Modal Symbolic Learning: Day 2

NATOPS: Interpretable gesture recognition

In [2]:
```
using Pkg
Pkg.activate(".")
Pkg.instantiate()
Pkg.update()
Pkg.status()
```

```
  Activating project at `~/Desktop/modal-symbolic-learning-course`
    Updating registry at `~/.julia/registries/General`
    Updating git-repo `https://github.com/JuliaRegistries/General.git`
  No Changes to `~/Desktop/modal-symbolic-learning-course/Project.toml`
  No Changes to `~/Desktop/modal-symbolic-learning-course/Manifest.toml`
Status `~/Desktop/modal-symbolic-learning-course/Project.toml`
  [a93c6f00] DataFrames v1.6.1
  [7806a523] DecisionTree v0.12.4
  [7073ff75] IJulia v1.24.2
  [033835bb] JLD2 v0.4.38
⌅ [add582a8] MLJ v0.19.5
  [c6f25543] MLJDecisionTreeInterface v0.4.0
  [e54bda2e] ModalDecisionTrees v0.3.3
  [91a5bcdd] Plots v1.39.0
  [7b3b3b3f] Sole v0.3.1
  [b002da8f] SoleLogics v0.6.12
  [4249d9c7] SoleModels v0.5.3
  [2913bbd2] StatsBase v0.34.2
  [9a3f8284] Random
Info Packages marked with ⌅ have new versions available but compatibility co
nstraints restrict them from upgrading. To see why use `status --outdated`
```

In [3]:
```
# Import libraries for statistics & Machine Learning
using Random
using DataFrames
using Plots
using StatsBase

using MLJ
using Sole
```

In [4]:
```
using JLD2
using DataFrames # (Maybe need version v1.6.1?)
JLD2.@load "COVID-dataset.jld2"
(X1_df, X2_df, y) = dataset;
```

In [5]:
```
countmap(y)
```

Out[5]:
```
Dict{String, Int64} with 2 entries:
  "POSITIVE" => 92
  "NEGATIVE" => 106
```

In [6]:
```
using SoleData

X_multimodal = MultiModalDataset([X1_df, X2_df])
```

Out[6]:
```
● MultiModalDataset{DataFrame}
   └─ dimensionalities: (1, 1)
- Modality 1 / 2
   └─ dimensionality: 1
198×30 SubDataFrame
 Row │ cough_F1                                 cough_F2
c …
     │ SubArray…                                SubArray…
S …
─────┼──────────────────────────────────────────────────────────────────────
─────
   1 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.0785252, 0.0786614, 0.0990329…
[ …
   2 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [1.28468, 3.24475, 3.20136, 1.59…
[
   3 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.888249, 0.787811, 5.26142, 6.…
[
   4 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [2.41169, 9.54773, 9.19982, 2.82…
[
   5 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [1.20103, 3.01929, 2.22691, 0.38…
[ …
   6 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.495915, 0.994237, 2.54288, 7.…
[
   7 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [1.90427, 6.09082, 6.85738, 3.17…
[
   8 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.958365, 3.90704, 27.6206, 37.…
[
   9 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.359918, 20.0994, 57.0254, 47.…
[ …
  10 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [2.02922, 4.27687, 6.02604, 6.44…
[
  11 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [2.23999, 4.85766, 3.8069, 2.351…
[
   ⋮ │                  ⋮                                  ⋮
⋱
 189 │ [0.0655681, 0.183465, 0.196258, …  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 190 │ [0.775358, 0.797555, 0.451902, 0…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[ …
 191 │ [0.00233577, 0.00146982, 0.00146…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 192 │ [0.054781, 0.220901, 0.21114, 0.…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 193 │ [0.0847868, 0.0649589, 0.0554716…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 194 │ [3.92422e-6, 5.71378e-6, 1.19569…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[ …
 195 │ [0.0911017, 0.051855, 0.0970747,…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 196 │ [0.00202282, 0.00423463, 0.00360…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 197 │ [0.000149194, 0.000536651, 0.001…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 198 │ [0.0844573, 0.085337, 0.141035, …  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[ …
                                                  28 columns and 177 rows om
```

```
itted
- Modality 2 / 2
   └─ dimensionality: 1
198×30 SubDataFrame
 Row │ breath_F1                              breath_F2
b …
     │ SubArray…                              SubArray…
S …
─────┼──────────────────────────────────────────────────────────────────────
─────
   1 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.0221009, 0.0145852, 0.0086312…
[ …
   2 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.000679687, 0.000366715, 0.000…
[
   3 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.00468303, 0.00279183, 0.00414…
[
   4 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.0589829, 0.035359, 0.00455235…
[
   5 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.000660904, 0.0132962, 0.04026…
[ …
   6 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.0318617, 0.0249504, 0.0400088…
[
   7 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.000429909, 0.000504201, 0.001…
[
   8 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.0380377, 0.0426292, 0.0145086…
[
   9 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.0227573, 0.029231, 0.0474346,…
[ …
  10 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.046796, 0.0428101, 0.014378, …
[
  11 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.139949, 0.10106, 0.0285217, 0…
[
   ⋮  │                  ⋮                                  ⋮
⋱
 189 │ [0.000197179, 8.33043e-5, 0.0001…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 190 │ [0.101633, 0.00446846, 0.0434787…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[ …
 191 │ [9.98121e-5, 8.11313e-5, 0.00015…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 192 │ [1.98063e-5, 1.50405e-5, 1.94345…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 193 │ [0.0100129, 0.00527175, 0.001193…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 194 │ [8.35598e-6, 9.95966e-6, 4.71731…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[ …
 195 │ [0.000305109, 4.17755e-5, 5.1256…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 196 │ [0.00015519, 0.000215899, 0.0001…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 197 │ [0.000506182, 0.00326664, 0.0037…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 198 │ [1.13929, 1.48325, 1.97274, 2.12…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[ …
                                                  28 columns and 177 rows om
itted
```

In [7]:
```
modality(X_multimodal, 1) # Cough modality
```

Out[7]: 198×30 SubDataFrame *173 rows omitted*

Row	cough_F1 SubArray…	cough_F2 SubArray…	cough_F3 SubArray…	cough_F4 SubArray…	cough_F5 SubArray…
1	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0785252, 0.0786614, 0.0990329, 0.0445775, 0.0399471, 0.0786838, 0.0457756, 0.00426709, 0.000980467, 0.00103658 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0155935, 0.0156205, 0.0196658, 0.00885215, 0.00793266, 0.0156249, 0.00909008, 0.000847355, 0.0001947, 0.000205843 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0109913, 0.0133814, 0.00546376, 0.00131151, 0.0012246, 0.00124394, 0.000484011, 0.000613768, 0.00108729, 0.000893859 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
2	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[1.28468, 3.24475, 3.20136, 1.59, 1.25703, 0.781295, 0.737144, 0.685029, 1.0493, 1.05133 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.25511, 0.644339, 0.635723, 0.315741, 0.24962, 0.155149, 0.146381, 0.136032, 0.208368, 0.208773 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.124351, 0.0984867, 0.0342663, 0.0857231, 0.0648201, 0.0334864, 0.0623284, 0.042826, 0.0545989, 0.0823318 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
3	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.888249, 0.787811, 5.26142, 6.34607, 4.24539, 3.91143, 1.44459, 0.293947, 0.266227, 0.258976 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.176388, 0.156443, 1.04481, 1.2602, 0.843046, 0.776728, 0.286865, 0.0583717, 0.0528672, 0.0514271 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0356732, 0.0336565, 0.465279, 0.751195, 0.537528, 0.317596, 0.11299, 0.0461445, 0.0443024, 0.0355204 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
4	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[2.41169, 9.54773, 9.19982, 2.82107, 1.15168, 0.257795, 1.88057, 2.0669, 0.468946, 0.209932 …	[0.478912, 1.89598, 1.82689, 0.560205, 0.2287, 0.0511927, 0.373443, 0.410442, 0.0931227, 0.0416881 …	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0864728, 1.16497, 1.61718, 0.676299, 0.277233, 0.334039, 0.405526, 0.209874, 0.00523093, 0.0100535 …

Row	cough_F1 SubArray…	cough_F2 SubArray…	cough_F3 SubArray…	cough_F4 SubArray…	cough_F5 SubArray…
		0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]		0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
5	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[1.20103, 3.01929, 2.22691, 0.381957, 0.424089, 0.427802, 1.19857, 2.49139, 1.86838, 0.827448 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.238499, 0.599568, 0.442217, 0.0758487, 0.0842152, 0.0849524, 0.238011, 0.494738, 0.371022, 0.164314 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0830592, 0.261478, 0.200136, 0.0243888, 0.0635885, 0.115569, 0.169187, 0.128275, 0.0685352, 0.0567436 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
6	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.495915, 0.994237, 2.54288, 7.6656, 9.81248, 5.78686, 3.64642, 1.99874, 0.176349, 0.155866 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0984784, 0.197435, 0.504963, 1.52223, 1.94855, 1.14915, 0.724103, 0.396907, 0.0350193, 0.0309518 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0133456, 0.0310985, 0.064594, 0.135595, 0.328474, 0.284642, 0.253371, 0.228533, 0.0345363, 0.0194849 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
7	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[1.90427, 6.09082, 6.85738, 3.17491, 3.14614, 3.73379, 1.79092, 0.896277, 0.407954, 0.272025 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.378147, 1.20951, 1.36173, 0.630471, 0.624757, 0.741452, 0.355638, 0.177982, 0.0810112, 0.0540184 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.130926, 0.472664, 0.346555, 0.00724088, 0.115631, 0.240525, 0.214053, 0.121031, 0.0507271, 0.0397091 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
8	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0,	[0.958365, 3.90704, 27.6206, 37.7196, 14.4782, 1.62699,	[0.190311, 0.775856, 5.48486, 7.49033, 2.87506, 0.323085,	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0,	[0.0997994, 0.183225, 1.27178, 1.87679, 0.736435, 0.0320454,

Row	cough_F1 SubArray…	cough_F2 SubArray…	cough_F3 SubArray…	cough_F4 SubArray…	cough_F5 SubArray…
	0.0, 0.0, 0.0, 0.0, 0.0]	0.444342, 0.247474, 0.309548, 0.3862 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	0.0882369, 0.0491431, 0.0614697, 0.0766912 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	0.0, 0.0, 0.0, 0.0, 0.0]	0.0184836, 0.0143935, 0.0470392, 0.0410738 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
9	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.359918, 20.0994, 57.0254, 47.1429, 13.1334, 4.95352, 2.27464, 2.11408, 2.04228, 1.51999 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0714722, 3.99132, 11.324, 9.36159, 2.60803, 0.983665, 0.451696, 0.419812, 0.405555, 0.301839 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0661794, 1.37233, 3.87948, 2.61952, 0.145436, 0.236887, 0.186483, 0.194598, 0.173557, 0.0815157 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
10	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[2.02922, 4.27687, 6.02604, 6.44009, 4.98728, 1.73571, 0.437956, 0.222406, 0.633753, 1.57527 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.402961, 0.849297, 1.19665, 1.27887, 0.99037, 0.344676, 0.0869689, 0.0441652, 0.12585, 0.312815 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.197969, 0.818105, 0.671579, 0.463143, 0.802507, 0.46406, 0.114277, 0.054087, 0.137064, 0.265332 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
11	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[2.23999, 4.85766, 3.8069, 2.35115, 2.81525, 2.06093, 0.504031, 0.0475342, 0.665366, 1.13394 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.444814, 0.964629, 0.755971, 0.466889, 0.55905, 0.409258, 0.10009, 0.00943929, 0.132128, 0.225177 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0664774, 0.710037, 1.09282, 0.492109, 0.37006, 0.334031, 0.0255459, 0.0348723, 0.0883853, 0.0835342 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
12	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0,	[1.54308, 4.40194,	[0.306422, 0.874132,	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0,	[0.155607, 0.270115,

Row	cough_F1 SubArray…	cough_F2 SubArray…	cough_F3 SubArray…	cough_F4 SubArray…	cough_F5 SubArray…
	0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	3.31183, 0.606782, 0.856285, 3.75008, 6.58074, 4.4986, 1.57232, 0.821402 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	0.65766, 0.120494, 0.17004, 0.744687, 1.3068, 0.893328, 0.312231, 0.163113 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	0.144422, 0.0271568, 0.123522, 0.190066, 0.394656, 0.53752, 0.307981, 0.086036 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
13	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.482869, 1.69527, 4.05378, 2.78207, 1.47997, 1.71612, 1.09331, 1.26579, 0.853612, 0.122273 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0958876, 0.336644, 0.804996, 0.55246, 0.293891, 0.340785, 0.217108, 0.251359, 0.169509, 0.0242809 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0936209, 0.204318, 0.469335, 0.335775, 0.107993, 0.125473, 0.141999, 0.11541, 0.0303423, 0.00877557 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
⋮	⋮	⋮	⋮	⋮	⋮
187	[0.0552184, 0.0346367, 0.00679207, 0.00234687, 0.00332229, 0.00235107, 0.00444243, 0.00504492, 0.000989146, 0.00074884 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0329837, 0.0369901, 0.0138507, 0.00131623, 0.00351097, 0.00413652, 0.00253019, 0.00218525, 0.000611488, 0.000411664 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0133451, 0.0149661, 0.00560395, 0.000532545, 0.00142053, 0.00167362, 0.00102371, 0.000884147, 0.000247407, 0.000166558 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
188	[0.0236068, 0.0532994, 0.0468775, 0.0272182, 0.021781, 0.00276494, 0.0244591, 0.0388653, 0.0166703, 0.00478709 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0797638, 0.0963726, 0.0592496, 0.0235339, 0.0157425, 0.0038016, 0.01393, 0.0138647, 0.00355454, 0.0031114 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,	[0.0322723, 0.0389921, 0.0239722, 0.00952177, 0.00636936, 0.00153812, 0.00563603, 0.00560962, 0.00143816, 0.00125887 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

Row	cough_F1 SubArray…	cough_F2 SubArray…	cough_F3 SubArray…	cough_F4 SubArray…	cough_F5 SubArray…
	0.0, 0.0, 0.0, 0.0]			0.0, 0.0, 0.0, 0.0]	0.0, 0.0, 0.0, 0.0]
189	[0.0655681, 0.183465, 0.196258, 0.0752565, 0.0245218, 0.00554621, 0.00420352, 0.0104721, 0.0107394, 0.008373 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0939916, 0.152787, 0.221585, 0.1576, 0.0210916, 0.00052029, 0.00659026, 0.00827054, 0.00279647, 0.00178117 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0380288, 0.0618172, 0.0896526, 0.0637646, 0.00853362, 0.000210508, 0.0026664, 0.00334624, 0.00113144, 0.000720659 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
190	[0.775358, 0.797555, 0.451902, 0.102062, 0.0429044, 0.129885, 0.133195, 0.0667011, 0.168251, 0.134291 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.448894, 0.229213, 0.195383, 0.0852792, 0.0812374, 0.148904, 0.0907529, 0.0259939, 0.0964742, 0.119557 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.181621, 0.0927389, 0.0790515, 0.0345038, 0.0328685, 0.0602462, 0.0367184, 0.0105171, 0.0390332, 0.0483726 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
191	[0.00233577, 0.00146982, 0.00146199, 0.00488409, 0.00498241, 0.00040649, 2.44607e-5, 8.5971e-5, 0.000140849, 0.0112882 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.000625756, 0.00110067, 0.00114407, 0.00222186, 0.00239491, 0.000501802, 3.88228e-5, 0.000172983, 0.000389273, 0.0106015 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.000253179, 0.000445328, 0.000462887, 0.000898959, 0.000968977, 0.000203028, 1.57076e-5, 6.99887e-5, 0.000157499, 0.00428933 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
192	[0.054781, 0.220901, 0.21114, 0.0542038, 0.0303408, 0.0235654, 0.0119132, 0.00392545,	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.239987, 0.257494, 0.161694, 0.0860483, 0.0555793, 0.027725, 0.00320995, 0.00260436,	[0.0970983, 0.104181, 0.0654209, 0.0348149, 0.0224873, 0.0112175, 0.00129874, 0.00105372,

Row	cough_F1 SubArray…	cough_F2 SubArray…	cough_F3 SubArray…	cough_F4 SubArray…	cough_F5 SubArray…
	0.00138098, 0.00151003 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]			0.00196054, 0.0012723 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	0.000793228, 0.000514768 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
193	[0.0847868, 0.0649589, 0.0554716, 0.0717362, 0.037787, 0.010012, 0.00316357, 0.00236503, 0.00124565, 0.00351257 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.00204523, 0.0404258, 0.0475562, 0.0146389, 0.0120549, 0.00618275, 0.00340599, 0.00238219, 0.000709623, 0.000828831 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.000827497, 0.0163562, 0.0192411, 0.00592285, 0.00487738, 0.00250153, 0.00137806, 0.00096383, 0.000287112, 0.000335343 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
194	[3.92422e-6, 5.71378e-6, 1.19569e-5, 1.20861e-5, 1.5383e-5, 3.17573e-5, 9.81268e-5, 0.000134689, 0.000127148, 8.57796e-5 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[3.11101e-7, 5.9307e-6, 9.71927e-6, 4.55087e-6, 4.40629e-6, 2.24785e-5, 6.15706e-5, 7.54206e-5, 0.000126442, 0.000106185 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[1.25871e-7, 2.39955e-6, 3.93239e-6, 1.84127e-6, 1.78277e-6, 9.09475e-6, 2.49113e-5, 3.0515e-5, 5.1158e-5, 4.2962e-5 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
195	[0.0911017, 0.051855, 0.0970747, 0.138937, 0.26077, 0.26588, 0.0621817, 0.0126505, 0.0948102, 0.121012 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0835796, 0.0972449, 0.124756, 0.184055, 0.216762, 0.231904, 0.120916, 0.0047731, 0.0528686, 0.1167 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0338161, 0.0393451, 0.0504759, 0.0744684, 0.0877015, 0.0938278, 0.0489225, 0.00193119, 0.0213905, 0.0472167 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
196	[0.00202282, 0.00423463, 0.00360257, 0.000574975,	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0,	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0,	[0.0011699, 0.00141079, 0.000994817, 0.000483729,	[0.000473337, 0.000570802, 0.0004025, 0.000195715,

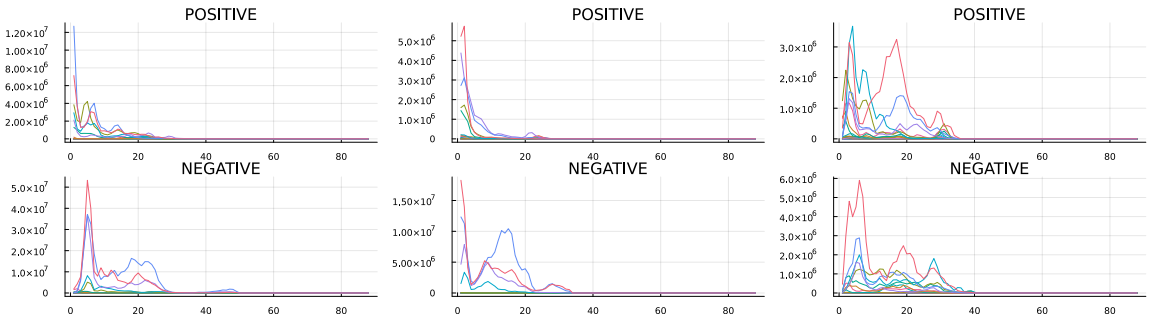
Row	cough_F1 SubArray…	cough_F2 SubArray…	cough_F3 SubArray…	cough_F4 SubArray…	cough_F5 SubArray…
	0.000259837, 0.000132237, 0.000103696, 0.00079894, 0.000845581, 0.000316039 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	0.000261735, 0.00017485, 5.15177e-5, 0.00032861, 0.000290001, 0.000117191 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	0.000105897, 7.0744e-5, 2.08439e-5, 0.000132955, 0.000117334, 4.74152e-5 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
197	[0.000149194, 0.000536651, 0.00171301, 0.00269341, 0.00145864, 0.000112078, 0.00677374, 0.00971658, 0.0669641, 0.0664799 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.000391549, 0.000916783, 0.000917613, 0.000662408, 0.000452083, 9.83684e-5, 0.00814364, 0.0143803, 0.0410952, 0.0555311 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.00015842, 0.000370928, 0.000371264, 0.000268009, 0.000182912, 3.97996e-5, 0.0032949, 0.00581822, 0.016627, 0.0224678 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
198	[0.0844573, 0.085337, 0.141035, 0.257834, 0.213644, 0.0818095, 0.0223061, 0.0394858, 0.0461354, 0.0437527 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0465652, 0.20439, 0.486119, 0.420521, 0.234654, 0.128405, 0.0582556, 0.0390655, 0.0763615, 0.104272 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]	[0.0188401, 0.0826958, 0.196683, 0.170142, 0.0949404, 0.0519525, 0.0235701, 0.0158058, 0.0308957, 0.0421881 … 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]

In [8]:
```
names(X1_df)
```

Out[8]:
```
30-element Vector{String}:
 "cough_F1"
 "cough_F2"
 "cough_F3"
 "cough_F4"
 "cough_F5"
 "cough_F6"
 "cough_F7"
 "cough_F8"
 "cough_F9"
 "cough_F10"
 "cough_F11"
 "cough_F12"
 "cough_F13"
 ⋮
 "cough_F19"
 "cough_F20"
 "cough_F21"
 "cough_F22"
 "cough_F23"
 "cough_F24"
 "cough_F25"
 "cough_F26"
 "cough_F27"
 "cough_F28"
 "cough_F29"
 "cough_F30"
```

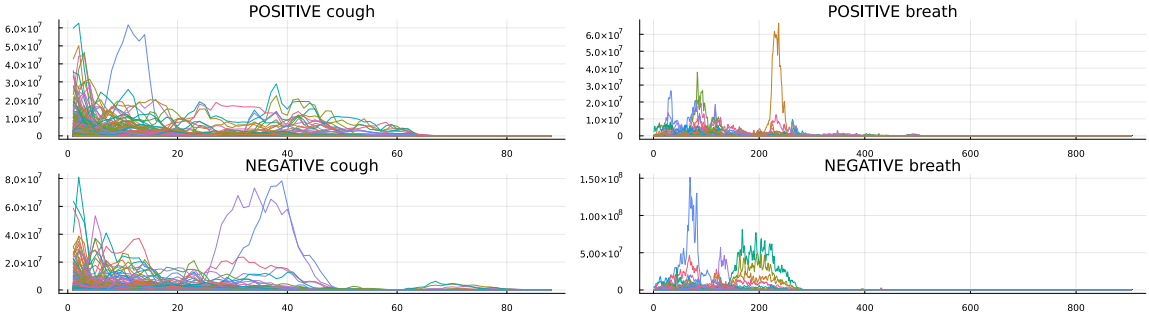
In [9]:
```
# Let's inspect some instances
plot(map(i->plot(collect(X1_df[i,:]), labels=nothing,title=y[i]), 1:35:(35*6
```

Out[9]:



In [10]:
```
# All instances, grouped by class
plot(map((((X_mod, mod_name), _y),)->plot(collect.(eachrow(X_mod[findall(__y
```

Out[10]:



In [11]:

```
X2_cough = broadcast(x->movingwindow(mean, x[1:60]; nwindows = 30, relative_
X2_breath = broadcast(x->movingwindow(mean, x[1:330]; nwindows = 30, relativ

# X2_cough = map(x->x[1:60], X1_df)
# X2_breath = map(x->x[1:330], X2_df)
X_multimodal = MultiModalDataset([X2_cough, X2_breath])
# broadcast(length, X2_cough)
# broadcast(length, X2_breath)
```

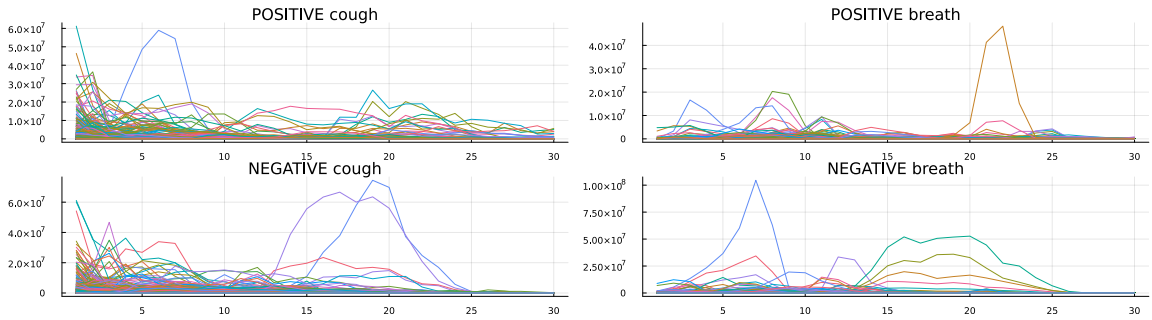
Out[11]:

```
● MultiModalDataset{DataFrame}
   └─ dimensionalities: (1, 1)
- Modality 1 / 2
   └─ dimensionality: 1
198×30 SubDataFrame
 Row │ cough_F1                                 cough_F2
c …
     │ Array…                                   Array…
A …
─────┼──────────────────────────────────────────────────────────────────────────
─────
   1 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.0785933, 0.0718052, 0.0593155…
[ …
   2 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [2.26471, 2.39568, 1.01916, 0.71…
[
   3 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.83803, 5.80374, 4.07841, 0.86…
[
   4 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [5.97971, 6.01044, 0.704739, 1.9…
[
   5 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [2.11016, 1.30443, 0.425945, 1.8…
[ …
   6 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.745076, 5.10424, 7.79967, 2.8…
[
   7 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [3.99754, 5.01615, 3.43996, 1.34…
[
   8 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [2.4327, 32.6701, 8.05258, 0.345…
[
   9 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [10.2297, 52.0841, 9.04348, 2.19…
[ …
  10 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [3.15305, 6.23306, 3.3615, 0.330…
[
  11 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [3.54882, 3.07903, 2.43809, 0.27…
[
   ⋮ │                 ⋮                                  ⋮
⋱
 189 │ [0.124517, 0.135757, 0.015034, 0…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 190 │ [0.786457, 0.276982, 0.0863947, …  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[ …
 191 │ [0.00190279, 0.00317304, 0.00269…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 192 │ [0.137841, 0.132672, 0.0269531, …  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 193 │ [0.0748728, 0.0636039, 0.0238995…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 194 │ [4.819e-6, 1.20215e-5, 2.35702e-…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[ …
 195 │ [0.0714784, 0.118006, 0.263325, …  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 196 │ [0.00312872, 0.00208877, 0.00019…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 197 │ [0.000342922, 0.00220321, 0.0007…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 198 │ [0.0848972, 0.199435, 0.147727, …  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[ …
                                                  28 columns and 177 rows om
```

```
itted
- Modality 2 / 2
   └─ dimensionality: 1
198×30 SubDataFrame
 Row │ breath_F1                              breath_F2
b …
     │ Array…                                 Array…
A …
─────┼──────────────────────────────────────────────────────────────────────
─────
   1 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.0142488, 0.0122529, 0.0121231…
[ …
   2 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.014266, 0.0145877, 0.00226485…
[
   3 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.0880448, 0.0142345, 0.0051122…
[
   4 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.179997, 0.0865722, 0.025112, …
[
   5 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.0741071, 0.0120109, 0.0083648…
[ …
   6 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.0447948, 0.023875, 0.0755589,…
[
   7 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.016167, 0.00786436, 0.0026574…
[
   8 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.0226184, 0.00625794, 0.007319…
[
   9 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.0584561, 0.0425855, 0.0726359…
[ …
  10 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.0255779, 0.0121605, 0.0079572…
[
  11 │ [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…  [0.058674, 0.045875, 0.0324705, …
[
   ⋮ │                  ⋮                                  ⋮
⋱
 189 │ [0.000613481, 0.000962912, 0.001…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 190 │ [0.028399, 0.0160754, 0.0207845,…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[ …
 191 │ [0.000173462, 0.0001914, 0.00029…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 192 │ [0.000238292, 0.000707364, 0.000…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 193 │ [0.00256501, 0.00167421, 0.00156…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 194 │ [0.00135499, 0.00103865, 0.00070…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[ …
 195 │ [6.97807e-5, 9.75927e-5, 0.00010…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 196 │ [0.000873765, 0.00209408, 0.0039…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 197 │ [0.00112673, 0.00166572, 0.00088…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[
 198 │ [1.19151, 0.140725, 0.0652271, 0…  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0…
[ …
                                                  28 columns and 177 rows om
itted
```

In [12]:
```
# All instances, grouped by class
plot(map((((X_mod, mod_name), _y),)->plot(collect.(eachrow(X_mod[findall(__y
```

Out[12]:



In [13]:
```
using ModalDecisionTrees
```

In [ ]:
```
# Train in cross-validation!
e = evaluate!(machine(ModalDecisionTree(; relations = :RCC8), X_multimodal,
        resampling=StratifiedCV(shuffle=true, nfolds = 3),
        measures=[accuracy],
        verbosity=0,
        check_measure=false
)
```

```
[ Info: Precomputing logiset...
```

In [47]:
```
# Accuracies per fold
e.per_fold
```

```
UndefVarError: `e` not defined

Stacktrace:
 [1] top-level scope
   @ In[47]:2
```

In [48]:
```
e.report_per_fold[2].solemodel
```

```
UndefVarError: `e` not defined

Stacktrace:
 [1] top-level scope
   @ In[48]:1
```

In [46]:
```
report(mach).solemodel
```

Out[46]:
```
▣ {1}(⟨G⟩(min[V37] ≥ 0.0013205573259328035))
├✔ {1}(⟨G⟩((min[V37] ≥ 0.0013205573259328035) ∧ ⟨L̄⟩(min[V27] ≥ 226321.9912113276)))
│├✔ NEGATIVE
│└✘ {1}(⟨G⟩((min[V37] ≥ 0.0013205573259328035) ∧ (min[V48] ≥ 1242.1904769961468)))
│ ├✔ {1}(⟨G⟩((min[V37] ≥ 0.0013205573259328035) ∧ (min[V48] ≥ 1242.1904769961468) ∧ ⟨AO⟩(min[V40] ≥ 3.51978641624239)))
│ │├✔ {1}(⟨G⟩((min[V37] ≥ 0.0013205573259328035) ∧ (min[V48] ≥ 1242.1904769961468) ∧ ⟨AO⟩((min[V40] ≥ 3.51978641624239) ∧ ⟨AO⟩(min[V31] ≥ 1.620774580004222 6e-5))))
│ ││├✔ POSITIVE
│ ││└✘ NEGATIVE
│ │└✘ POSITIVE
│ └✘ NEGATIVE
└✘ {1}(⟨G⟩(min[V27] ≥ 2.170219509362793e6))
 ├✔ {1}(⟨G⟩((min[V27] ≥ 2.170219509362793e6) ∧ ⟨G⟩(min[V46] ≥ 5838.390775457752)))
 │├✔ POSITIVE
 │└✘ {1}(⟨G⟩((min[V27] ≥ 2.170219509362793e6) ∧ (min[V30] < 1.6492884918659376e6)))
 │ ├✔ POSITIVE
 │ └✘ NEGATIVE
 └✘ {1}(⟨G⟩(min[V28] ≥ 672777.1743629333))
  ├✔ POSITIVE
  └✘ {1}(⟨G⟩(min[V30] ≥ 1.0114283593133552e6))
   ├✔ NEGATIVE
   └✘ POSITIVE
```

In [ ]:
```
X = scalarlogiset(X_df)
println(X)
```

In [ ]:
```

```

In [ ]:
```
# Accessibles on multirelation frames
# Get the structure ("frame") of the first instance. It is a "dimensional" f
fr = SoleLogics.frame(X, 1)
```

In [ ]:
```
# Enumerate all worlds
collect(allworlds(fr))
```

In [ ]:
```
using SoleLogics: Interval
# Enumerate the intervals that are "Later" than [1,10]
accessibles(fr, Interval(1,10), IA_L) |> collect
```

In [ ]:
```
# Remember that features are computed on each world
# Let's compute the minimum of the first variable on an arbitrary interval,
feature = UnivariateMin(1)
Sole.featvalue(feature, X, 1, Interval(10,30))
```

In [ ]:
```
# Remember that atoms are *scalar conditions on features*
# Let's check one on an interval of the first instance
```

```
p = Atom(ScalarCondition(feature, >, -0.5))
check(p, X, 1, Interval(10,30))
```

In [ ]:
```
# I can check any formula
p = Atom(ScalarCondition(UnivariateMin(1), >, -0.5))
q = Atom(ScalarCondition(UnivariateMin(2), <=, 10))
φ = p ∨ q
check(φ, X, 1, Interval(10,30))
```

In [ ]:
```
# Generate a random HS formula with scalar conditions on features, and check
features = [UnivariateMin(i_variable) for i_variable in 1:ncol(X_df)]
alpha = [Atom(ScalarCondition(feat, >, thresh)) for feat in features for thr

HS_connectives = SoleLogics.diamondsandboxes(SoleLogics.IARelations)
propo_connectives = SoleLogics.BASE_PROPOSITIONAL_CONNECTIVES

println("Propositional connectives: $(join(syntaxstring.(propo_connectives),
println("HS connectives: $(join(syntaxstring.(HS_connectives), ", "))")

propo_weights = fill(1/length(propo_connectives), length(propo_connectives))
HS_weights = fill(1/length(HS_connectives), length(HS_connectives))

connectives = vcat(propo_connectives, IA_connectives)

opweights = vcat(propo_weights, HS_weights)

treeheight = 3
φ2 = randformula(Random.MersenneTwister(30), treeheight, alpha, connectives;
println()
println("Random formula:")
println(syntaxstring(φ2))

check(φ2, X, 1, Interval(10,30))
```

In [ ]:
```
# Let's check a formula on all the instances
check_mask = check(φ2, X, Interval(10,30))
```

In [ ]:
```
# It holds on part of the instances
sum(check_mask)
```

In [ ]:
```
# Let's ask whether the formula holds *all* intervals, instead of checking i
println("Applying the universal global operator: ", SoleLogics.globalbox)
println()

universal_φ = globalbox(φ2)
println("Formula: ", syntaxstring(universal_φ))
check_mask = check(universal_φ, X)

# It holds on no instance... Too restrictive!
sum(check_mask)
```

In [ ]:
```
# Let's ask whether there exists any interval where the formula holds
println("Applying the existential global operator: ", SoleLogics.globaldiamo
println()
```

```
existential_φ = globaldiamond(φ2)
println("Formula: ", syntaxstring(existential_φ))
check_mask = check(existential_φ, X)

# It holds on more instances
sum(check_mask)
```

In [ ]:
```
# Question: does it lead to a good rule?

println(syntaxstring(existential_φ))

println()
println(SoleLogics.experimentals.formula2natlang(existential_φ))
```

In [ ]:
```
neg_existential_φ = normalize(¬ existential_φ;
    profile = :readability,
    remove_implications = true,
    allow_atom_flipping = true
)

syntaxstring(neg_existential_φ; remove_redundant_parentheses = false)
```

In [ ]:
```
(SoleLogics.precedence(∧), SoleLogics.precedence(∨))
```

In [ ]:
```
countmap(y)
```

In [ ]:
```
println(existential_φ)
countmap(y[check_mask])
```

In [ ]:
```
println(neg_existential_φ)
countmap(y[(!).(check_mask)])
```

In [ ]:
```
branch = Branch(neg_existential_φ, "I have command", "Spread wings")
```

In [ ]:
```
y_preds = SoleModels.apply(branch, X)
println("Accuracy of this branch: $(sum(y .== y_preds)/length(y))")
println("Random chanc e: $(60/length(y))")
```

In [ ]:
```

```

In [ ]:
```
println(X)
```

In [ ]:
```
# Randomly split the data: 20% training, 80% testing
N = nrow(X_df)
perm = randperm(Random.MersenneTwister(1), N)
train_idxs, test_idxs = perm[1:round(Int, N*.2)], perm[round(Int, N*.2)+1:en
println("Using $(length(train_idxs)) instances for training")
println("Using $(length(test_idxs)) instances for testing")
```

In [ ]:
```
using ModalDecisionTrees
```

```
# Bind a machine learning algorithm to logiset & labels
mach = machine(ModalDecisionTree(; relations = :IA7, features = [minimum]),

# Train!
@time fit!(mach; rows=train_idxs);

# Compute accuracy
yhat = predict_mode(mach; rows=test_idxs)
MLJ.accuracy(yhat, y[test_idxs])
```

In [ ]:
```
# Show the restricted MDT learnt
printmodel(report(mach).rawmodel_full; hidemodality = true)
```

In [ ]:
```
# Show its *pure* version
printmodel(report(mach).solemodel_full; show_metrics = true, hidemodality =
```

In [ ]:
```
simplified_restricted_tree = ModalDecisionTrees.prune(report(mach).rawmodel_
printmodel(simplified_restricted_tree)

println()
println("# Leaves: ", nleaves(simplified_restricted_tree))
```

In [ ]:
```
solemodel = ModalDecisionTrees.translate(simplified_restricted_tree)
```

In [ ]:
```
# Print leaf rules + their training performances
ruleset = listrules(solemodel)
printmodel.(ruleset; show_metrics = true, threshold_digits = 2, variable_nam
```

In [ ]:
```
last_rule = ruleset[end]
last_antd = antecedent(last_rule)

println("First formula, translated:")
println(SoleLogics.experimentals.formula2natlang(last_antd; threshold_digits


for (i_rule, rule) in enumerate(ruleset)
    println()
    println("[$i_rule]")
    antd = antecedent(rule)
    println(SoleLogics.experimentals.formula2natlang(antd; threshold_digits
end
```

In [ ]:
```
# Print rules + their *test* performances

# Sprinkle the model with the test instances!
predictions, tree_test = report(mach).sprinkle(X_df[test_idxs,:], y[test_idx

# Extract ruleset and print its metrics
ruleset_test = listrules(tree_test);

# printmodel.(ruleset_test; show_metrics = true, threshold_digits = 2, varia
printmodel.(ruleset_test; show_metrics = true, threshold_digits = 2, parenth
```

In [ ]:
```
println("IF\n\t", SoleLogics.experimentals.formula2natlang(antecedent(rulese
println("THEN\n\t", consequent(ruleset_test[4]))
```

In [ ]:
```
# Obtain class rules & show their *test* metrics
condensed_ruleset_test = joinrules(ruleset_test)
printmodel.(condensed_ruleset_test; show_metrics = true, threshold_digits =
```

Exercise

`ModalDecisionTrees.jl` can also handle images! In which case, they use a 2D logic of rectangles instead of a 1D logic of intervals.

Apply ModalDecisionTrees to Indian Pines, a benchmark dataset for Land Cover Classification with 16 classes. The dataset consists of a hyperspectral image (i.e., 200 color channels instead of the typical 3 RGB channels) where many pixels have been labelled as belonging to one of the classes.

Sketch of the idea:

- Load the image and the ground truths. The package MAT.jl can be helpful;
- From the 145×145 image provided, sample a (small) number m of 3×3 patches for each class, and label each patch with the class label for the central pixel;
- Gather the ground truths into a vector of strings `y`, and the samples into a Julia `DataFrame` `X` with 200 columns, $16m$ rows and 3×3 *matrices* in the cells;
- Use MLJ to train a ModalDecisionTree on `X` and `y`, similarly to the above case.

Suggestion: since the formulas are desirably rotation-invariant, ask the algorithm to use *topological* relations instead of *directional* relations. Rrefer to the doc to know more.