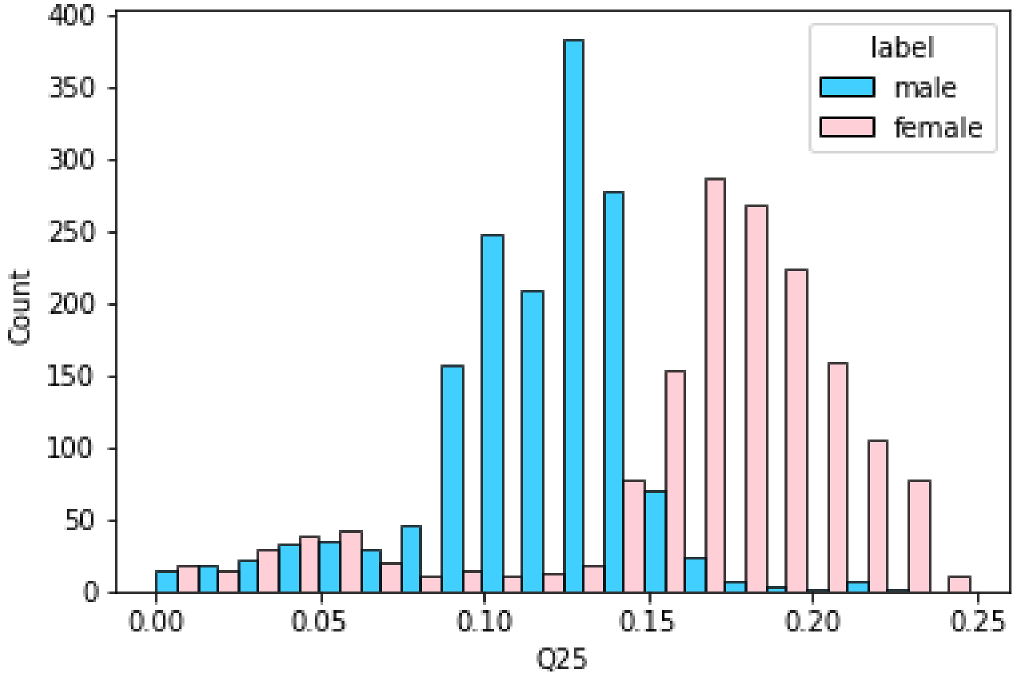


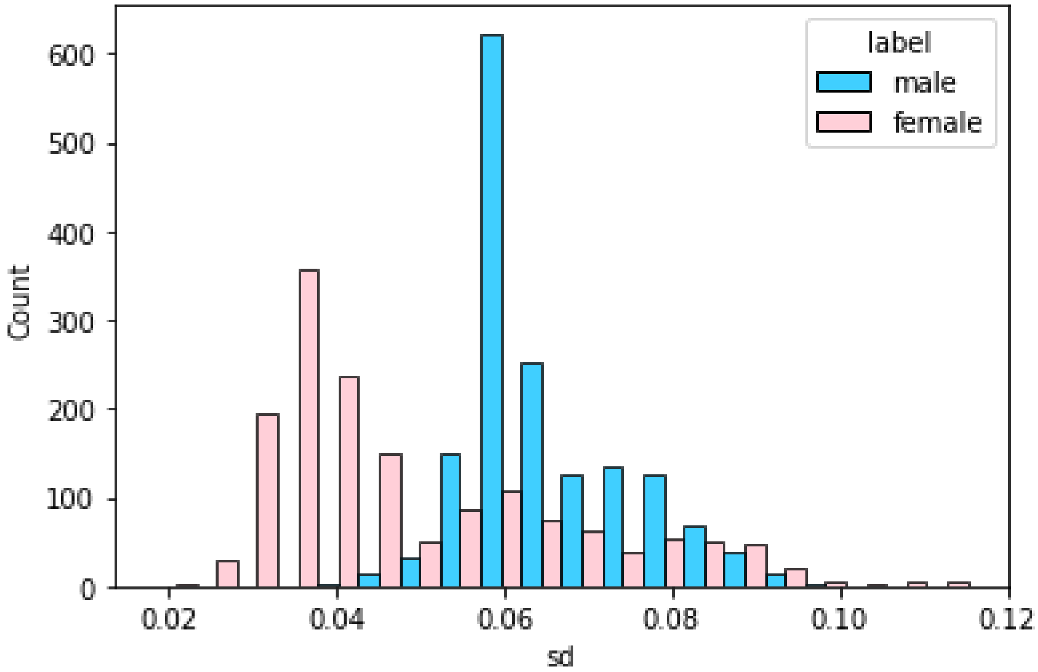


Count
500
400
300
200
100
0
0.05
0.10
0.15
0.20
0.25
IQR
label
male
female



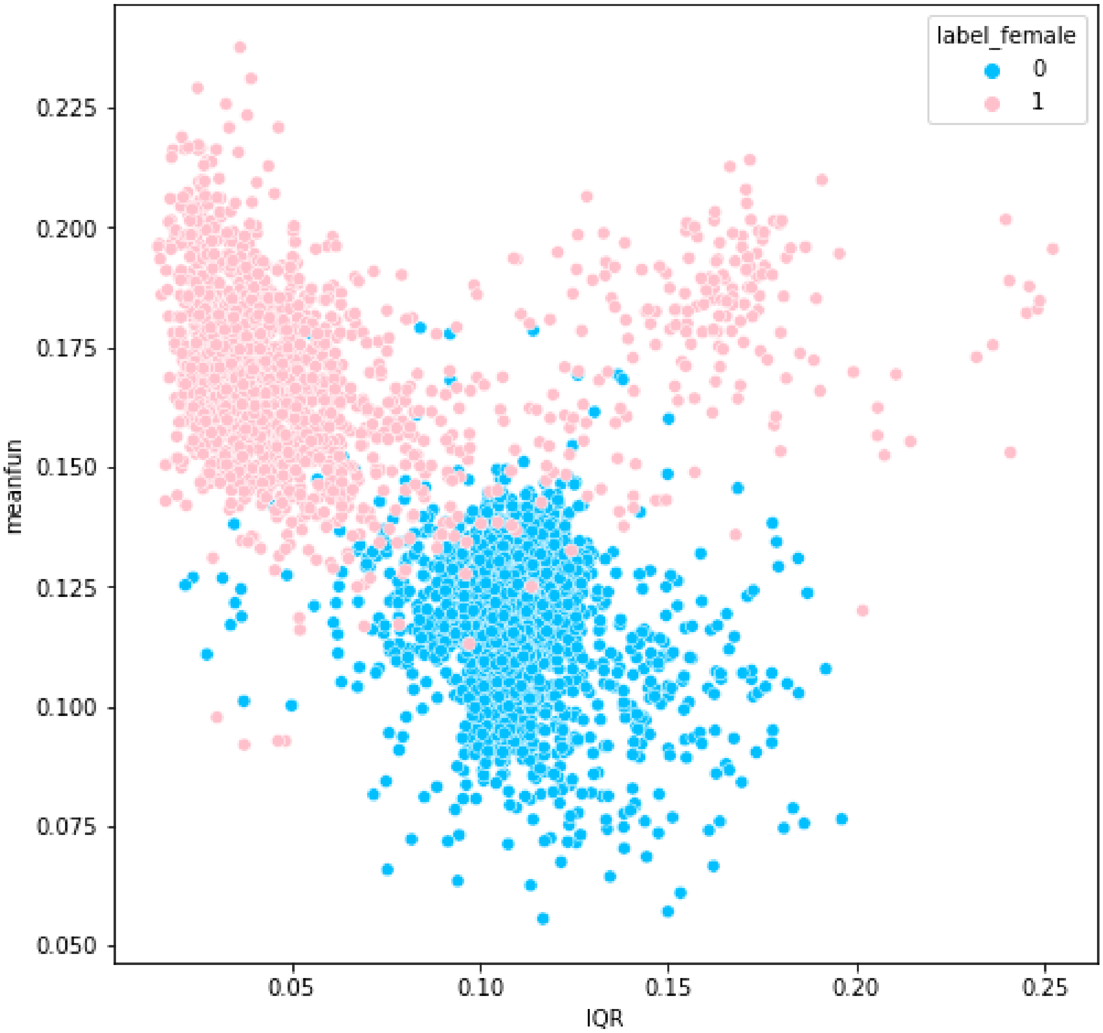
Count
250
200
150
100
50
0
0.75
0.80
0.85
0.90
0.95
sp.ent
label
male
female



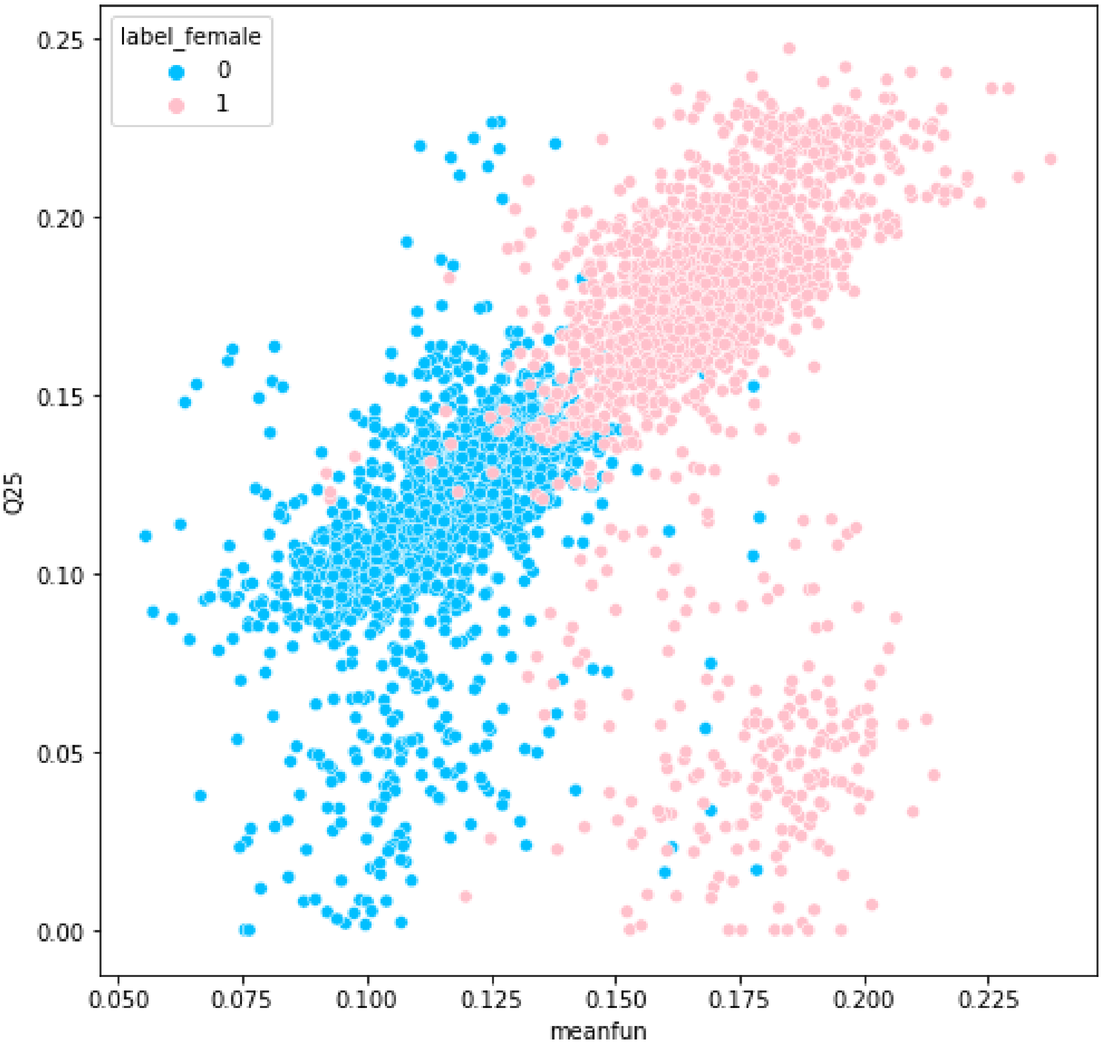


[22]:
```
fig, ax = plt.subplots(figsize=(8,8))
```

```
sns.scatterplot(data=df_voice,x="IQR",y="meanfun",hue="label_female", palette =␣
↪["deepskyblue","pink"])
plt.show()
```



```
[23]: fig, ax = plt.subplots(figsize=(8,8))
sns.scatterplot(data=df_voice,x="meanfun",y="Q25",hue="label_female", palette =␣
↪["deepskyblue","pink"])
plt.show()
```


label_female
0
1
Q25
0.25
0.20
0.15
0.10
0.05
0.00
meanfun
0.050
0.075
0.100
0.125
0.150
0.175
0.200
0.225