threshhold_cluster_mean_dillution

November 12, 2023

[ ]:

```
import numpy as np
import pandas as pd
import data_lib
import plot_lib
import transform_lib
import decision_lib
from sklearn import cluster, mixture

np.random.seed(200)
```

[ ]:

```
# print available data summary
_ = data_lib.explore_datasets(datafolder="../../Data",verbose=True)
print(data_lib.LABELS_LIST)
```

```
--------------------------------------------------------------------------------
--------------
-- The following 4 groups were found
-- They contain 40 datasets
-- The first printed entity is the key to the returned dictionary
-----------------------------------
Group: ../../Data/6P-positive-dilution-series-2-labelled/droplet-level-
data/RawData
po-di-se-2-A4, files: 13                po-di-se-2-C4, files: 13
po-di-se-2-A1, files: 13
po-di-se-2-B1, files: 13                po-di-se-2-D1, files: 13
po-di-se-2-B4, files: 13
po-di-se-2-C1, files: 13                po-di-se-2-D4, files: 13
-----------------------------------
Group: ../../Data/6P-positive-dilution-series-1-labelled/droplet-level-
data/RawData
po-di-se-1-D4, files: 13                po-di-se-1-A4, files: 13
po-di-se-1-A1, files: 13
po-di-se-1-D1, files: 13                po-di-se-1-B1, files: 13
po-di-se-1-C1, files: 13
po-di-se-1-B4, files: 13                po-di-se-1-C4, files: 13
-----------------------------------
Group: ../../Data/6P-positive-dilution-series-labelled/droplet-level-
data/RawData
```

```
po-di-se-B8, files: 13                  po-di-se-A8, files: 13
po-di-se-C8, files: 13
po-di-se-D8, files: 13
-----------------------------------
Group: ../../Data/6P-wastewater-samples-labelled/droplet-level-data/RawData
wa-sa-A2, files: 13                     wa-sa-B4, files: 13
wa-sa-C5, files: 13
wa-sa-C4, files: 13                     wa-sa-B3, files: 13
wa-sa-B2, files: 13
wa-sa-A5, files: 13                     wa-sa-A3, files: 13
wa-sa-C2, files: 13
wa-sa-C3, files: 13                     wa-sa-D3, files: 13
wa-sa-D4, files: 13
wa-sa-B1, files: 13                     wa-sa-A4, files: 13
wa-sa-A1, files: 13
wa-sa-D2, files: 13                     wa-sa-D5, files: 13
wa-sa-C1, files: 13
wa-sa-B5, files: 13                     wa-sa-D1, files: 13
-----------------------------------
['IAV-M_POS', 'IAV-M_NEG', 'IBV-M_POS', 'IBV-M_NEG', 'MHV_POS', 'MHV_NEG', 'RSV-
N_POS', 'RSV-N_NEG', 'SARS-N1_POS', 'SARS-N1_NEG', 'SARS-N2_POS', 'SARS-N2_NEG']
```

0.0.1 Get samples for negative control

[ ]:

```
# load the necessary datasetes
df_di = data_lib.load_dataset(None, [
    "po-di-se-2-A4", "po-di-se-2-B1", "po-di-se-2-C1", "po-di-se-2-C4",
    "po-di-se-2-D1", "po-di-se-2-D4", "po-di-se-2-A1", "po-di-se-2-B4",

    #"po-di-se-1-D4", "po-di-se-1-D1", "po-di-se-1-B4", "po-di-se-1-A4",
    #"po-di-se-1-B1", "po-di-se-1-C4", "po-di-se-1-A1", "po-di-se-1-C1",

    #"po-di-se-B8", "po-di-se-D8", "po-di-se-A8", "po-di-se-C8",
                                     ],
                              datafolder="../../Data")



df_negative_control = data_lib.load_dataset([],[
    "po-di-se-1-D1", "po-di-se-1-D4",
    "po-di-se-2-D1", "po-di-se-2-D4",
    "po-di-se-D8",
                                                ],
                                            datafolder="../../Data")
# Everything is positive contol
df_positive_control = df_di.iloc[:,:6]

# convert to numpy
```

```
np_di = df_di.to_numpy(copy=True)[:,:6]
np_negative_control = df_negative_control.to_numpy()
np_positive_control = df_positive_control.to_numpy()

# generate decorrelation transform
ZCA_whitener = transform_lib.WhitenTransformer(transform_lib.Whitenings.ZCA_COR)

# get the axis-disease correspondence
prediction_axis =␣
 ↪['SARS-N2_POS','SARS-N1_POS','IBV-M_POS','RSV-N_POS','IAV-M_POS','MHV_POS']

# fix clustering algorithm
cluster_engine = cluster.KMeans(n_clusters=256)
#cluster_engine = mixture.BayesianGaussianMixture(n_components=64)
#cluster_engine = cluster.DBSCAN(eps=0.1, n_jobs=8)
```

[ ]:

```
# Define classifier
zca_decitions = decision_lib.ThresholdMeanClassifier(
                                     negative_control=np_negative_control,
                                     positive_control=np_positive_control,
                                     cluster_algorithm=cluster_engine,
                                     transform_base="pos",
                                     whitening_transformer=ZCA_whitener,
                                     prediction_axis=prediction_axis,
                                     )
# train classifier and predict labels
df_zca_preds = zca_decitions.fit_predict(np_di)
```

```
/home/nico/.cache/pypoetry/virtualenvs/ds-lab-4Qf2VVQw-
py3.11/lib/python3.11/site-packages/sklearn/cluster/_kmeans.py:1416:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/home/nico/.cache/pypoetry/virtualenvs/ds-lab-4Qf2VVQw-
py3.11/lib/python3.11/site-packages/sklearn/cluster/_kmeans.py:1416:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```

0.1 Plot the predictions

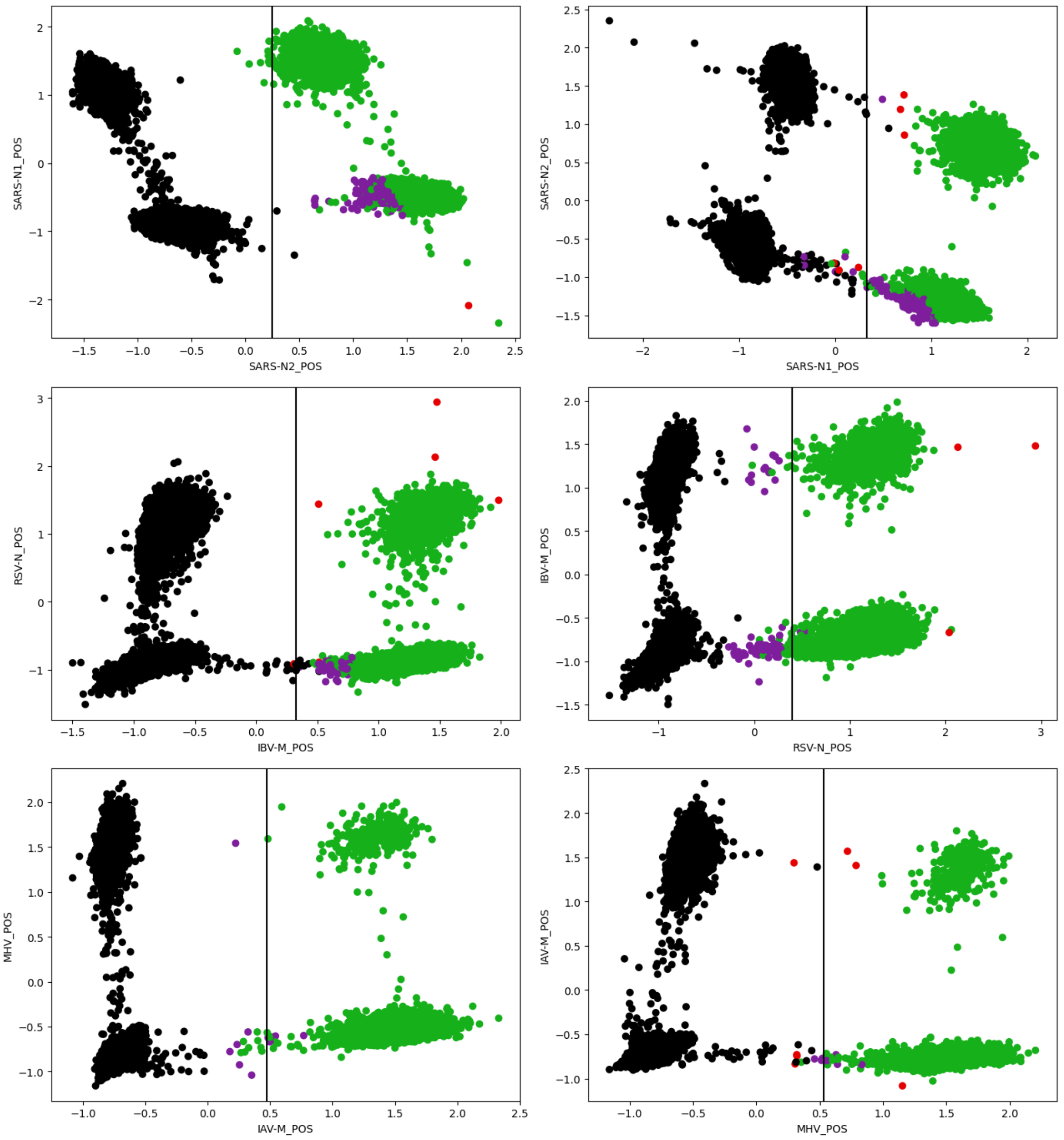
- Black = True negative prediction
- Green = True positive prediciton
- Purple = False negative
- Red = False positive

Plot for all diseases predictions against ground truth Firs plot is in the decorrelated coordinates, whereas the second is in original coordinates

[ ]:

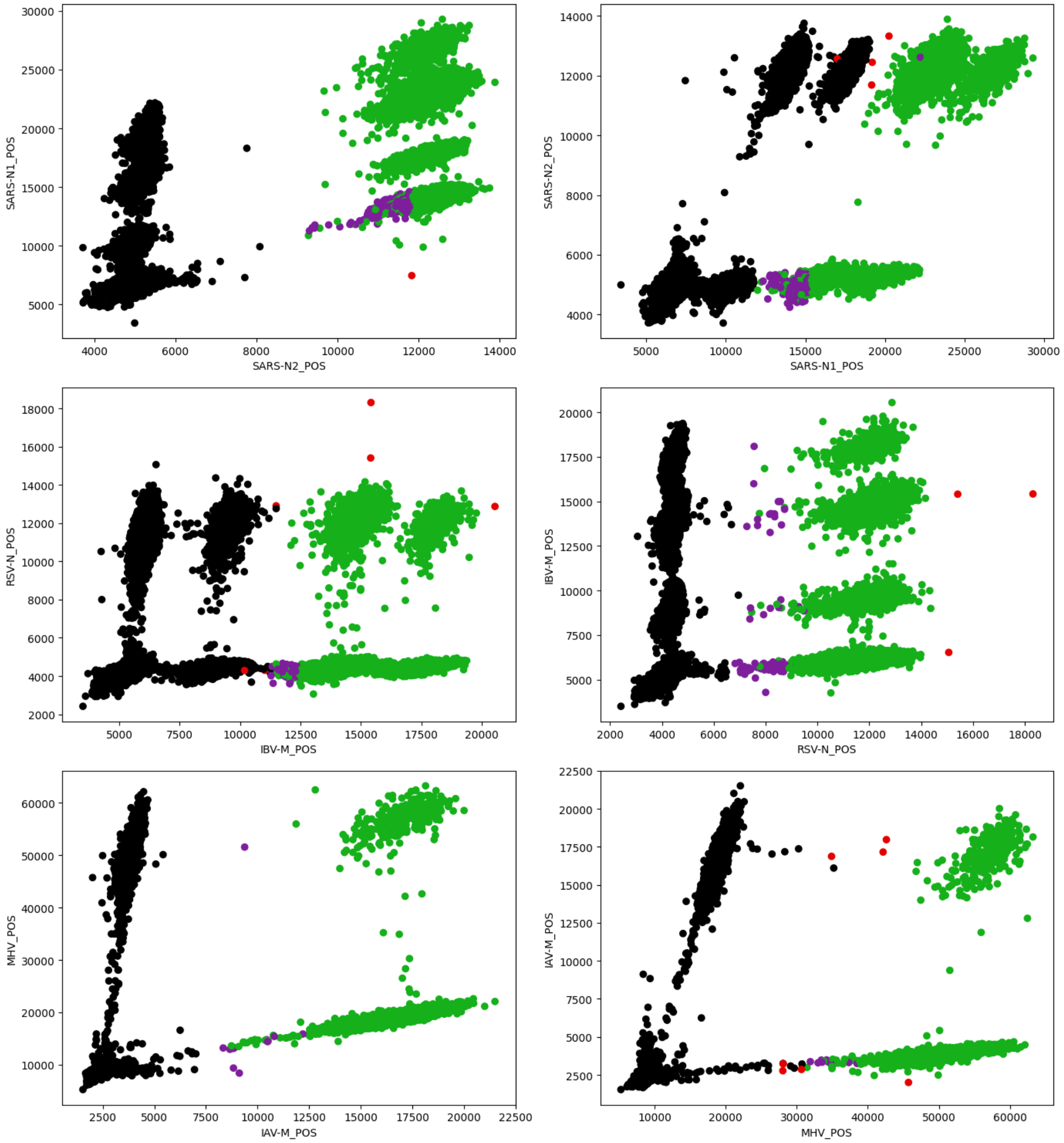
```
df_data_points = pd.DataFrame(data=zca_decitions.X_all_transformed,␣
 ↪columns=prediction_axis)
df_predictions = df_zca_preds
df_ground_trouth = df_di
selected_pairs = [
                ('SARS-N2_POS','SARS-N1_POS'),
                ('SARS-N1_POS','SARS-N2_POS'),
                ('IBV-M_POS','RSV-N_POS'),
                ('RSV-N_POS','IBV-M_POS'),
                ('IAV-M_POS','MHV_POS'),
                ('MHV_POS','IAV-M_POS'),
                ]
axis_thres = pd.DataFrame(data=zca_decitions.axis_threshholds.reshape(1,-1),␣
 ↪columns=prediction_axis)

plot_lib.plot_pairwise_selection(
        df_data_points,
        df_predictions,
        df_ground_trouth,
        selected_pairs,
        axis_thresh=axis_thres,
        n_cols=2,
        )
```



```
[ ]: df_data_points = pd.DataFrame(data=zca_decitions.X, columns=prediction_axis)
     df_predictions = df_zca_preds
     df_ground_trouth = df_di
     selected_pairs = [
                     ('SARS-N2_POS','SARS-N1_POS'),
                     ('SARS-N1_POS','SARS-N2_POS'),
                     ('IBV-M_POS','RSV-N_POS'),
                     ('RSV-N_POS','IBV-M_POS'),
                     ('IAV-M_POS','MHV_POS'),
                     ('MHV_POS','IAV-M_POS'),
                     ]
     # axis_thres = pd.DataFrame(data=zca_decitions.axis_threshholds.reshape(1,-1),␣
      ↪columns=prediction_axis)

     plot_lib.plot_pairwise_selection(
             df_data_points,
             df_predictions,
             df_ground_trouth,
             selected_pairs,
             axis_thresh=None,
             n_cols=2,
             )
```


SARS-N1_POS
SARS-N2_POS
SARS-N2_POS
SARS-N1_POS
RSV-N_POS
IBV-M_POS
IBV-M_POS
RSV-N_POS
MHV_POS
IAV-M_POS
IAV-M_POS
MHV_POS