

Gender voice

Wczytanie odpowiednich bibliotek

```
library("tuneR")
library("seewave")
library("reticulate")
use_python("C:\Users\01143557\AppData\Local\Programs\Python\Python39\python.exe", required = TRUE)
library(signal, warn.conflicts = F, quietly = T)
library(oce, warn.conflicts = F, quietly = T)
```

```
## Linking to GEOS 3.8.0, GDAL 3.0.4, PROJ 6.3.1
```

Stworzenie funkcji odpowiadajacych za dalsze wykresy oraz za stworzenie finalnej ramki danych

```
meski <- readwave("sample.wav")
zenski <- readwave("mamiko.wav")

filter <- function(wave){
  wave <- ffilt(wave, from=0, to=280)
  wave
}

duration <- function(sample){
  dur = length(sample@left)/sample@rate
  dur
}

sample_rate <- function(sample){
  fs = sample@samp.rate
  fs
}

frequency_plot <- function(sample){
  plot(sample@left[30700:31500], type = "l", main = "Sample",
       xlab = "Time", ylab = "Frequency")
}

waveform <- function(sample){
  sl <- sample@left
  sl = sl - mean(sl)
  plot(sl, type = 'l', xlab = 'Samples', ylab = 'Amplitude')
}

spectrogram <- function(sample){
  sl <- sample@left
  spec = spectrogram(x = sl,
                     n = 1024,
                     Fs = sample_rate(sample),
                     window = 256,
                     overlap = 128)

  P = abs(spec$S)
  P = P/max(P)
  P = 10*log10(P)
  t = spec$t

  image(x = t, y = spec$f, z = P, col = oce.colors$viridis,
        ylab = 'Frequency [Hz]', xlab = 'Time [s]', drawPalette = T, decimate = F)
}

to_data_frame <- function(sample){
  filtered_glos <- filter(sample)
  specprop(meancor(filtered_glos, f = 44100), f=44100)
  minfun <- min(fund(sample, fmax=280)[,2], na.rm=TRUE)
  meanfun <- mean(fund(sample, fmax=280)[,2], na.rm=TRUE)
  df <- data.frame(specprop(meancor(filtered_glos, f = 44100), f=44100), "meanfun" <- meanfun,
  n, "minfun" <- minfun)
  colnames(df)[1] <- c("meanfreq")
  colnames(df)[16] <- c("meanfun", "minfun")
  colnames(df)[9] <- c("centroid")
  colnames(df)[13] <- c("sp.ent")
  df
}
```

Dlugosc naszego dzwieku:

```
meski:
duration(meski)
```

```
## [1] 7.692222
```

```
zenski:
duration(zenski)
```

```
## [1] 9.080499
```

Liczba sampli na sekunde mierzona w Hz:

```
meski:
sample_rate(meski)
```

```
## [1] 44100
```

```
zenski:
sample_rate(zenski)
```

```
## [1] 44100
```

Wykres zaleznosci kolejnego sampla od amplitudy:

```
meski:
waveform(meski)
```

```
zenski:
waveform(zenski)
```

Wykres zaleznosci kolejnego sampla od czestotliwosci:

```
meski:
frequency_plot(meski)
```

```
zenski:
frequency_plot(zenski)
```

Histogramy przedstawiajace rozklad kazdej zmiennej objasniajacej

```
voice_df.hist(figsize=(18, 12), bins=30);
plt.show()
```

Heatmap przedstawiajaca korelacje pomiedzy zmiennymi

```
sns.set(font_scale=2.5)
fig, ax = plt.subplots(figsize=(30, 25))
sns.heatmap(round(abs(cov(voice_df.corr())), 2), cmap="cividis", annot = True, vmax=1, linewidths=1, linecolor='white', square=True, fontweight="bold");
plt.show()
```

```
sns.set(font_size=1)
```

Pairplot dla czterech najbardziej charakterystycznych zmiennych

```
g = sns.pairplot(voice_df[['sd', 'IQR', 'Q25', 'label']], hue = 'label', height = 3, plot_kws=[{'alpha': 0.6}])
g.map_lower(sns.kdeplot, levels=4, color="#2ca02c")
plt.show()
```

Ridgeplot dla dwóch zmiennych, których nie udalo nam sie zreprodukowac

```
def ridgeplot(voice_df, label1):
    rp = sns.FacetGrid(voice_df, row="label", hue="label", aspect=5, height=1.25)
    rp.map(sns.kdeplot, label1, bw_adjust=.5, clip_on=False, fill=True, alpha=.1, linewidth=.5)
    rp.map(sns.kdeplot, label1, bw_adjust=.5, clip_on=False, color="w", lw=2, bw_adjust=.5)
    def ab(x, color):
        ax = plt.gca()
        ax.text(x, 0, color, color=color, ha="left", va="center", transform=ax.get_xaxis_transform())
    ormax_transpose()
    rp.map(label1, label1, fontweight="bold", color=color, ha="left", va="center", transform=ax.get_xaxis_transform())
    rp.set_titles("")
    rp.despine(bottom=True, left=True)
    rp.fig.tight_layout()
```

```
cols = voice_df.columns.values[0:18].tolist
```

```
ridgeplot(voice_df, "dfrange")
plt.show()
```

```
ridgeplot(voice_df, "modindx")
plt.show()
```

Jak widac zmienne te sa niemal identyczne dla glosu zejskiego jak i dla glosu meskiego. Oznacza to, ze w przyszlosci bedziemy mogli je poninac w naszym modelu, bez znaczącego wpływu na jego zdolnosci predykcyjne.