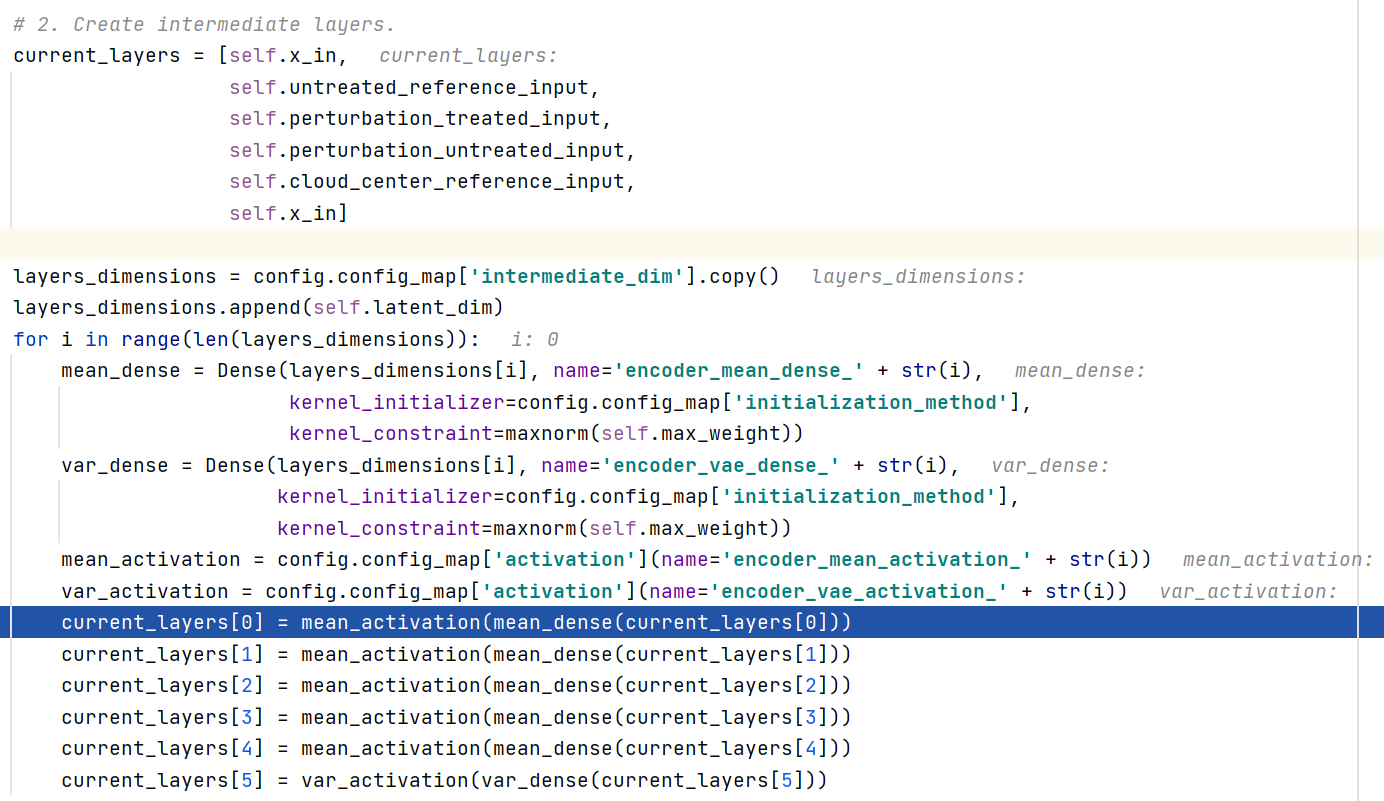
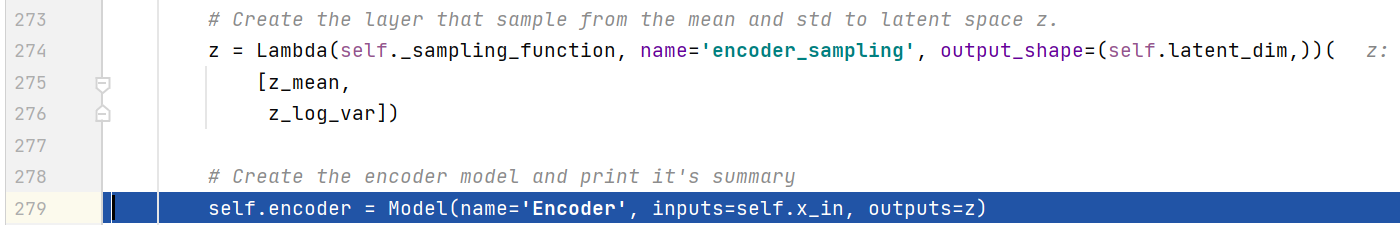
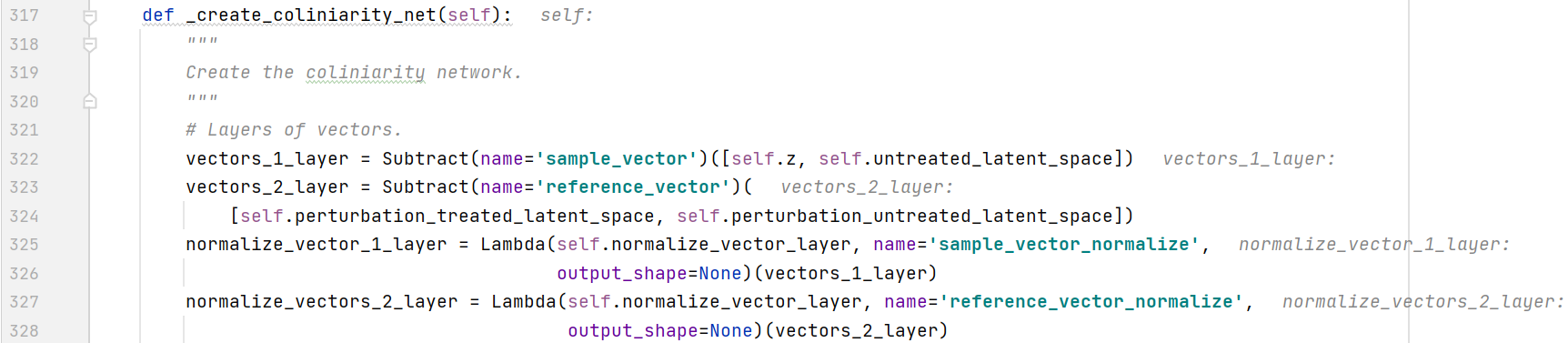
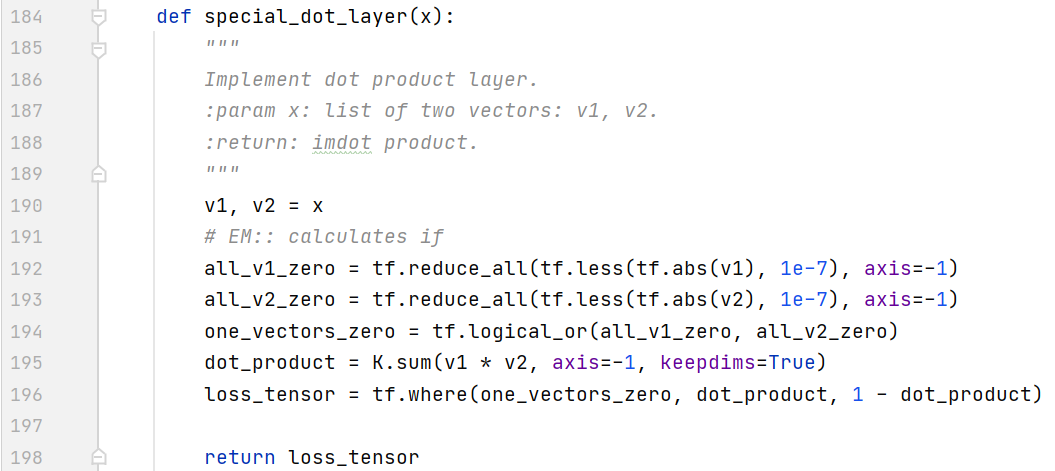
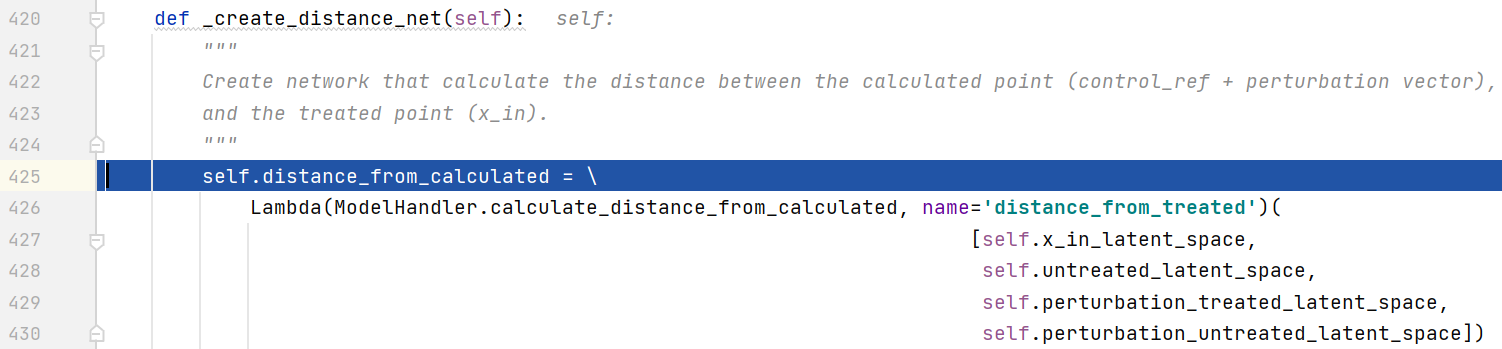
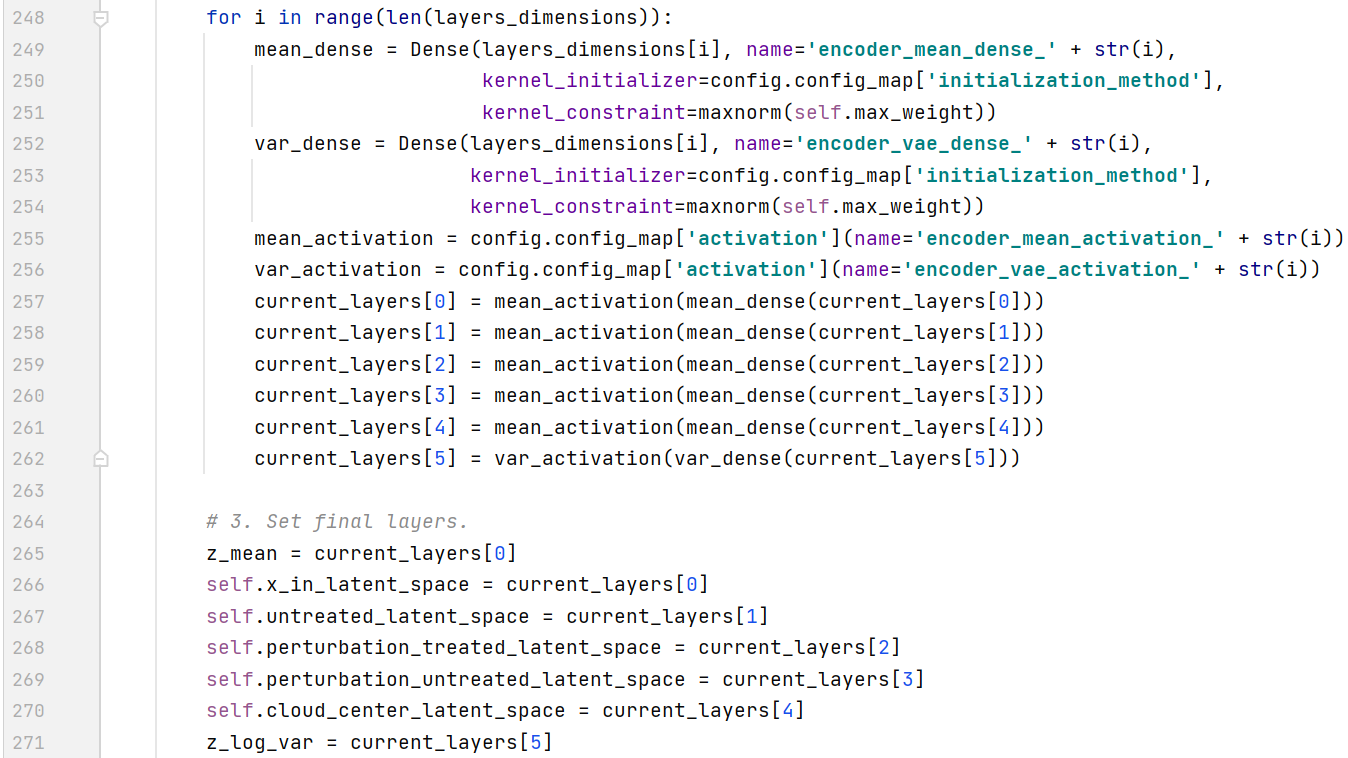
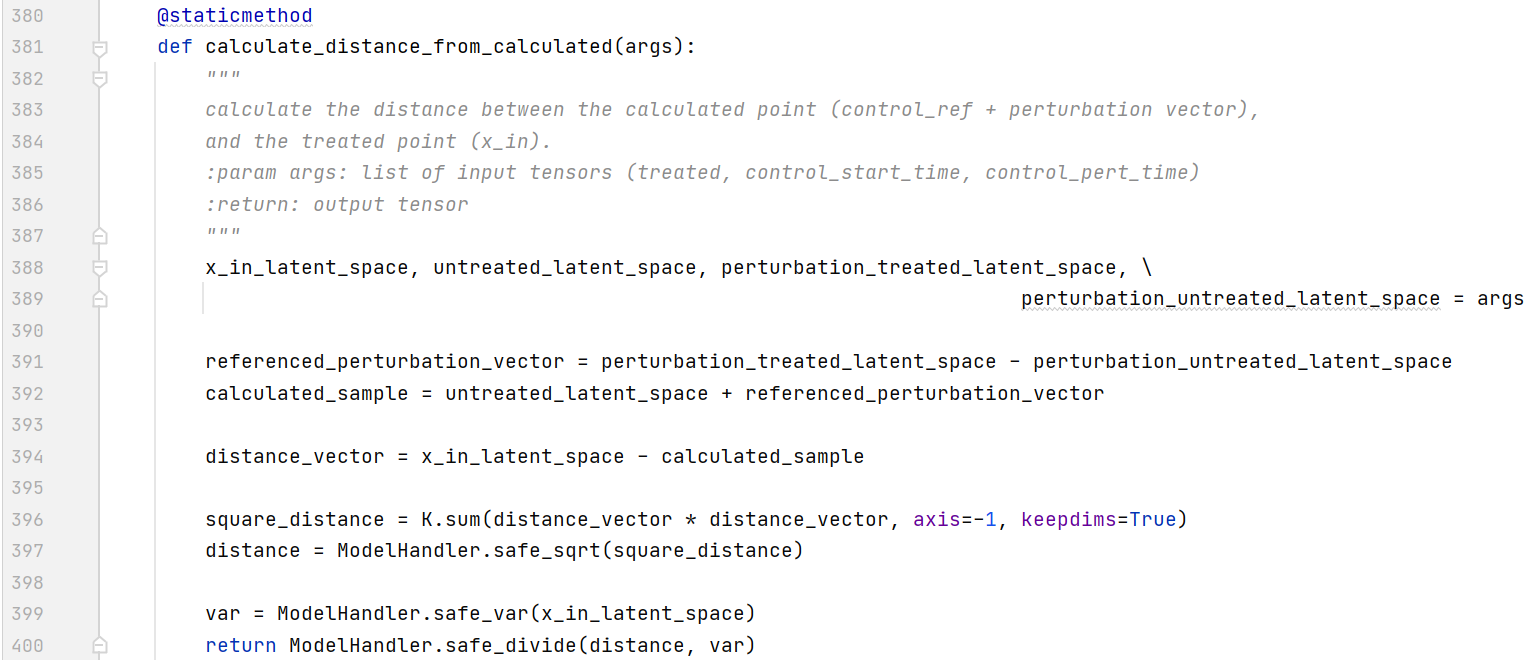
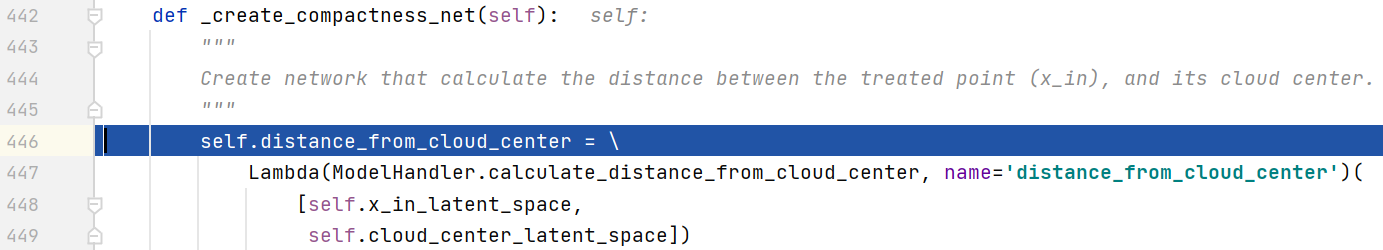
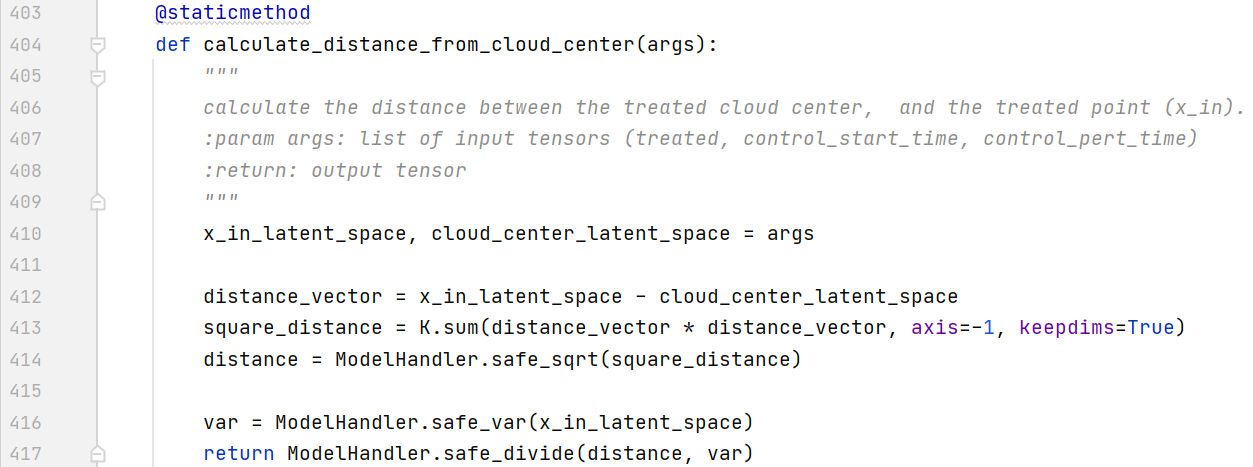
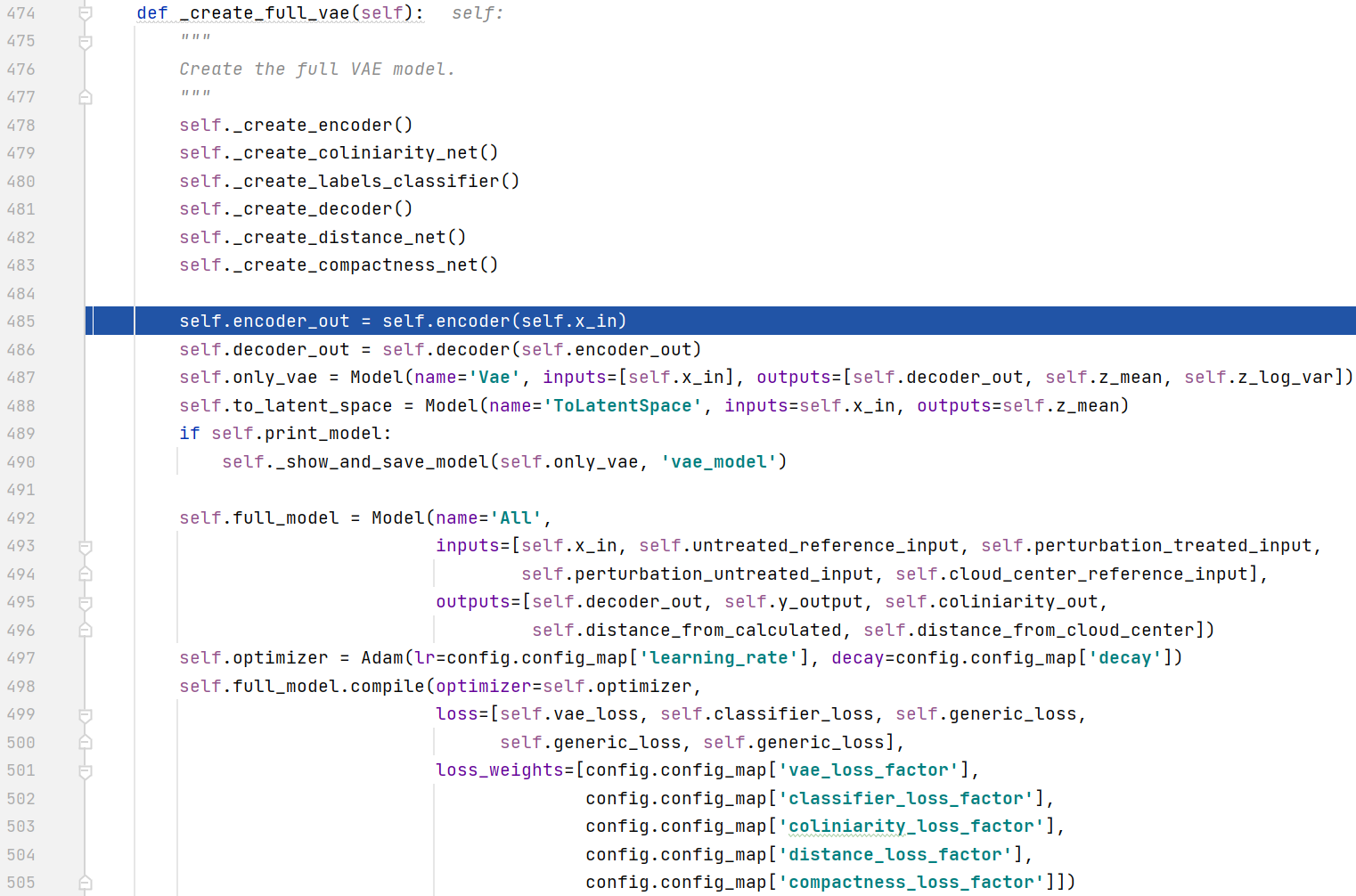
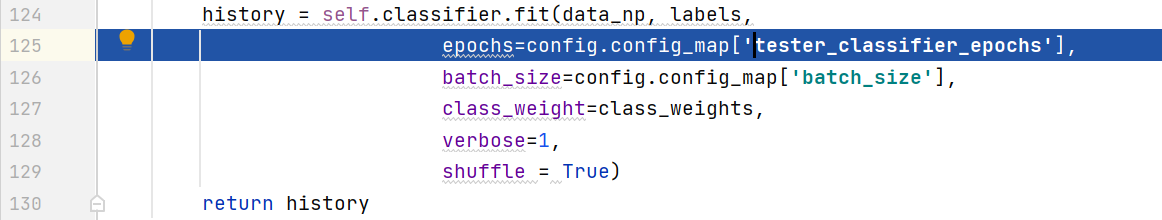
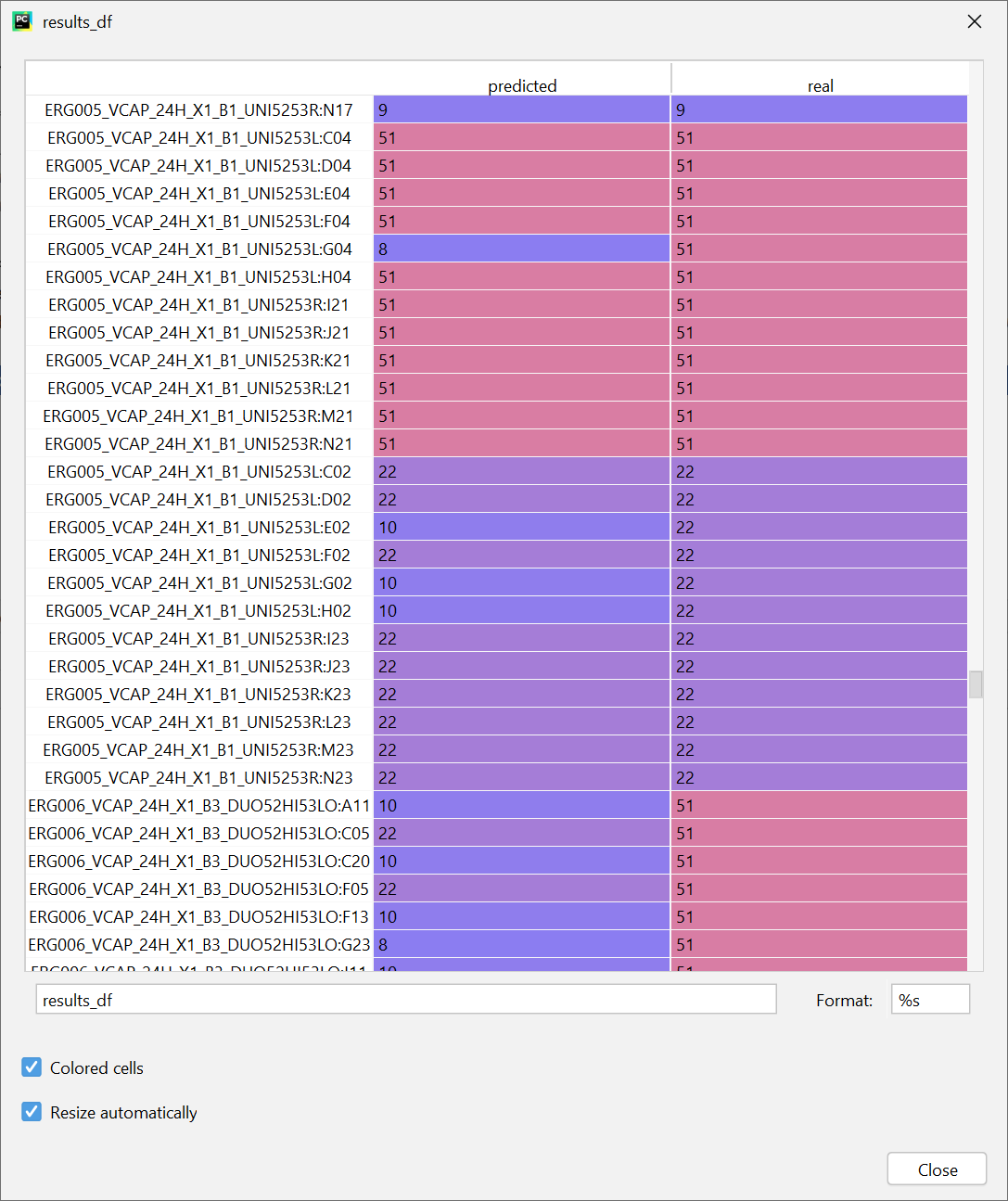
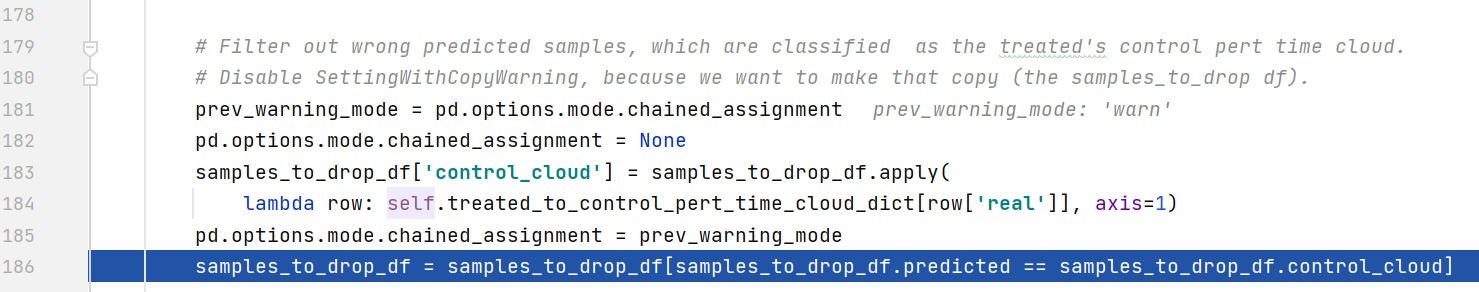
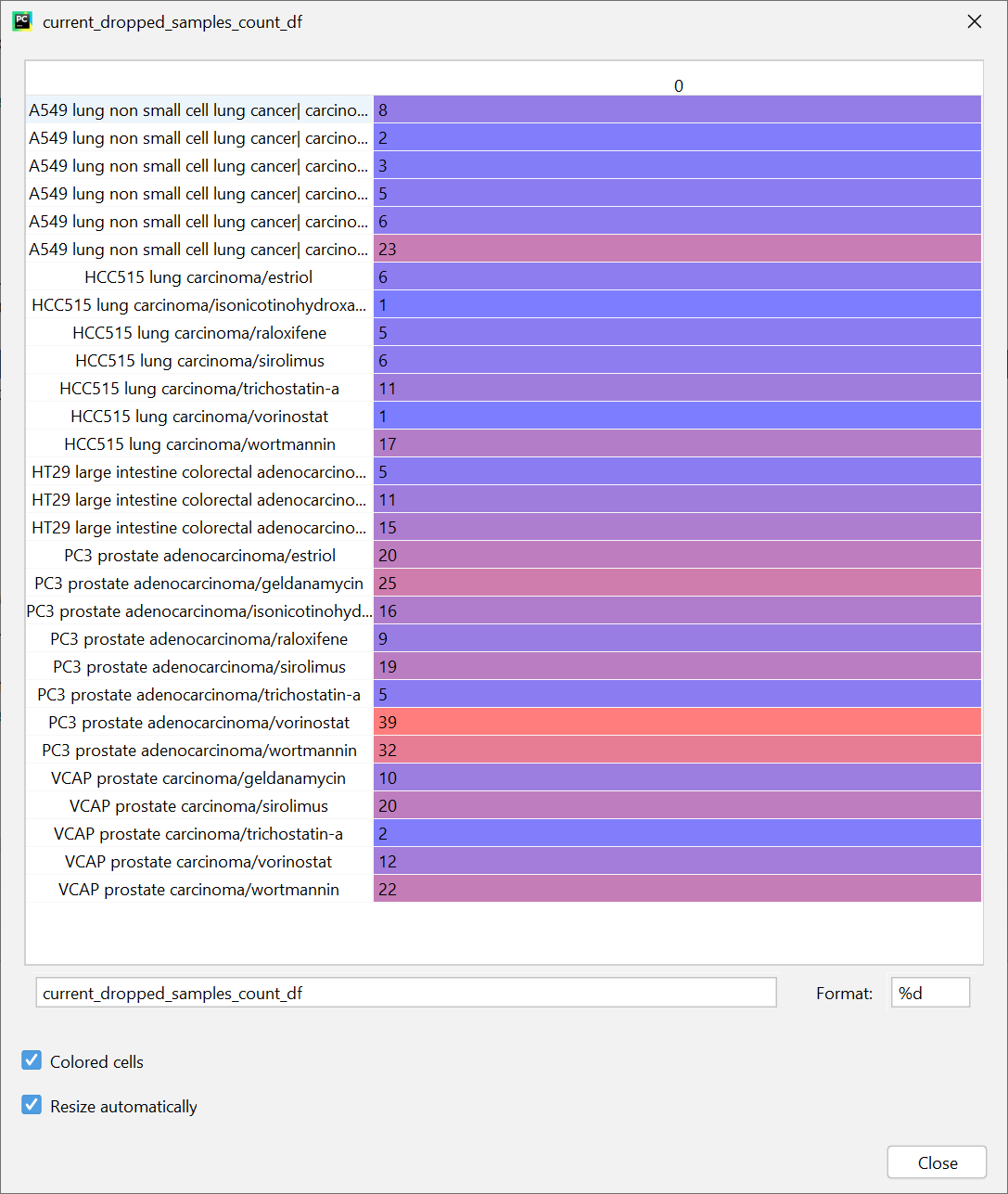
* Data handler 280: if the perturbation is DMSO, then we set the reference points to be the sample itself!
  + Or if the tissue if untreated one.
* Model handler 248: there are no ‘intermediate\_dim’ prior to the lantent dim.
* Model handler 249: there are two FC networks, one which calculates the mean, and one for the log\_var.
  + The initialization of these layers is Xavier.
  + And there is a kernel constraint, each weight can be up to 1.
  + 255 - There is PRelu after these layers!!!
* 
  + This is from the same model handler, here we calculate mean!!! For each of the layers, and calculate var only for the “x\_in” layer.
* Encoder Net:
  + 
* Co-linearity Net:
  + We calculate two vectors and normalize them, we take the sampled z:
    - 
  + We then calculate the loss by doing:
    - 
    - Notice the “1-”!!!
* Labels Classifier Net:
  + There are **no** inner dims for the classifier!
  + We do z\_mean 🡪 softmax(# of classes).
    - The softmax has Xavier + kernel\_constraint=maxnorm(self.max\_weight**🡪1**)
* Decoder Net:
  + Model handler 288.
  + Does 20 🡪 977 🡪 PRelu, yes there is a PRelu at the end of it!!!
  + There is no info about the point (z\_mean or z) yet.
  + As I understand from line 485 it is **z**.
* Distance Net:
  + We take all these point as z\_mean, not z!!!:
    - 
    - Explanation to the input layers:  
      
  + 
    - As I get it, this function returns the normalized L2 distance between the two vectors.
* Compactness Net:
  + 
  + 
* Full Model:
  + 
    - Notice that the decoder out is on the **z** and not **z\_mean**!!!
    - Vae loss is kld + recon mse.
    - Classifier loss is “0.1 \* metrics.categorical\_crossentropy(y, y\_predicted)”
    - The generic loss is just the output of the layer!
* **WE NOW FINISHED THE FULL MODEL DEFINITION!!!**
* Tester classifier net:
  + Located in “tester\_classifier.py”
  + There are no intermediate/inner dims.
  + 20 🡪 #classes 🡪 softmax (only with Xavier, not kernel constraint).
  + Loss is MSE?!!?!?!?!?
    - 
* find\_warmed\_up\_reference\_points – does the warm up
  + takes each cloud, and calculates its z\_mean using “self.to\_latent\_space.predict(samples\_df.values)”.
* \_filter\_out\_points – filters the bad points!
  + We take all the point and calculate their z\_mean.
  + We calculate for 300 epochs the tester classifier with class weights!!!
    - 
  + After that we predict on all the threated samples 🡪 NO DMSO samples.
  + Then we take the max value of the softmax vector “np.argmax(prediction\_df.values, axis=-1)”
  + After that we use sklearn.classification.accuracy\_score to calculate the accuracy, which is 50% in this test!!!!!!!!!!!
    - 
    - ?!?!?!?!?!!?
  + 
  + We filter out only the points that are classified as the DMSO!
    - 
  + The dropped samples are:
    - 
  + 
  + After dropping the points, we get the reference points again by using “self.get\_reference\_points\_to\_data\_slice(self.train\_data\_df.index)”.