Federico Alvetreti
Ioan Corrias
Lucia Dicunta
Leonardo Di Nino

TherMike

Statistical Learning

Final Project

TherMike - Hearing hot loud

Data Collection

Feature Engineering

Statistical Framework

Failures and Successes

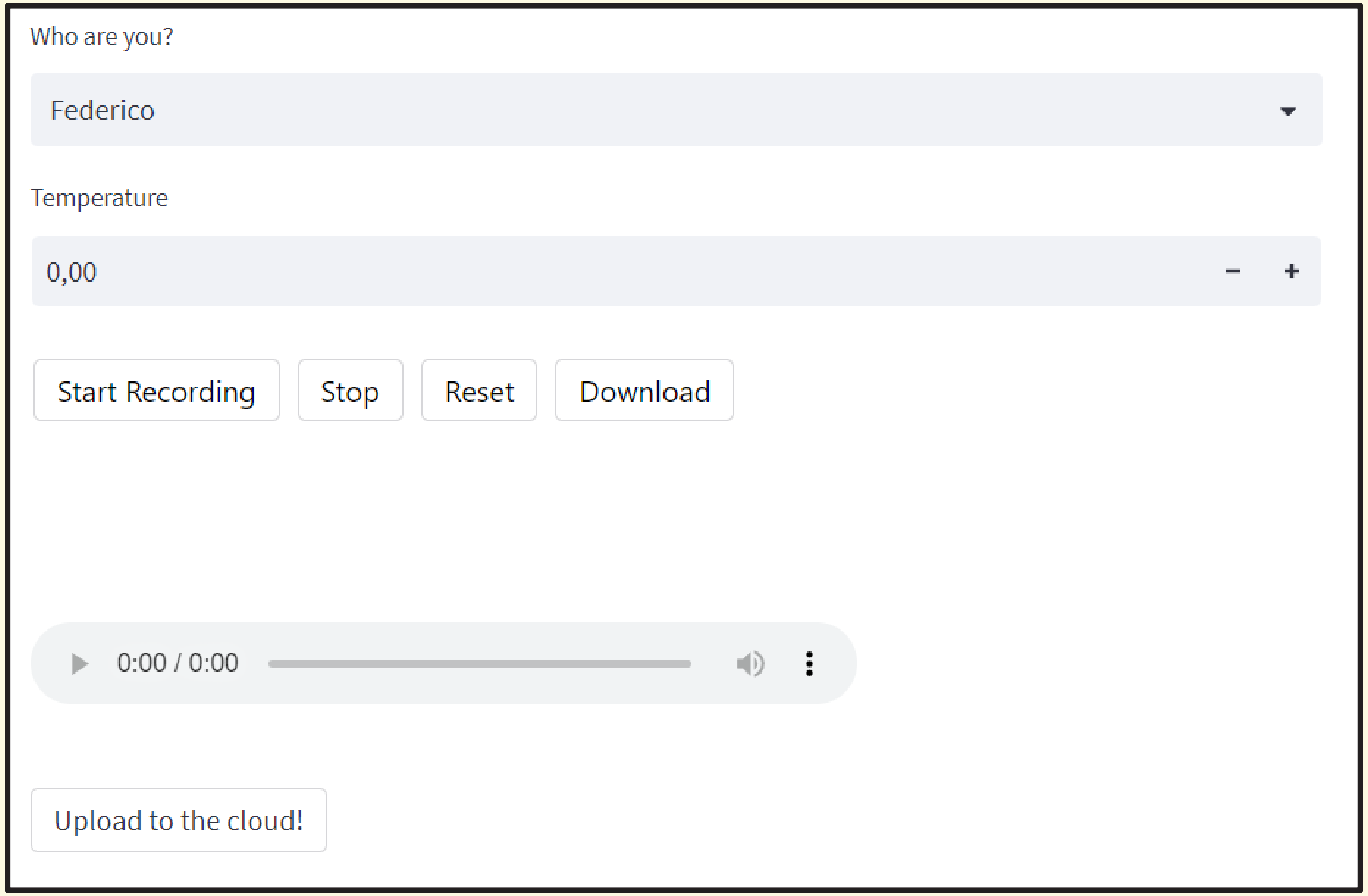
TherMike- Hearing hot loud

We were inspired by Mr.Brutti sound experiments and we decided to dig deeper in the problem. TherMike arises as the opportunity to deploy a machine learning model that can interpret or explain a phenomenon that is actually well known in neurosciences and psychoacoustics: how is it possible for our ear and brain to collect and decode information that are apparently unrelated with what we are actually hearing?

So, we decided not to build a model “simply” capable of classifying if the water is cold or hot given its splashing sound: *we wanted to build a real thermometer able to listen to temperature.*

This makes our problem a regression one: given a pouring sound, we want to predict the temperature. And now lots of questions arises!

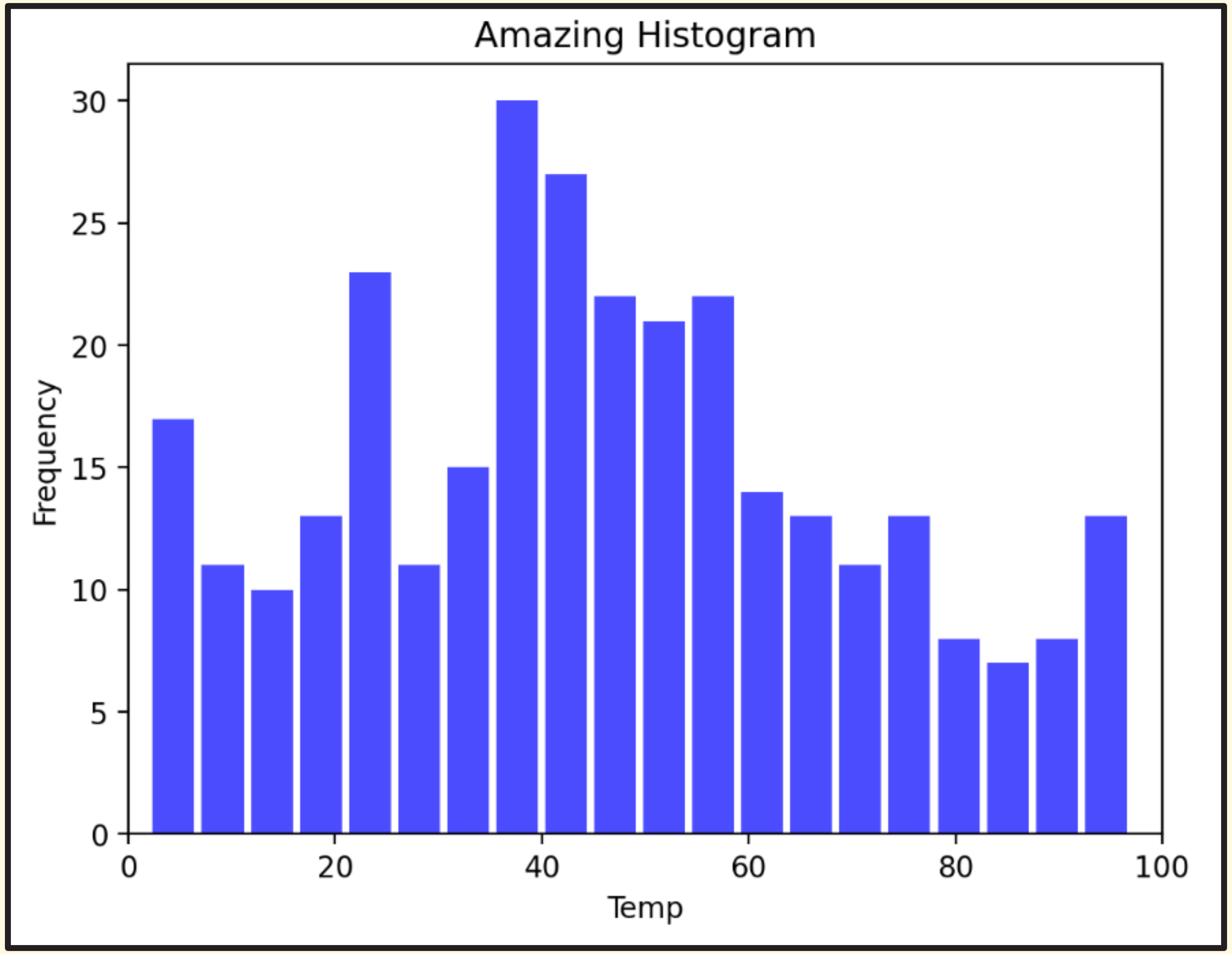
Data Collection



Strictly from a practical perspective, Ioan built an app on StreamLit to allow each of us to collect data on our own. The data stream was processed and stored on a shared Google Drive and each file has been labeled with the author name and the detected temperature.

Data Collection

A **histogram** updating after each update shows the **distribution of the labels**: this was an implementation that has been inserted in order to keep us from building an unbalanced dataset.

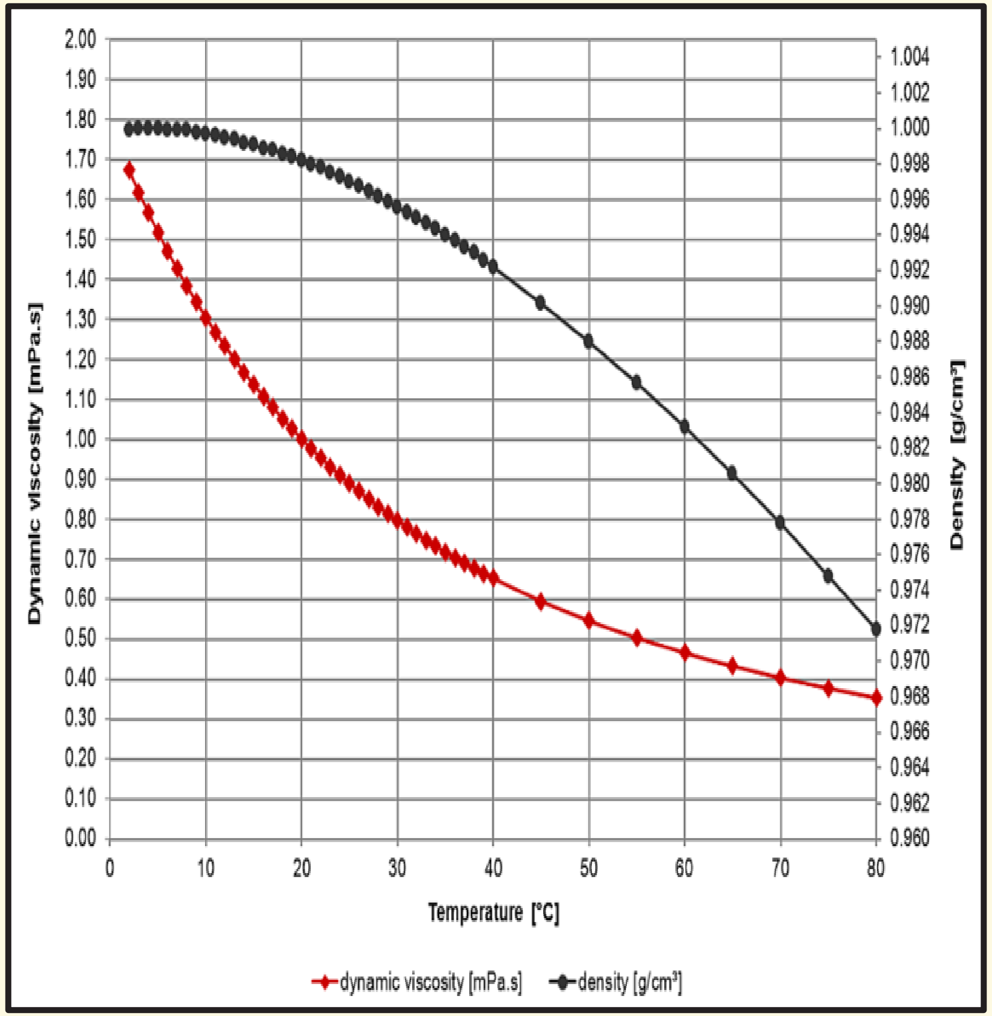


The experiment was easy in its guiding lines: pour some water, record its splash, detect the temperature. We just decided for some parameters to be similar in order to gather homogenenous data:

- **We used two metal vessels;**
- **We poured water from no more than 10 cm away;**
- **We detected the temperature right after pouring the water;**
- **We recorded audio lasting between 5 and 7 seconds.**

In the end we collected 309 audio samples on our own. Additionally, we also used as benchmark dataset the one from the similar study we read about, that consisted of 333 audio samples.

Feature Engineering

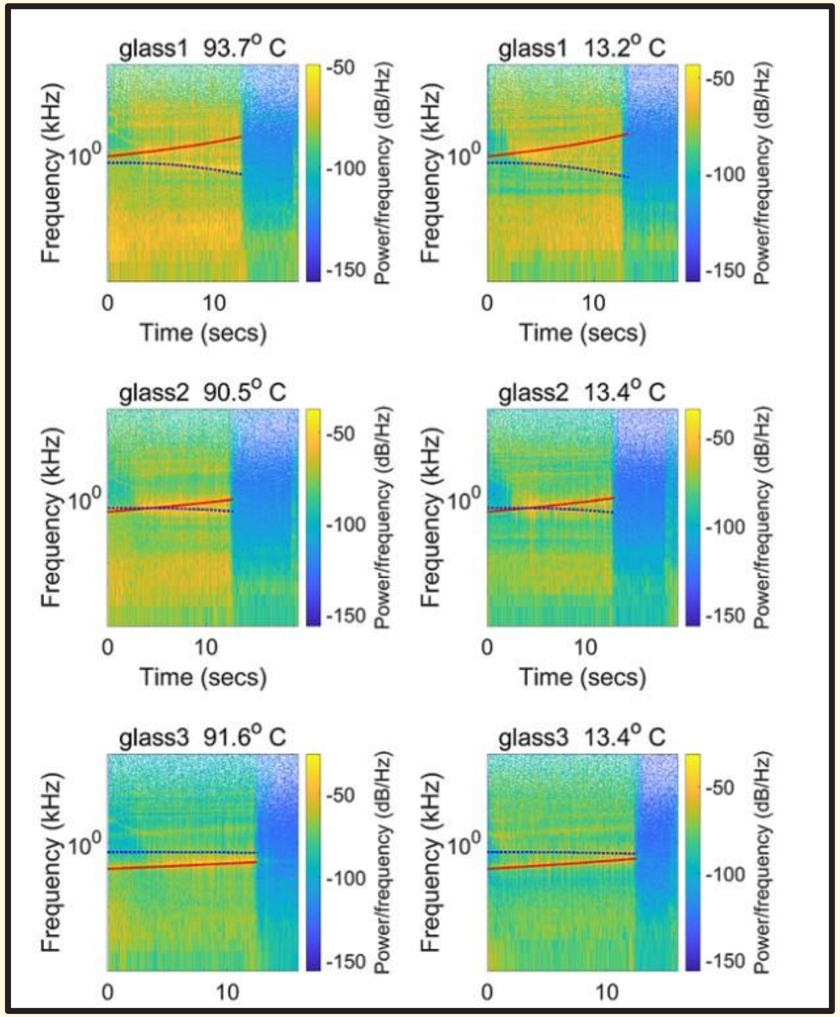


There clearly is an empirical evidence in discerning between hot and cold water: the viscosity and density of the liquid change with respect to the temperature. This is a good starting point, but at the same time it is not enough to explain acoustic phenomena.

Feature Engineering

Infact, when comparing spectrograms of audio recording of liquid poured at the different temperature there is no apparent difference between the two scenarios.

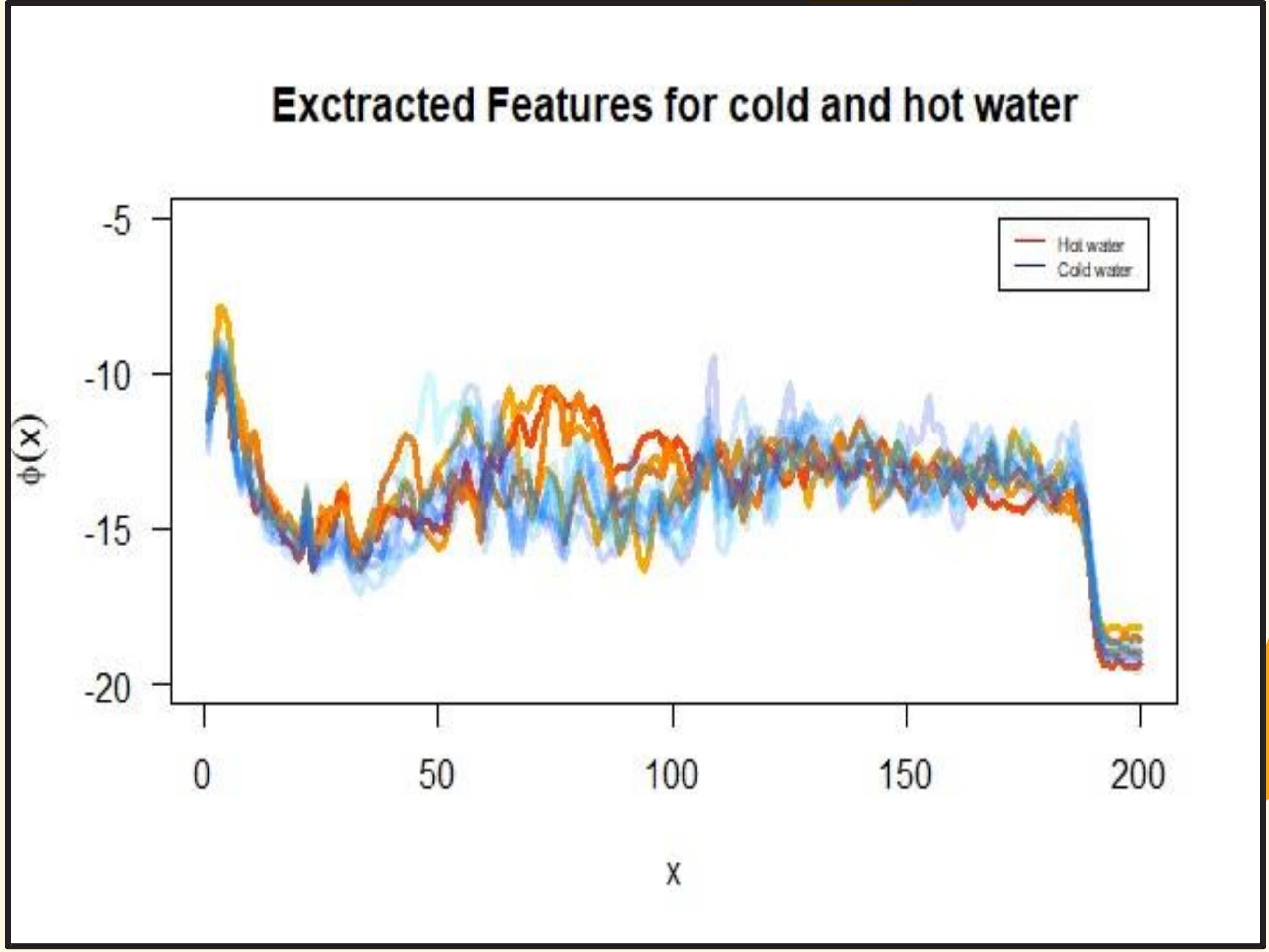
This meant to us that the interesting findings were not in the harmonic content per se but somewhere else.



Feature Engineering

We tried to model the perception of tone color with respect to temperature designing a customized feature extraction pipeline inspired by *Mel filters cepstrum coefficients* and the human hearing mechanism.

We did the following:
1. Filter the power spectrum with a Mel-Filter-bank to return an estimate of the perceived pitch;
2. Apply Weber-Fechner law of psychophysics that models the intensity of the perception as a *logarithm*.



Statistical Framework

1. Define the functional setting
2. Rebuild distances
3. Rebrand nonparametric models
4. Validate hyperparameters
5. Deploy the model!

In almost any of our attempt we ended up working with **functional covariates**. Since we thought that a linear model wouldn't be powerful enough to recover the underlying richness of the problem, we had to inject non-linearity. The easiest ways to do so were basically two:

- **Implement the *continuously additive model* $Y_i = \alpha + \int f(X_i(t), t)dt + \varepsilon_i$ where the functional form has to be estimated through a splines expansion;**
- **Go for *nonparametric approaches* correctly adapted for functional covariates.**

We need to define some useful ways to approximate **distance between functions**. Assuming $x(t), y(t) \in L^2([0,1])$ the distance is defined as consequence of the existence of a norm, so that we have $||x(t) - y(t)||_{L_2} = \sqrt{\int (x(t) - y(t))^2 dt}$. We could do two things:

- Approximate the integral through a finite sum over the data points;
- Approximate the distance through a basis expansion on an orthonormal basis and then leveraging Parseval's identity and bilinearity of inner product.

In the second case, given an orhonormal basis $\{\phi_j\}_{j=1}^{\infty}$ what happens is the following:

$$\left|\left|x(t)-y(t)\right|\right|^2_{L_2} = \sum_{j=1}^{\infty}\left|\left\langle x(t)-y(t),\phi_j\right\rangle\right|^2 = \sum_{j=1}^{\infty}\left|\left\langle x(t),\phi_j\right\rangle - \left\langle y(t),\phi_j\right\rangle\right|^2 = \left|\left|\beta_\infty^x - \beta_\infty^y\right|\right|$$

So from the last equality descends an approximation on a finite orthonormal basis which we expand our functions on to gather the empirical Generalized Fourier Coefficients.

This two approximations for the distance between functions are useful to be plugged into two well known estimators for the nonparametric model $Y = m(x) + \epsilon, m(x) = E[Y|X = x]$**:**

- **The Nadaraya-Watson estimator** $\widehat{m}(x) = \frac{\sum_{j=1}^{N} Y_j K(h^{-1} d(X_j, x))}{\sum_{j=1}^{N} K(h^{-1} d(X_j, x))}$ **;**
- **The k-nearest-neighbours regression** $\widehat{m}(x) = \frac{1}{|K_{nn}|} \sum_{k \in K_{nn}} Y_k$

Failures and Successes

Thank you!