Submission - Exercise [7]

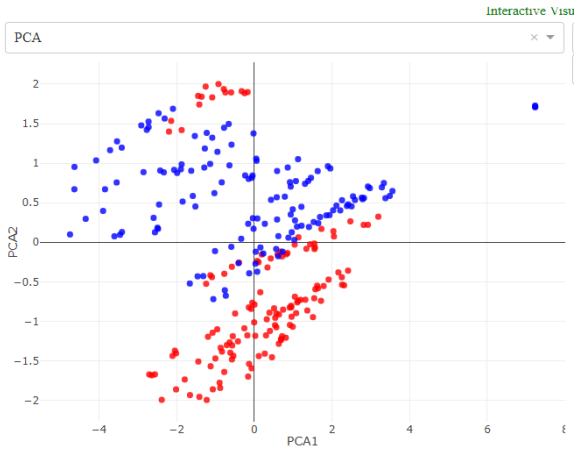
Visual Data Analysis

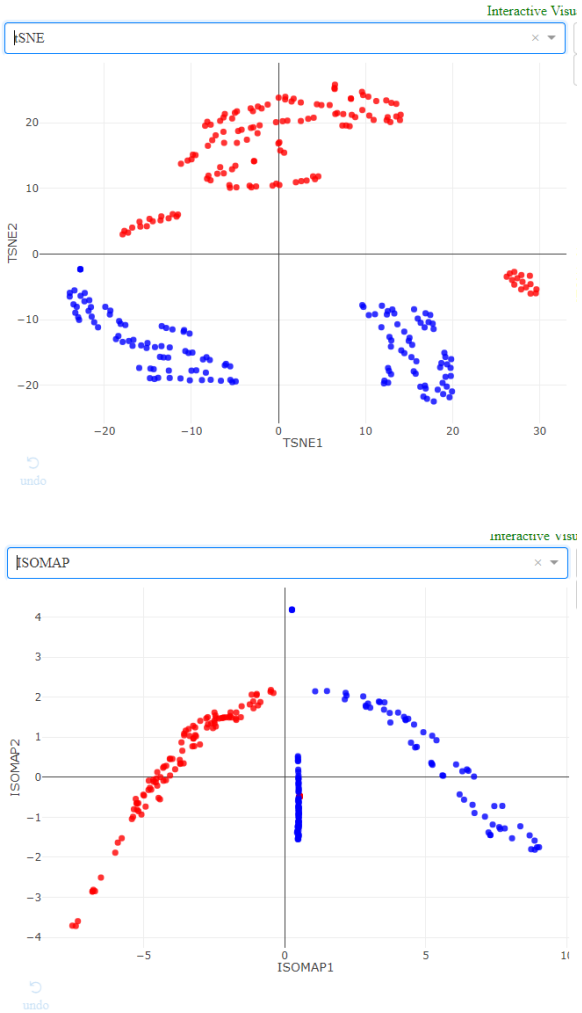
[Maryam Assaedi, maryam.assaedi@rwth-aachen.de]
[Mst. Mahfuja Akter , s6msakte@uni-bonn.de]
[Mahpara Hyder Chowdhury, s6machow@uni-bonn.de]

May 19, 2019

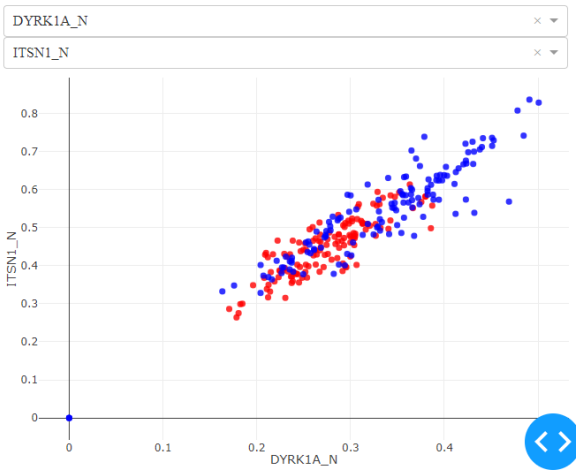
Exercise 1 (Interactive Visualization with Dash)

a) Read the file Data_Cortex_Nuclear.xls, and restrict it to the classes t-CS-s and c-CS-s. As in Sheet 5, run PCA, ISOMAP and t-SNE as dimensionality reduction techniques. Within the Dash framework, create a single scatter plot that will show the output from one of the techniques, as selected by the user. Add a dropdown component from Dash to switch between the different techniques. A callback function is a function which is triggered by an event such as a mouse click, double click, or selection. In Dash, you can use @app.callback to define your event callback, with the dropdown value as the input. Here is our output of implementation in one dropdown data for PCA, ISOMAP and t-SNE.



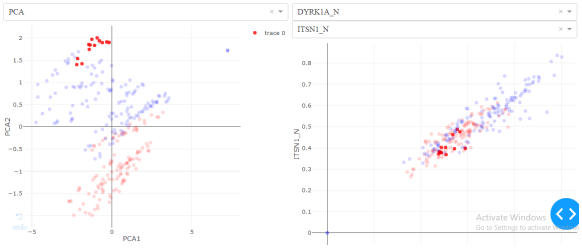


b) Add a second scatter plot with corresponding dropdown menus that should give you the opportunity to map any individual feature (i.e., protein) to any of the axes.
Here is our implementation plot for any two different type of protein.



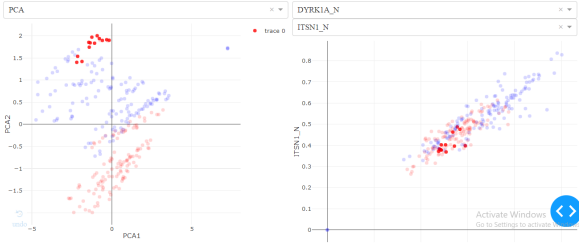
c) Implement brushing and linking: Modify the previously defined callbacks so that points belonging to a selection made in the first (dimensionality reduction) plot is highlighted in that same plot, and the corresponding samples are also highlighted in the second plot. You can use selectedData as the corresponding input for your callback functions. Note: Dash only permits a single callback per plot, so you need to modify the previously defined functions, you cannot add new ones.

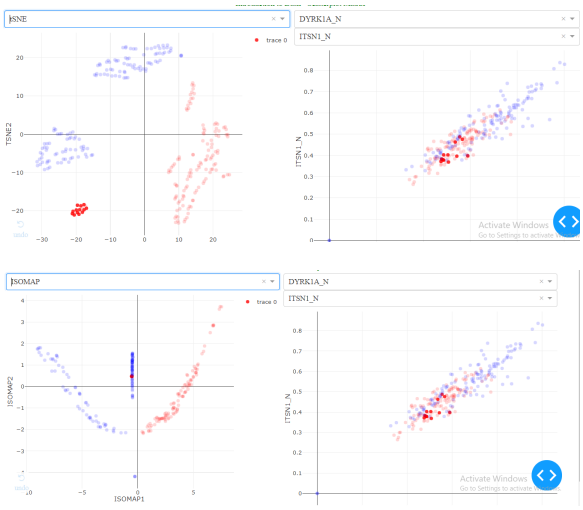
Our implementation of brushing and linking from dimensionality reduction plot to protein set plot.



d) In the PCA projection, you should notice a cluster of samples from the c-SC-s class that get mixed with samples from the t-SC-s class. Submit screenshots that show where these points project to when using ISOMAP and t-SNE. Do these nonlinear techniques manage to separate them from the other class?

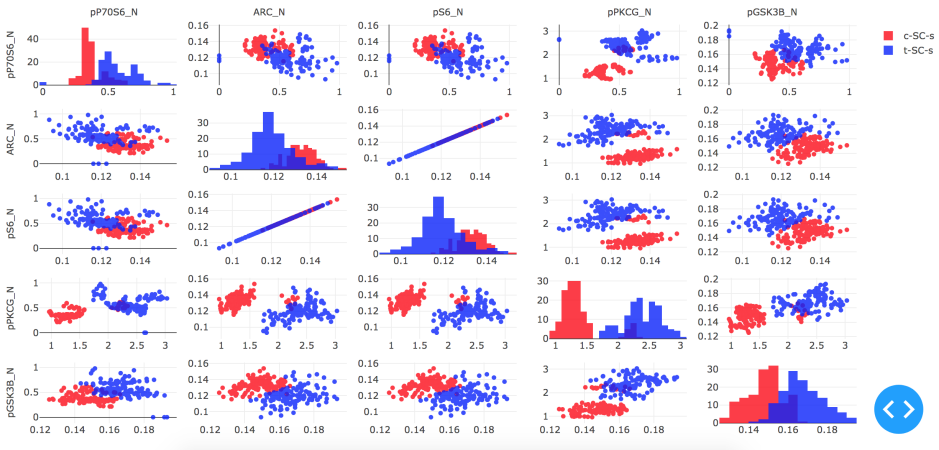
Here is our observation when we select one specific set of samples from mixed area and change it into tSNE and ISOMAP. Plot represents the changes.



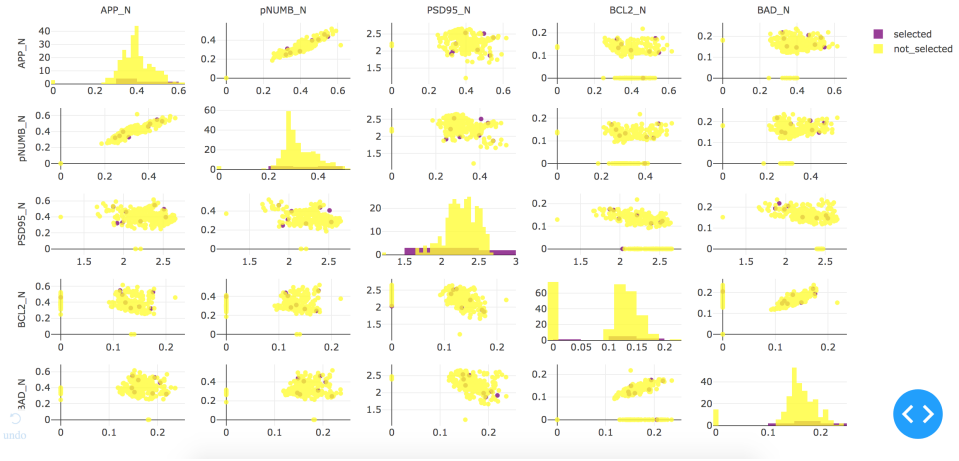


e) Add a 5 * 5 scatterplot matrix, similar to the one you made in Sheet 6. This time, it should automatically display the five most relevant attributes, as judged by the F score (Sheet 3). When no selection has been made, the F score should be computed with respect to the classes t-CS-s and c-CS-s. Whenever the user makes a selection in the dimensionality reduction plot, F scores should be computed with respect to selected vs. unselected data, and the scatterplot matrix should be updated to show the corresponding top 5 highest-ranking features. Colors in the scatterplot matrix should reflect the classification that is currently used for feature ranking.

Here is our implementation plot for scatter matrix for most relevant attributes before the user selects any points.



After the user selects some points, the scatterplot matrix changes to reflect the top 5 attributes with the highest F-score with respect to selected vs. not selected data. As an example, we show this as follows

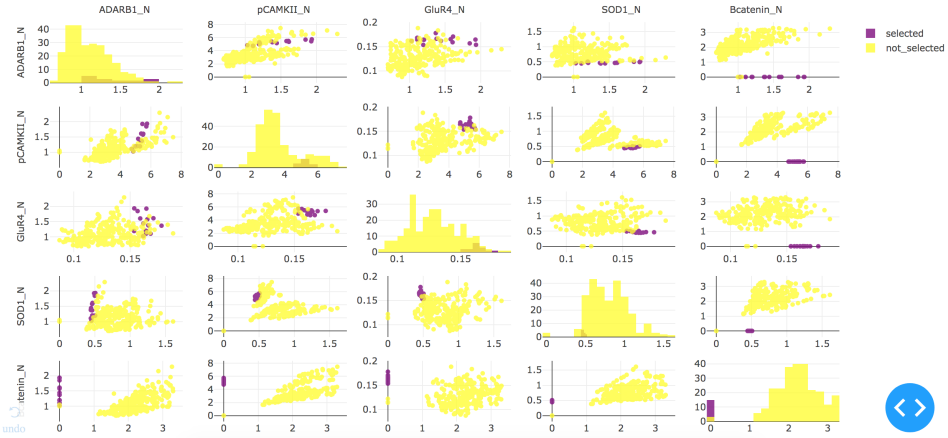


f) Select a cluster in the t-SNE embedding and report how samples within this cluster differ from the rest with respect to the expression levels of specific proteins. Submit a screenshot that illustrates your reasoning.

We first select the following cluster from t-SNE embedding

Then we look at the scatterplot matrix to see how the values of the top affecting

features are distributed.



We notice that values of protein "Bcatenin_N" has value 0 for all points in that cluster, while other points have values greater than 0 (with the exception of 2 points). Also we see values of protein "GluR4_N" has values greater than 0.15 whereas the majority of the other points have values less than 0.15. In protein "pCAMKII_N, the cluster has point values ranging only from 4 to 6 whereas the highest distribution of this protein is located in range 2 to 4. Finally, in protein "SOD1_N", the cluster values are centered around value 0.5, however the range of values for the other points for this protein is 0.5-1.

Exercise 2 (GAN Lab)

a) As explained in the paper, GANs aim to find a mapping that transforms random noise into a distribution that is as similar as possible to the distribution of a given reference dataset. Which part of the interface allows you to check visually whether this goal has been met? Which number that is displayed in the interface quantifies this?

The layered distributions view overlays the visualizations of the components from the model overview graph, so we can more easily compare the component outputs when analyzing the model. The Discriminator loss displayed in the interface quantifies this.

b) Run the training from scratch (without using the pre-trained model), with the default dataset ("mixture of Gaussians") and settings, for exactly 1000 epochs. Take a screenshot and repeat the experiment. Did you obtain the same result in both cases? Why?

For the default settings and for approximately 1000 epochs, every time we are getting different results as we are taking random noise each time. Below are the results.

c) Watch a few epochs in slow-motion mode. Focus on the gradients (pink lines) and the movement of the pink points from iteration

Figure 1: For 2D Gaussian, epochs 1008

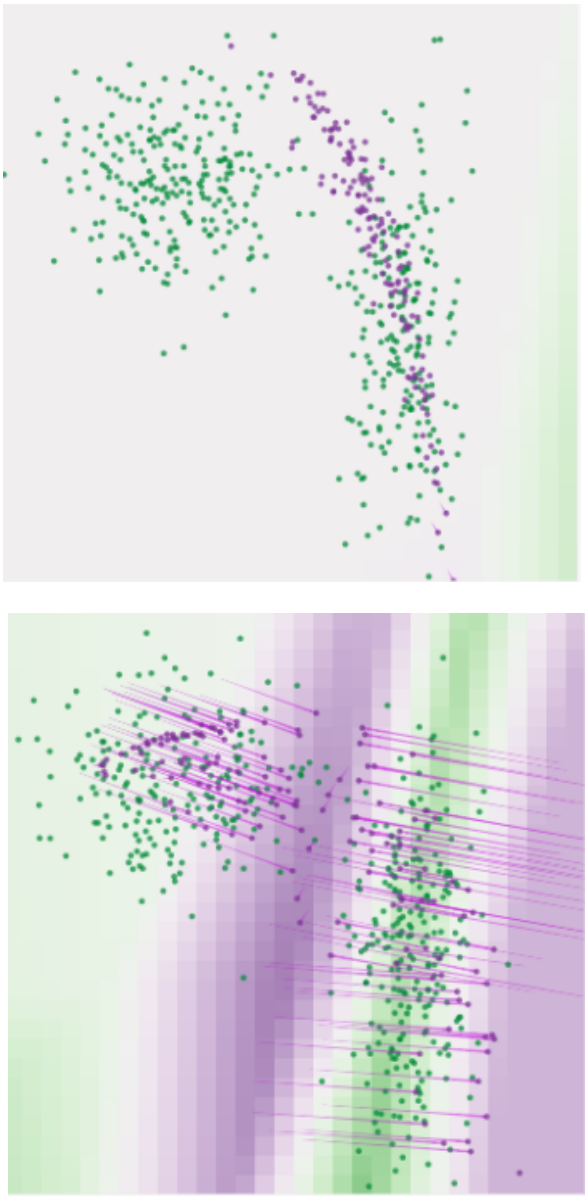


Figure 2: For 2D Gaussian, epochs 1002

to iteration. Do the pink points always follow the gradient direction? Why?

Yes, the pink points always follow the gradients, because every time the gradients are updated to minimize the distances to the real points so the pink(fake) points always follow it.

d) When training a neural network for classification, successful

training goes along with a substantial reduction in loss. However, this is not what we observe in this interface. Briefly explain why

Because, here, each time the Generator tries to create better samples and at some point the Discriminator cannot able to separate fake samples from the real ones so the Discriminator loss becomes higher.

e) Continue running the training with the default parameters, for a large number of epochs (e.g., 5000). Did it converge to a stable state? Propose a strategy to improve the convergence behavior.

We run the training with default parameters for 5010 epochs and found this result below.

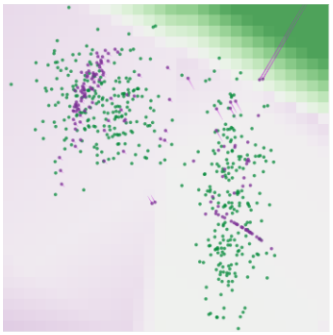


Figure 3: For 2D Gaussian, epochs 5010

Again, we run the training for 5011 epochs but this time we have changed the settings to 5 updates per epoch to the Gradients of the Discriminator side which gives us a better convergence. The figure is given below.

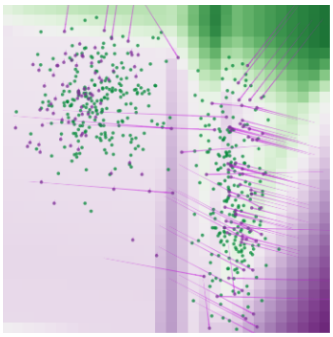


Figure 4: For 2D Gaussian, epochs 5011

f) As explained in the paper, mode collapse is a frequent problem in the training of GANs. In the "three disjoint region" dataset, it happens when all generated samples fall into a single cluster (Note: They do not necessarily have to collapse into a single point, as in the example shown in the paper) Can you reproduce this problem? Can you provoke this even in the "mixture of Gaussians"? Please submit corresponding screenshots, and briefly describe what you did.

Yes, we have reproduced this problem in 1D Gaussian and 2D Gaussian dataset. In both cases, we have increased the layer of neurons up to five for both Generator and Discriminator network. The figures are given below.

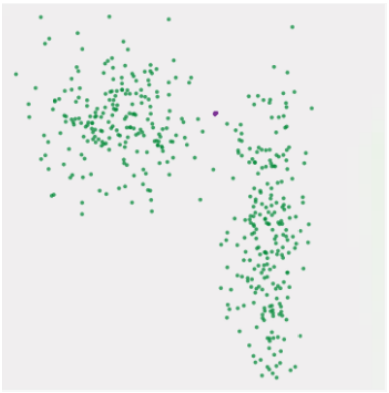


Figure 5: For 1D Gaussian

Figure 6: For 2D Gaussian