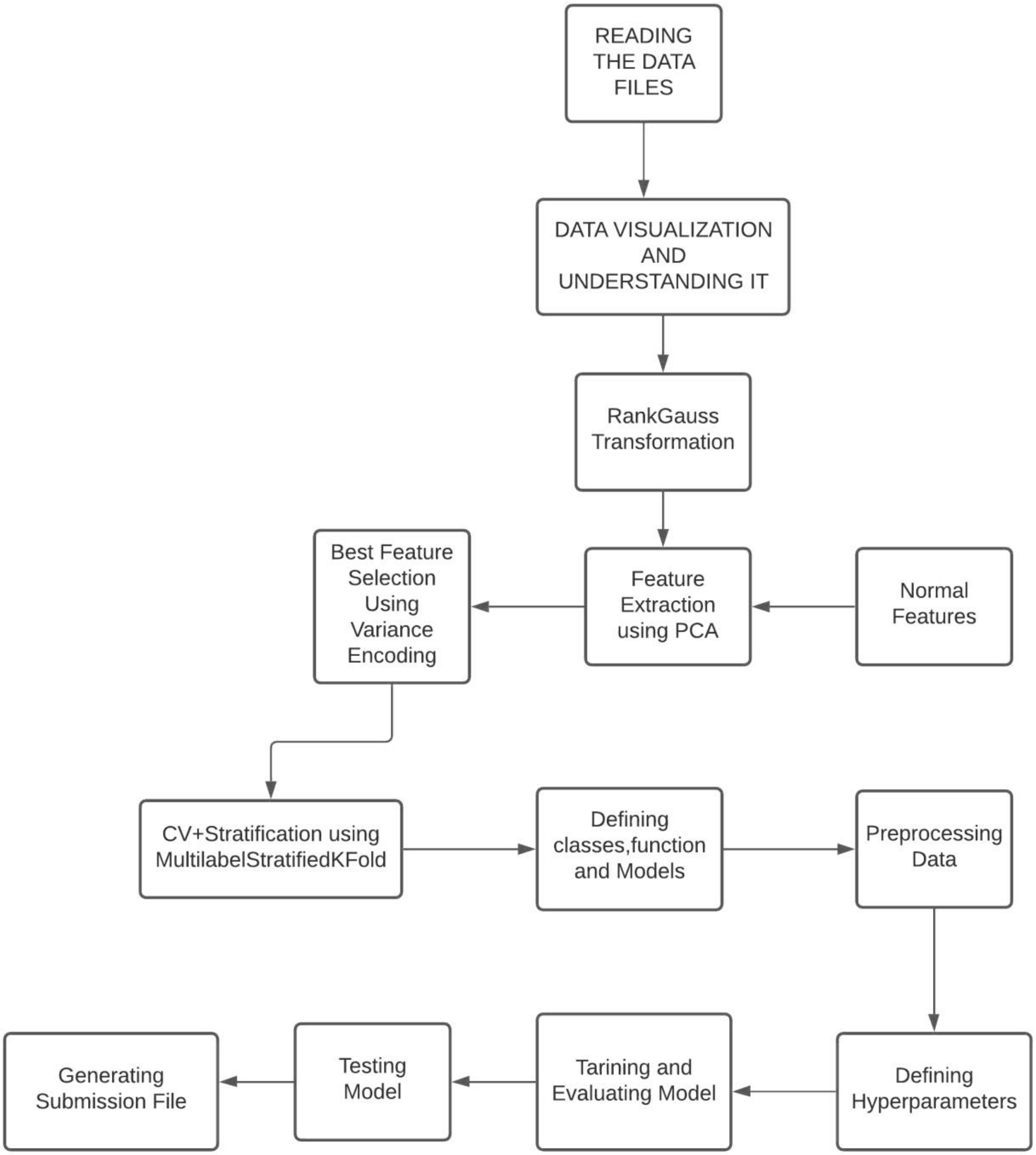
“[ENPM808A] FINAL PROJECT”

Introduction to Machine Learning

PROJECT REPORT

NEVIL PATEL:- 116897068

PROJECT PIPELINE


READING THE DATA FILES
DATA VISUALIZATION AND UNDERSTANDING IT
RankGauss Transformation
Best Feature Selection Using Variance Encoding
Feature Extraction using PCA
Normal Features
CV+Stratification using MultilabelStratifiedKFold
Defining classes,function and Models
Preprocessing Data
Generating Submission File
Testing Model
Tarining and Evaluating Model
Defining Hyperparameters

KAGGLE MOA COMPETITION

The Connectivity Map, a project within the Broad Institute of MIT and Harvard, the Laboratory for Innovation Science at Harvard (LISH), and the NIH Common Funds Library of Integrated Network-Based Cellular Signatures (LINCS), present this challenge with the goal of advancing drug development through improvements to MoA prediction algorithms.

What is the Mechanism of Action (MoA) of a drug? And why is it important?

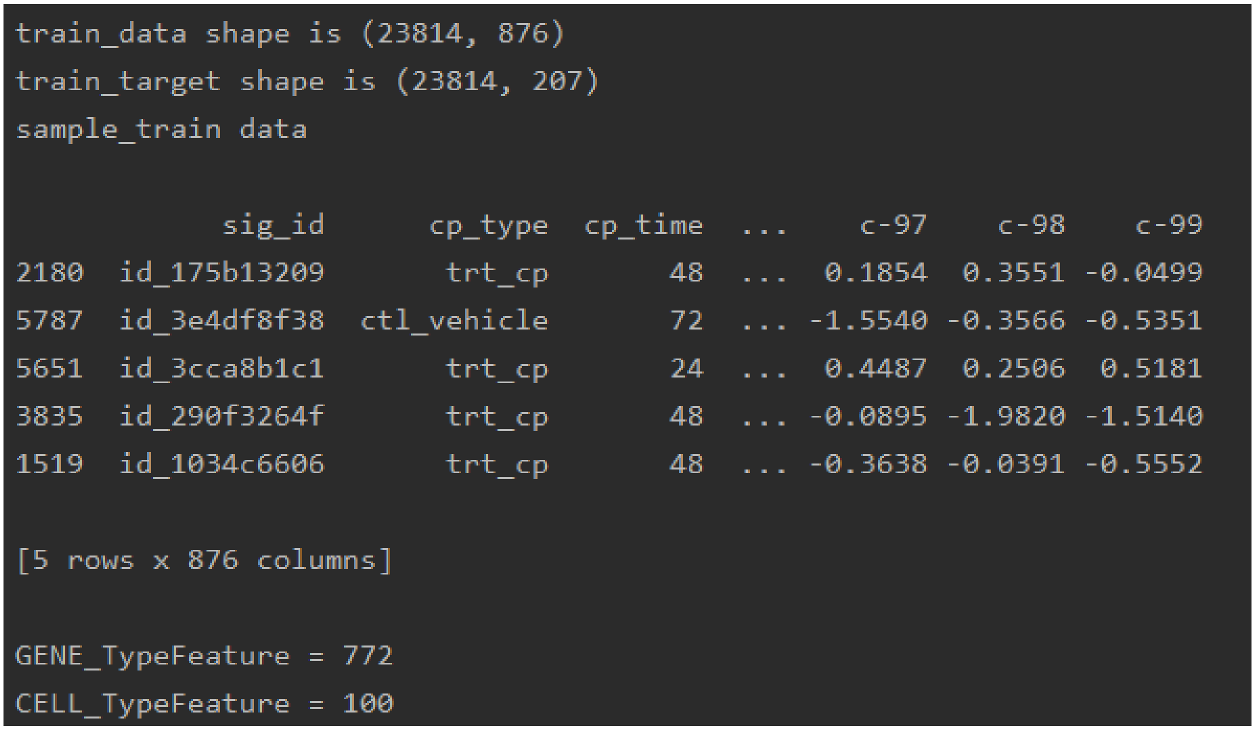
In the past, scientists derived drugs from natural products or were inspired by traditional remedies. Very common drugs, such as paracetamol, known in the US as acetaminophen, were put into clinical use decades before the biological mechanisms driving their pharmacological activities were understood. Today, with the advent of more powerful technologies, drug discovery has changed from the serendipitous approaches of the past to a more targeted model based on an understanding of the underlying biological mechanism of a disease. In this new framework, scientists seek to identify a protein target associated with a disease and develop a molecule that can modulate that protein target. As a shorthand to describe the biological activity of a given molecule, scientists assign a label referred to as mechanism-of-action or MoA for short.

READING MOA DATASET

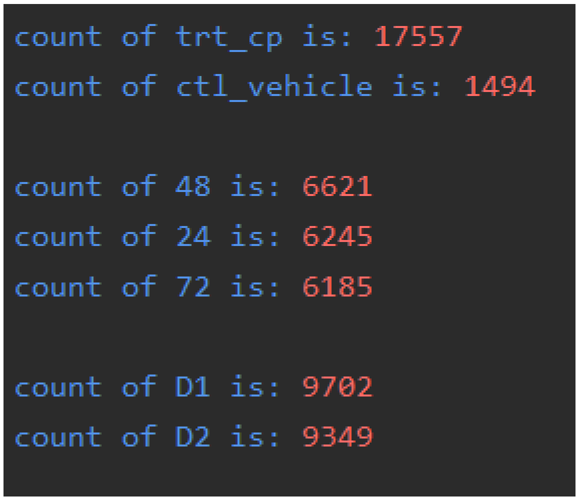
train_features.csv: Features for the training set. Features g- signify gene expression data, and c- signify cell viability data. cp_type indicates samples treated with a compound (cp_vehicle) or with a control perturbation (ctrl_vehicle); control perturbations have no MoAs; cp_time and cp_dose indicate treatment duration (24, 48, 72 hours) and dose (high or low).
cp_type (categorical): Samples treated with a compound or with a control perturbation. Categories include "trt_cp" and "ctl_vehicle", respectively. There is no MoA for "ctl_vehicle".
cp_time (categorical): Treatment duration in hours. Categories include 24, 48, and 72 hours.
cp_dose (categorical): Drug dose. Categories include "D1", "D2" for low and high dose, respectively.
g-[0-771] (continous): Gene expression data - a measure of activation in a given gene after the drug is applied.
c-[0-99] (continous): Cell viability. Basically count of live cells after the drug is applied.
train_targets_scored.csv: The binary MoA targets that are scored. There are 206 MoA targets.
test_features.csv: Features for the test data. We must predict the probability of each scored MoA for each row in the test data.

You might have your data in .csv files or SQL tables. Maybe Excel files. Or .tsv files. Or something else. But the goal is the same in all cases. If you want to analyze that data using **Pandas**, the first step will be to read it into a data structure that's compatible with pandas. There are two types of data structures in pandas: **Series** and **DataFrames**. Our project had the type dataFrames.

DATA UNDERSTANDING AND VISUALIZATION:-

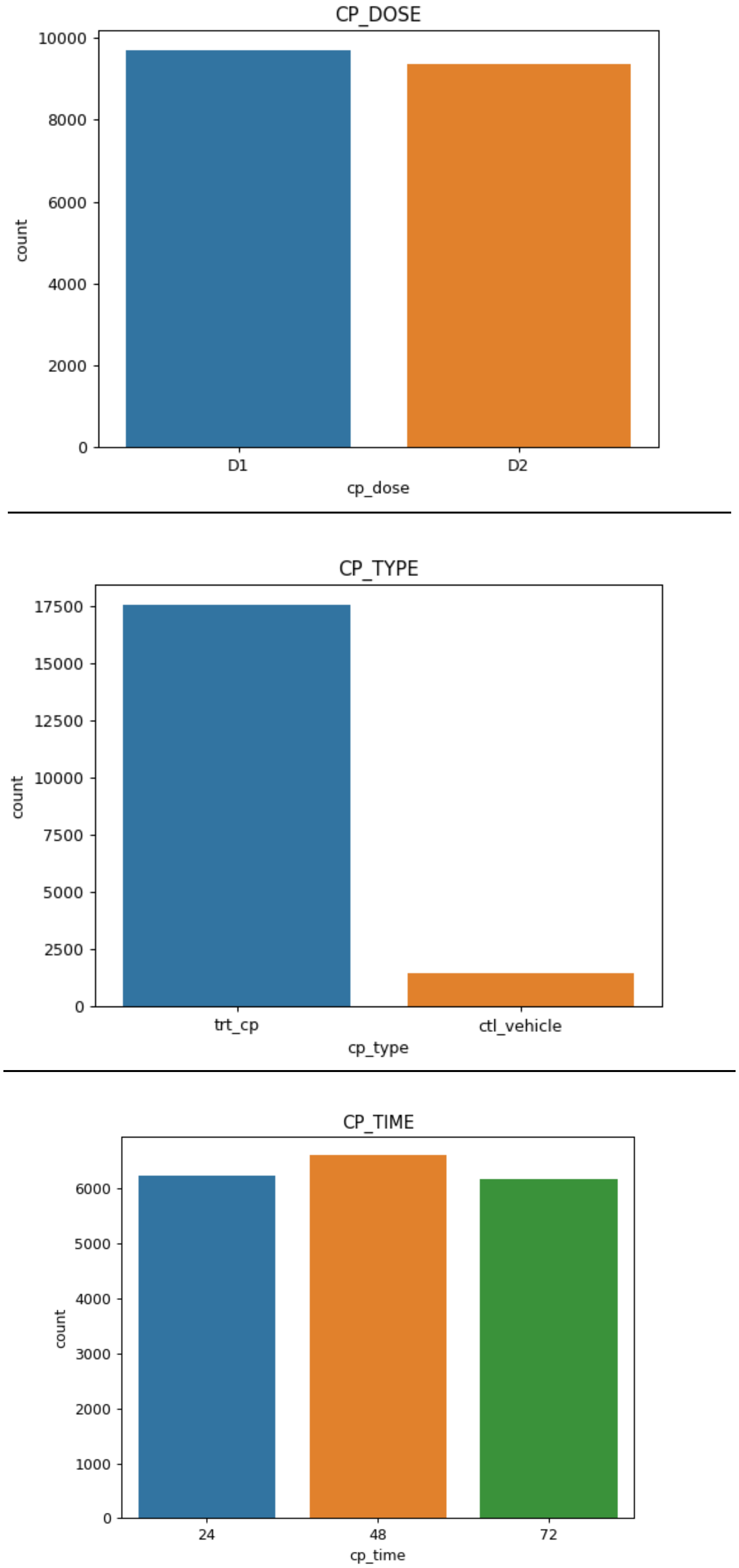


```
train_data shape is (23814, 876)
train_target shape is (23814, 207)
sample_train data

            sig_id      cp_type  cp_time  ...     c-97    c-98    c-99
2180  id_175b13209       trt_cp       48  ...   0.1854  0.3551 -0.0499
5787  id_3e4df8f38  ctl_vehicle       72  ...  -1.5540 -0.3566 -0.5351
5651  id_3cca8b1c1       trt_cp       24  ...   0.4487  0.2506  0.5181
3835  id_290f3264f       trt_cp       48  ...  -0.0895 -1.9820 -1.5140
1519  id_1034c6606       trt_cp       48  ...  -0.3638 -0.0391 -0.5552

[5 rows x 876 columns]

GENE_TypeFeature = 772
CELL_TypeFeature = 100
```

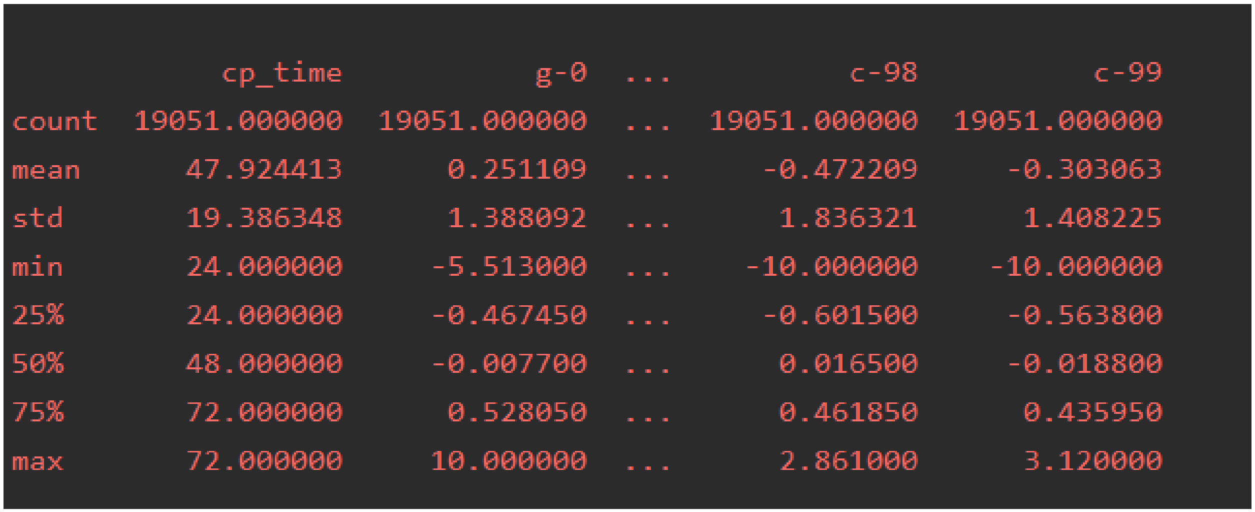


```
count of trt_cp is: 17557
count of ctl_vehicle is: 1494

count of 48 is: 6621
count of 24 is: 6245
count of 72 is: 6185

count of D1 is: 9702
count of D2 is: 9349
```

COUNT ON TRAINING DATA

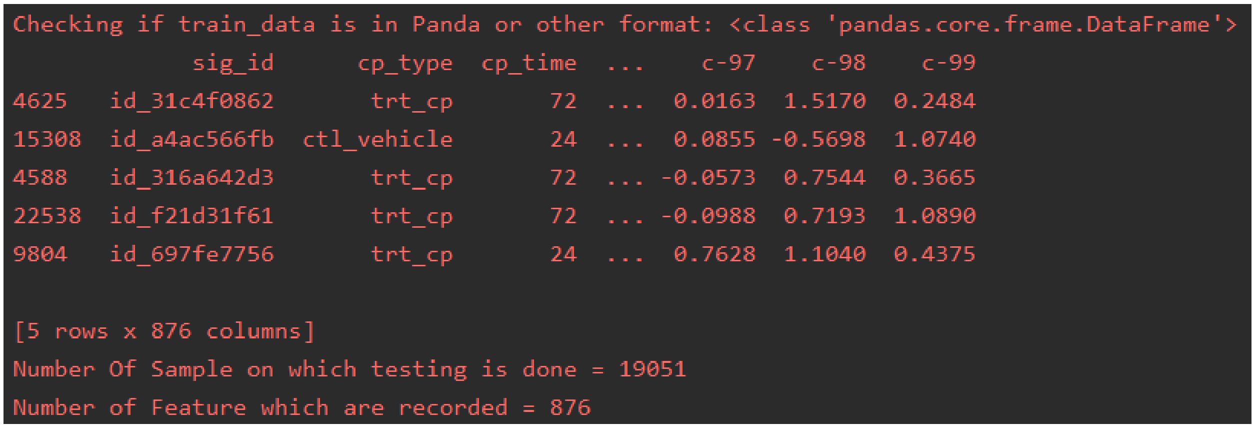


<u>HISTOGRAMS OF THE VALUES OF DOSE,TREATED COMPUND AND TREATMENT TIME</u>



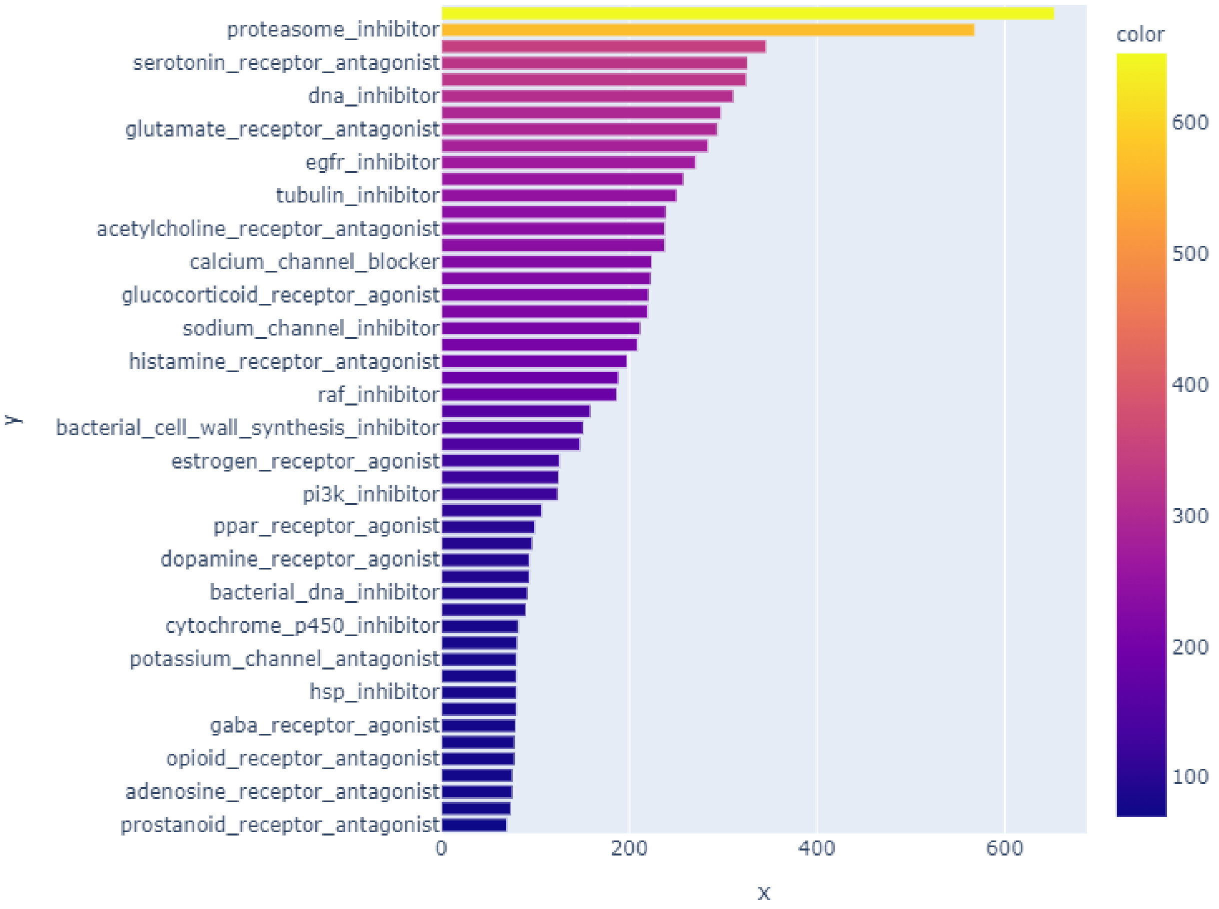
```
           cp_time           g-0  ...           c-98          c-99
count  19051.000000  19051.000000  ...  19051.000000  19051.000000
mean      47.924413      0.251109  ...     -0.472209     -0.303063
std       19.386348      1.388092  ...      1.836321      1.408225
min       24.000000     -5.513000  ...    -10.000000    -10.000000
25%       24.000000     -0.467450  ...     -0.601500     -0.563800
50%       48.000000     -0.007700  ...      0.016500     -0.018800
75%       72.000000      0.528050  ...      0.461850      0.435950
max       72.000000     10.000000  ...      2.861000      3.120000
```

DESCRIPTION OF THE TRAINING DATASET



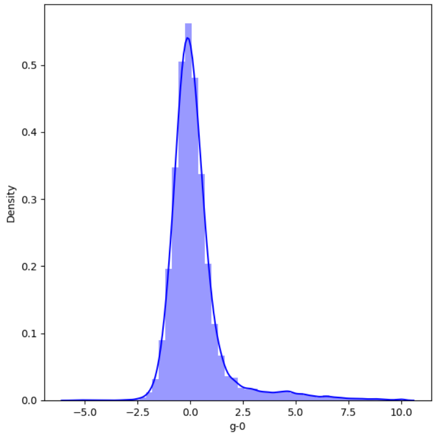
```
Checking if train_data is in Panda or other format: <class 'pandas.core.frame.DataFrame'>
             sig_id      cp_type  cp_time  ...    c-97    c-98    c-99
4625   id_31c4f0862       trt_cp       72  ...  0.0163  1.5170  0.2484
15308  id_a4ac566fb  ctl_vehicle       24  ...  0.0855 -0.5698  1.0740
4588   id_316a642d3       trt_cp       72  ... -0.0573  0.7544  0.3665
22538  id_f21d31f61       trt_cp       72  ... -0.0988  0.7193  1.0890
9804   id_697fe7756       trt_cp       24  ...  0.7628  1.1040  0.4375

[5 rows x 876 columns]
Number Of Sample on which testing is done = 19051
Number of Feature which are recorded = 876
```

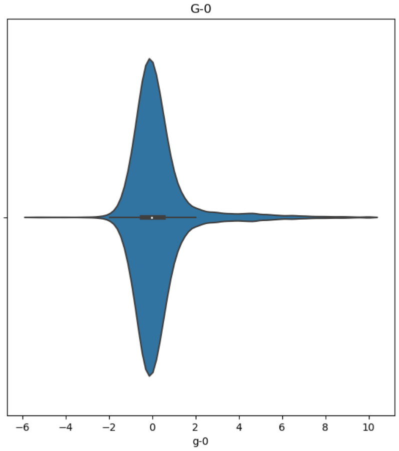
DISPLAYING TRAIN DATA AND IT'S SIZE

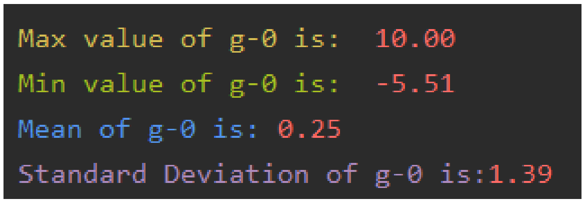


TARGET DATA SCORED

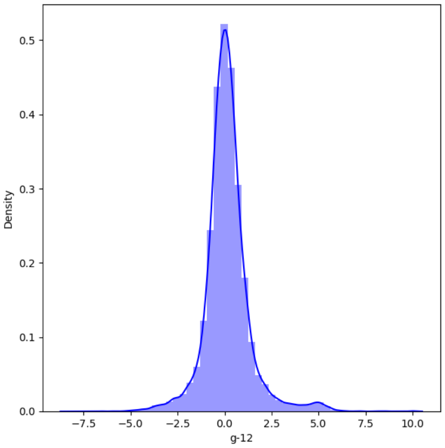
DISPLAYING GENE AND CELL FEATURES(TRAINING DATA)

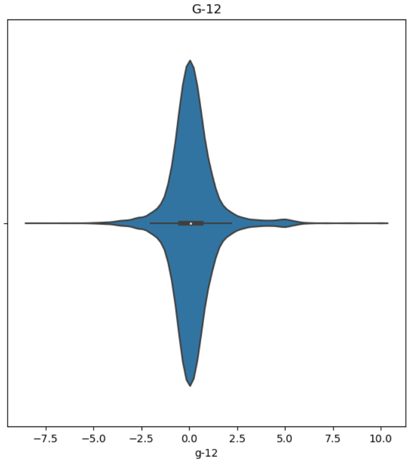


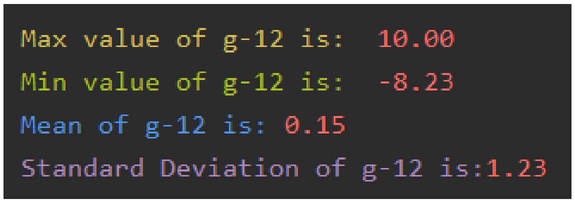




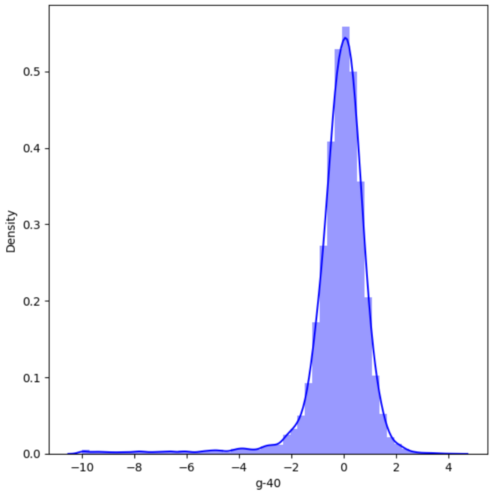
```
Max value of g-0 is:  10.00
Min value of g-0 is:  -5.51
Mean of g-0 is: 0.25
Standard Deviation of g-0 is:1.39
```

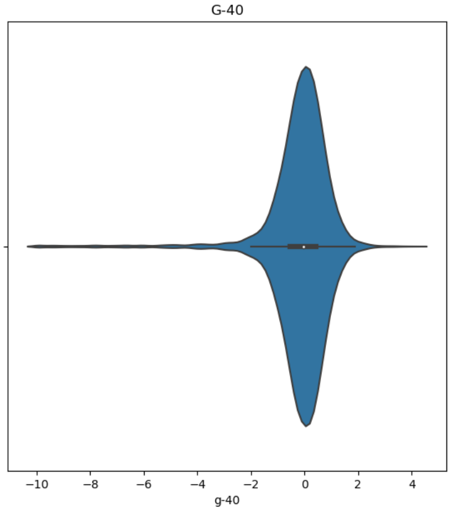


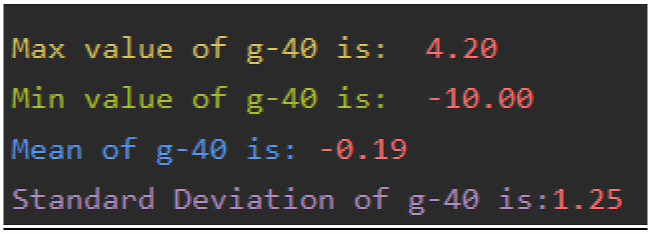




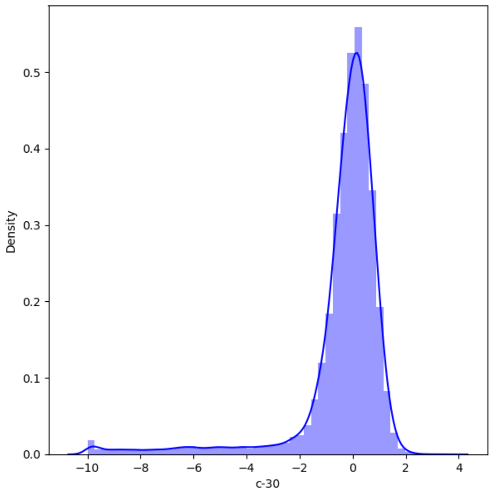
```
Max value of g-12 is:  10.00
Min value of g-12 is:  -8.23
Mean of g-12 is: 0.15
Standard Deviation of g-12 is:1.23
```

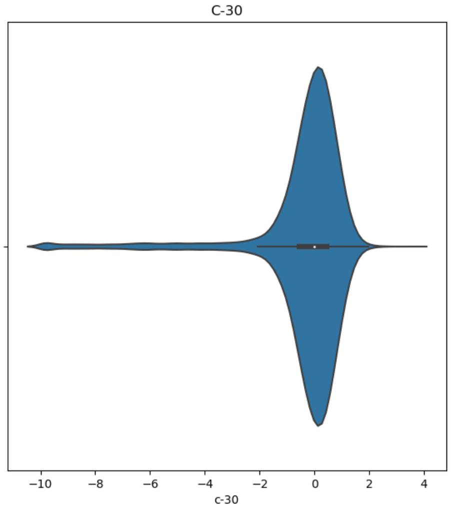


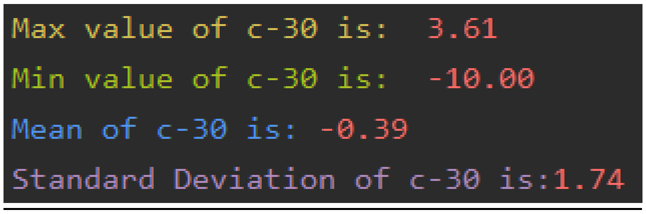




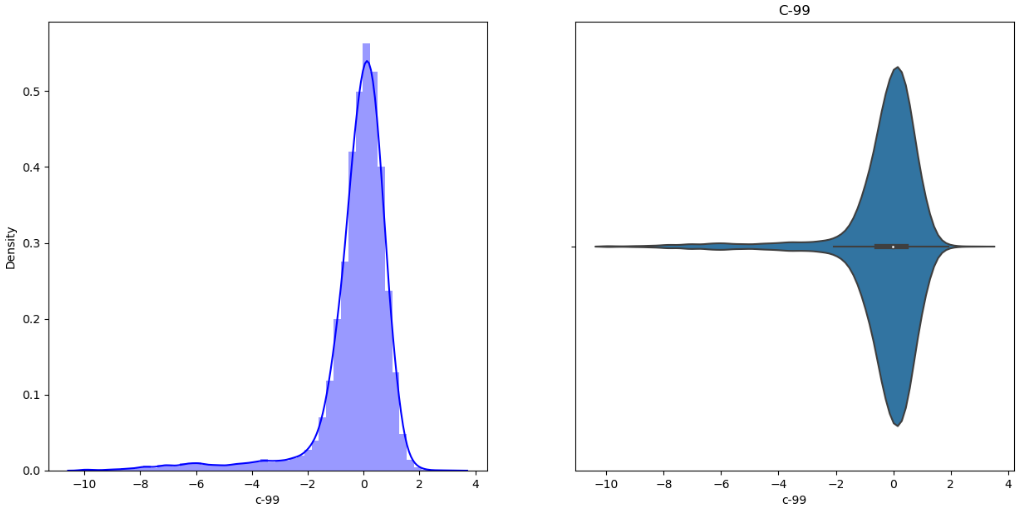
```
Max value of g-40 is:  4.20
Min value of g-40 is:  -10.00
Mean of g-40 is: -0.19
Standard Deviation of g-40 is:1.25
```

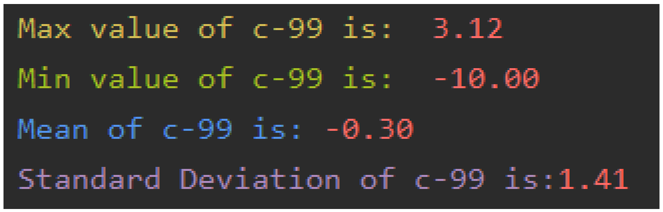






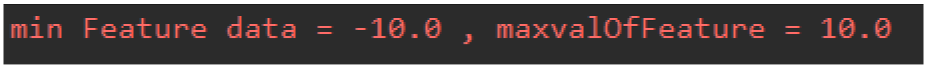
```
Max value of c-30 is:  3.61
Min value of c-30 is:  -10.00
Mean of c-30 is: -0.39
Standard Deviation of c-30 is:1.74
```





```
Max value of c-99 is:  3.12
Min value of c-99 is:  -10.00
Mean of c-99 is: -0.30
Standard Deviation of c-99 is:1.41
```

VISULISATION CONCLUSION(MIN AND MAX VALUE OF FEATURES)



```
min Feature data = -10.0 , maxvalOfFeature = 10.0
```

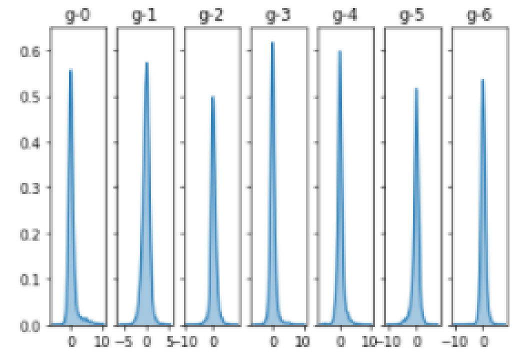
ALL OF THE FEATURES ARE IN RANGE OF -10 TO +10

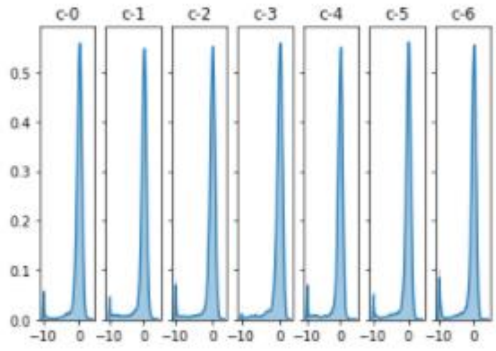
RANKGAUSS TRANSFORMATION:-

Input normalization for gradient-based models such as neural nets is critical. For lightgbm/xgb it does not matter.The best what I found during the past and works straight of the box is "RankGauss". It is based on rank transformation.
This works usually much better than standard mean/std scaler or min/max. Here after checking distributions of g- *and c*- of train and test set. They are spiky distribution rather than normal distribution. Regardless of the train and test, they look be in the same shape. It may be a too simple idea, it appears that the gene expression data and cell viability data can be controlled by the experimenter, so it is safe to assume that these data are independent of each other.Also, since the shape of the distribution is close to normal distribution to begin with, I don't think there is much of a problem if it is forced to be transformed into a Gaussian distribution. We can confirme that the shapes of data got close to the normal distribution. It appears that we were able to transform the distribution of each data to resemble a normal distribution, as intended. So, let's enter the data into the benchmarking method to see the improvement.

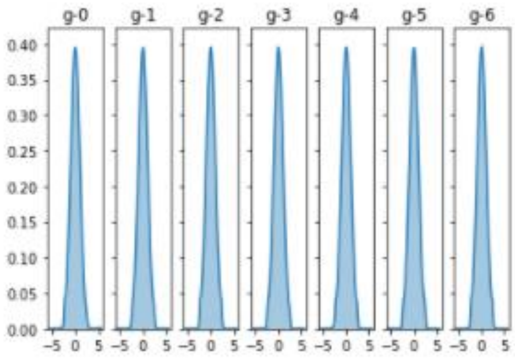
Train Set :

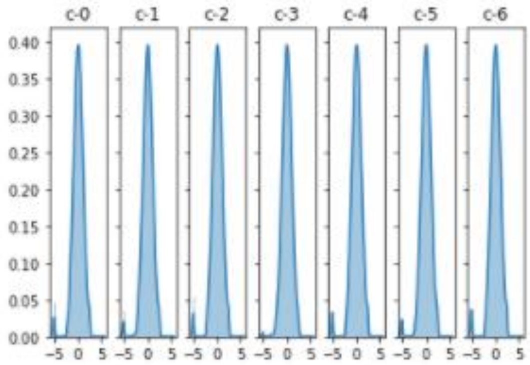
<u>Before</u>





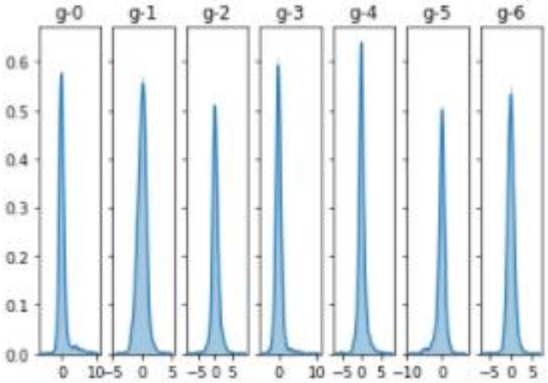
<u>After</u>

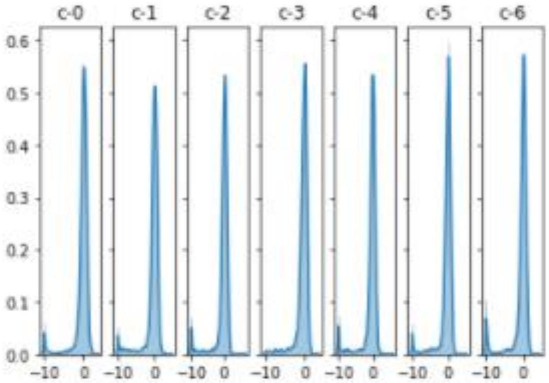




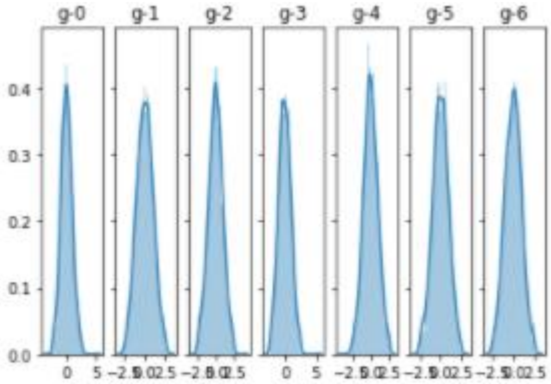
Test Set :

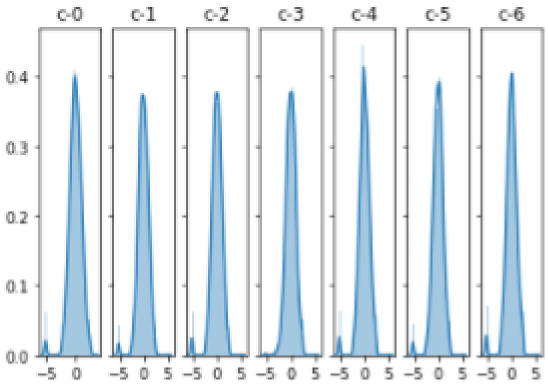
Before





After



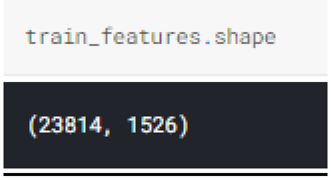


FEATURE EXTRACTION USING PCA AND ADDING TO NORMAL FEATURES:-

Principle Component Analysis (PCA) is a common feature extraction method in data science. Technically, PCA finds the eigenvectors of a covariance matrix with the highest eigenvalues and then uses those to project the data into a new subspace of equal or less dimensions. Practically, PCA converts a matrix of n features into a new dataset of (hopefully) less than n features. That is, it reduces the number of features by constructing a new, smaller number variables which capture a significant portion of the information found in the original features.

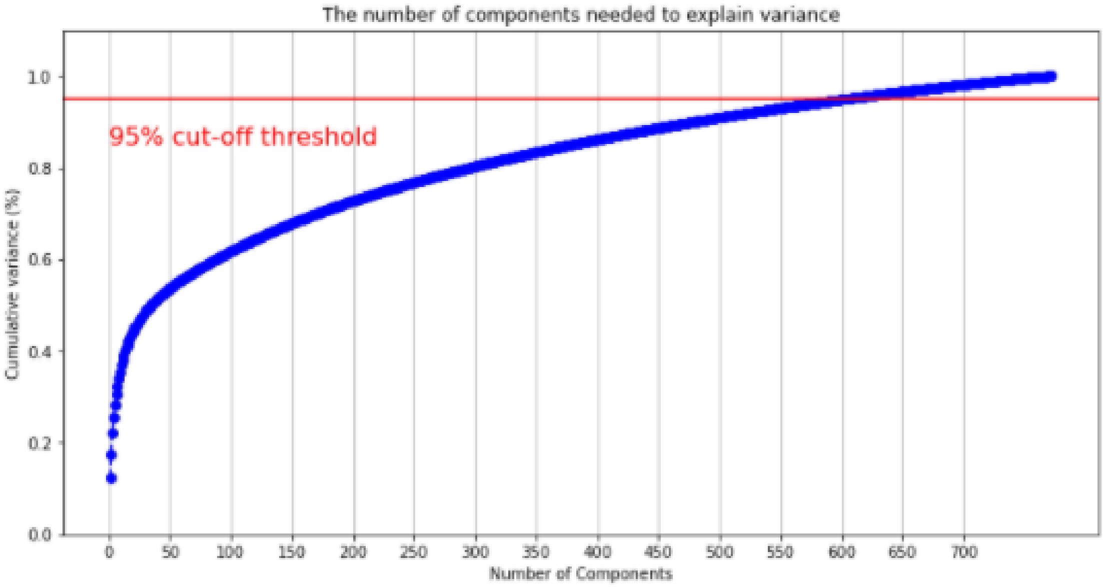
Don’t choose the number of components manually.Instead of that, use the option that allows you to set the variance of the input that is supposed to be explained by the generated components. Remember to scale the data to the range between 0 and 1 before using PCA that is done during RankGauss Transformation. Typically, we want the explained variance to be between 95–99%. **In our MOA case, to get 95% of variance explained we need 600 principal components.**

Size of New train Features after PCA



```
train_features.shape
```

```
(23814, 1526)
```



FEATURE SELECTION USING VARIANCE ENCODING:-

This feature selection algorithm looks only at the features (X), not the desired outputs (y), and can thus be used for unsupervised learning. Variance Threshold is a simple baseline approach to feature selection. It removes all features whose variance doesn't meet some threshold. By default, it removes all zero-variance features, i.e. features that have the same value in all samples.

As an example, suppose that we have a dataset with Boolean features, and we want to remove all features that are either one or zero (on or off) in more than 80% of the samples. Boolean features are Bernoulli random variables, and the variance of such variables is given by

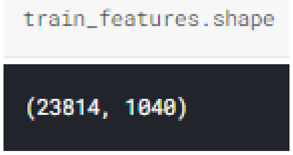
$\mathrm{Var}[X]=p(1-p)$

So we can select using the threshold .8 * (1 - .8):

Here we have used 0.8 as variance threshold.

```
var_thresh = VarianceThreshold(0.8)
```

Size of New train Features after Variance Encoding



```
train_features.shape
```

```
(23814, 1040)
```

CV FOLDS:-

Cross-validation, how I see it, is the idea of minimizing randomness from one split by makings n folds, each fold containing train and validation splits. You train the model on each fold, so you have n models. Then you take average predictions from all models, which supposedly give us more confidence in results. These we will see in following code. I found iterative-stratification package that provides scikit-learn compatible cross validators with stratification for multilabel data.

mskf = MultilabelStratifiedKFold(n_splits=7)

CLASSES AND MODEL(PYTORCH):-

PyTorch is the **fastest growing** Deep Learning framework and it is also used by **Fast.ai** in its MOOC, Deep Learning for Coders and its library.

PyTorch is also very *pythonic*, meaning, it feels more natural to use it if you already are a Python developer.

Besides, using PyTorch may even *improve your health*,

CLASSES:-

DATASET:-

Here the class Dataset is used to convert numpy type data to tensor data format. That's what **from_numpy** is good for. It returns a **CPU/GPU tensor.** If you compare the **types** of both variables, you'll get what you'd expect: numpy.ndarray for the first one and torch.Tensor for the second one.

TRAIN:-

model.train() tells your model that you are training the model. So effectively layers like dropout, batchnorm etc. which behave different on the train and test procedures know what is going on and hence can behave accordingly. Our train function then goes to optimizer, Loss and scheduler.

Adam Optimizer:-

We use one of PyTorch's **optimizers Adam**. An optimizer takes the **parameters** we want to update, the **learning rate** we want to use (and possibly many other hyper-parameters as well!) and **performs the updates** through its **step()** method Besides, we also don't need to zero the gradients one by one anymore. We just invoke the optimizer's **zero_grad()** method and that's it! **Adam** is a replacement optimization algorithm for stochastic gradient descent for training deep learning models. **Adam** combines the best properties of the AdaGrad and RMSProp algorithms to provide an optimization algorithm that can handle sparse gradients on noisy problems.

```
    optimizer = torch.optim.Adam(model.parameters(), lr=LEARNING_RATE, weight_dec
ay=WEIGHT_DECAY)
```

BCE with Logitloss:-

This loss combines a *Sigmoid* layer and the *BCELoss* in one single class. This version is more numerically stable than using a plain *Sigmoid* followed by a *BCELoss* as, by combining the operations into one layer, we take advantage of the log-sum-exp trick for numerical stability.

The unreduced (i.e. with reduction set to 'none') loss can be described as:

$$\ell(x, y) = L = \{l_1, \dots, l_N\}^\top, \quad l_n = -w_n \left[y_n \cdot \log \sigma(x_n) + (1 - y_n) \cdot \log(1 - \sigma(x_n)) \right],$$

N is the batch size. If reduction is not 'none' (default 'mean'), then

$$\ell(x, y) = \begin{cases} \operatorname{mean}(L), & \text{if reduction} = \text{'mean';} \\ \operatorname{sum}(L), & \text{if reduction} = \text{'sum'.} \end{cases}$$

This is used for measuring the error of a reconstruction in for example an auto-encoder. Note that the targets *t[i]* should be numbers between 0 and 1.

```
    loss_fn = nn.BCEWithLogitsLoss()
```

Scheduler:-

It allows dynamic learning rate reducing based on some validation measurements.

```
scheduler = optim.lr_scheduler.OneCycleLR(optimizer=optimizer, pct_start=0.1, div_factor=1e3,
                                              max_lr=1e-2, epochs=EPOCHS, steps_per_epoch=len(trainloader))
```

VALIDATE:-

In machine learning, model validation is referred to as the process where a trained model is evaluated with a testing data set. The testing data set is a separate portion of the same data set from which the training set is derived. The main purpose of using the testing data set is to test the generalization ability of a trained model Model validation is carried out after model training. Together with model training, model validation aims to find an optimal model with the best performance.Here our function Valid_fn does the same task on MOA dataset.

```
 valid_loss, valid_preds = valid_fn(model, loss_fn, validloader, DEVICE)
```

INFERENCE:-

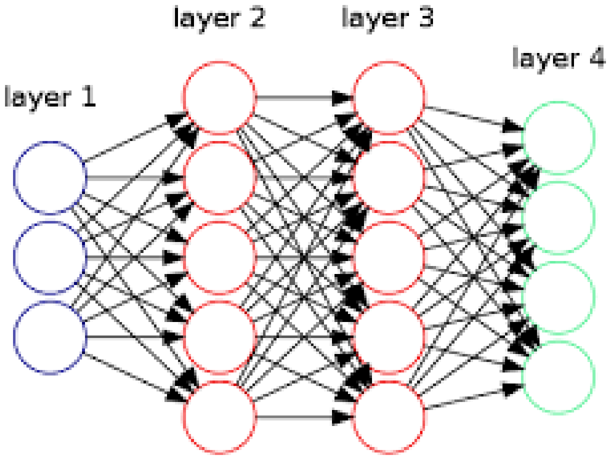
Inference refers to the process of using a trained machine learning algorithm to make a prediction. IoT data can be used as the input to a trained machine learning model, enabling predictions that can guide decision logic on the device, at the edge gateway or elsewhere in the IoT system. Here our function inference_fn predicts the output from the trained model on the train or new dynamic dataset.

```
  predictions = inference_fn(model, testloader, DEVICE)
```

MODEL (Neural Network):-

Multi-label classification involves predicting zero or more class labels. Unlike normal classification tasks where class labels are mutually exclusive, multi-label classification requires specialized machine learning algorithms that support predicting multiple mutually non-exclusive classes or "labels." Deep learning neural networks are an example of an algorithm that natively supports multi-label classification problems. Neural network models for multi-label classification tasks can be easily defined and evaluated using the pytorch deep learning library If the dataset is small, it is good practice to evaluate neural network models repeatedly on the same dataset and report the mean performance across the repeats. This is because of the stochastic nature of the learning algorithm. Additionally, it is good practice to use k-fold cross-validation instead of train/test splits of a dataset to get an unbiased estimate of model performance when making predictions on new data. Again, only if there is not too much data that the process can be completed in a reasonable time. Taking this into account, we will evaluate the MLP model on the multi-output regression task using repeated k-fold cross-validation with 7 folds and 7 repeats.

The MLP model will predict the probability for each class label by default. This means it will predict three probabilities for each sample. These can be converted to crisp class labels by rounding the values to either 0 or 1. We can then calculate the classification accuracy for the crisp class labels. The scores are collected and can be summarized by reporting the mean and standard deviation across all repeats and cross-validation folds. The number of output nodes should match the number of values you want to estimate. For binary classification, you need only one output node. For multiclass classification or multiple regression, you require multiple output nodes. In precise, binary classification using a feedforward neural net is done by computing the activation of a single output node, then checking whether it is larger than some threshold value (commonly 0 or .5). For multiclass classification with a k number of classes, you can calculate the values of k output nodes, then select the index 'i' of the largest value to predict class i..



FEED FORWARD NEURAL NETWORK FOR MULTILABEL CLASSIFICATION

Here in our NN model It does Batch Normalize at every step right from the input features:-1041 than it goes into the hidden layer of network which consist of 2048 neurons. There are 2 identical hidden layers of same size. When the features move from 1st hidden layer to 2nd one we have dropout with low weights below 0.4. In the Last layer we scale the neurons as per the required target i.e. 206

Batch Normalization `self.batch_norm1 = nn.BatchNorm1d(num_features)`
Applies Batch Normalization over a 2D or 3D input (a mini-batch of 1D inputs with optional additional channel dimension)

$$y = \frac{x - \mathrm{E}[x]}{\sqrt{\mathrm{Var}[x] + \epsilon}} * \gamma + \beta$$

The mean and standard-deviation are calculated per-dimension over the mini-batches and \gamma$\gamma$ and \beta$\beta$ are learnable parameter vectors of size C (where C is the input size). By default, the elements of \gamma$\gamma$ are set to 1 and the elements of \beta$\beta$ are set to 0. The standard-deviation is calculated via the biased estimator, equivalent to *torch.var(input, unbiased=False).*

Weight Normalization: `weight_norm(nn.Linear(num_features, hidden_size))`
A Simple Reparameterization to Accelerate Training of Deep Neural Networks. Tim Salimans, Diederik P. Kingma. Download PDF. We present **weight normalization**: a reparameterization of the **weight** vectors in a neural network that decouples the length of those **weight** vectors from their direction.

Linearization: `nn.Linear(num_features, hidden_size))`
It is a strategy used in **machine learning** to reduce the dimension with a linear projector. A low-dimensional linear projection of x can separate the values of f if this function remains constant in the direction of a high-dimensional linear space. **nn**. **Linear**(n,m) is a module that creates single layer feed forward network with n inputs and m output. Mathematically, this module is designed to calculate the **linear** equation Ax = b where x is input, b is output, A is weight.

Dense layer : `self.dense1 = nn.utils.weight_norm(nn.Linear(num_features, hidden_size))`
It is the regular deeply connected neural network **layer**. It is most common and frequently used **layer**. **Dense layer** does the below operation on the input and return the output. (self.dense)

Dropout:

```
self.dropout3 = nn.Dropout(0.4)
```

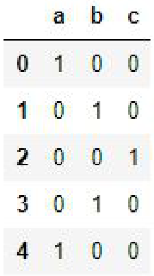
It is a regularization method that approximates training a large number of neural networks with different architectures in parallel. During training, some number of layer outputs are randomly ignored or "*dropped out*." This has the effect of making the layer look-like and be treated-like a layer with a different number of nodes and connectivity to the prior layer. In effect, each update to a layer during training is performed with a different "*view*" of the configured layer. Dropout has the effect of making the training process noisy, forcing nodes within a layer to probabilistically take on more or less responsibility for the inputs.This conceptualization suggests that perhaps dropout breaks-up situations where network layers co-adapt to correct mistakes from prior layers, in turn making the model more robust Dropout simulates a sparse activation from a given layer, which interestingly, in turn, encourages the network to actually learn a sparse representation as a side-effect. As such, it may be used as an alternative to activity regularization for encouraging sparse representations in autoencoder models.

PREPROCESSING DATA:-

_pandas.get_dummies() is used for data manipulation. It converts categorical data into dummy or indicator variables.

```
import pandas as pd
con = pd.Series(list('abcba'))
print(pd.get_dummies(con))
```

Output:

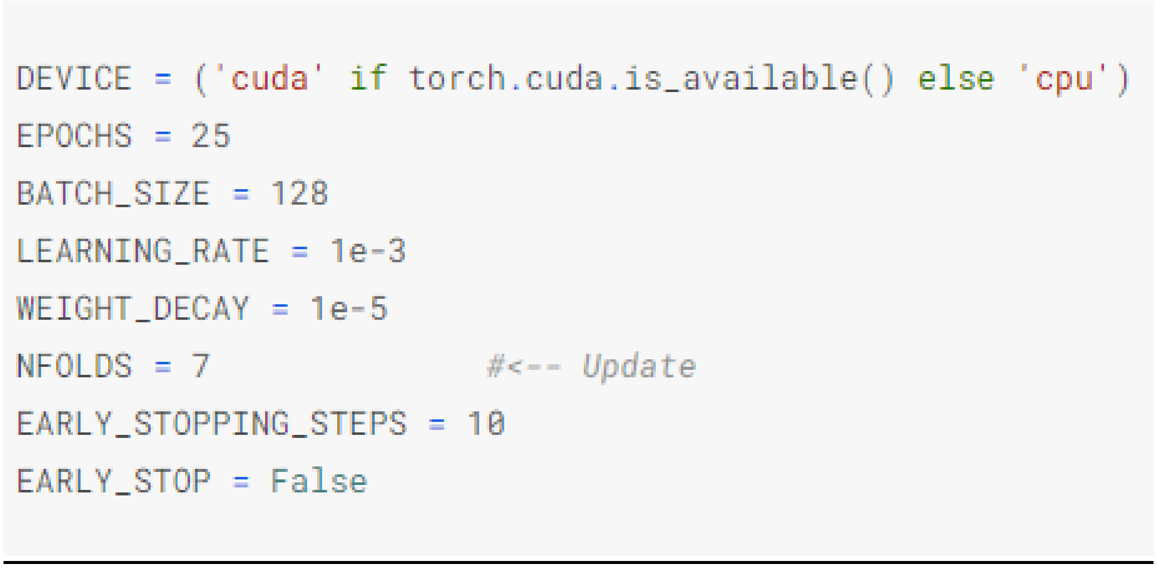


	a	b	c
0	1	0	0
1	0	1	0
2	0	0	1
3	0	1	0
4	1	0	0

```
   data = pd.get_dummies(data, columns=['cp_time','cp_dose'])
```

We did it for the 2 columns cp_time and cp_dose.

CHOOSING HYPERPARAMTERES BASED ON OUR MODEL:-



```
DEVICE = ('cuda' if torch.cuda.is_available() else 'cpu')
EPOCHS = 25
BATCH_SIZE = 128
LEARNING_RATE = 1e-3
WEIGHT_DECAY = 1e-5
NFOLDS = 7                #<-- Update
EARLY_STOPPING_STEPS = 10
EARLY_STOP = False
```

In machine learning, **hyperparameter optimization** or tuning is the problem of choosing a set of optimal hyperparameters for a learning algorithm. A hyperparameter is a parameter whose value is used to control the learning process. By contrast, the values of other parameters (typically node weights) are learned.

The same kind of machine learning model can require different constraints, weights or learning rates to generalize different data patterns. These measures are called hyperparameters, and have to be tuned so that the model can optimally solve the machine learning problem. Hyperparameter optimization finds a tuple of hyperparameters that yields an optimal model which minimizes a predefined loss function on given independent data.[1] The objective function takes a tuple of hyperparameters and returns the associated loss.[1] Cross-validation is often used to estimate this generalization performance.[2]

Epoch:-

It is a term used in machine learning and indicates the number of passes of the entire training dataset the machine learning algorithm has completed. Datasets are usually grouped into batches (especially when the amount of data is very large). Some people use the term iteration loosely and refer to putting one batch through the model as an iteration.

If the batch size is the whole training dataset then the number of epochs is the number of iterations. For practical reasons, this is usually not the case. Many models are created with more than one epoch. The general relation where dataset size is *d*, number of epochs is *e*, number of iterations is *i*, and batch size is *b* would be $d*e = i*b$. We choose epochs=128 because a model should run to train is based on many parameters related to both the data itself and the goal of the model, and while there have been efforts to turn this process into an algorithm and deep understanding of the data. Here we have decided the epoch and batch size according to the model and data visualization. We got good output and results after trial and error with many other model epochs and batch size

The learning rate :-

It is a hyperparameter that controls how much to change the model in response to the estimated error each time the model weights are updated. Choosing the learning rate is challenging as a value too small may result in a long training process that could get stuck, whereas a value too large may result in learning a sub-optimal set of weights too fast or an unstable training process. The learning rate may be the most important hyperparameter when configuring your neural network. Therefore it is vital to know how to investigate the effects of the learning rate on model performance and to build an intuition about the dynamics of the learning rate on model behavior. Here we have used a general learning rate for multilabel classification problem and it worked out for our dataset. By increasing the learning rate te system showed too much of loss and by decreasing the rate it became slower. So based on that we have generalized our learning rate. We also decided our learning rate on the factos such as :- How large learning rates result in unstable training and tiny rates result in a failure to train, Momentum can accelerate training and learning rate schedules can help to converge the optimization process, Adaptive learning rates can accelerate training and alleviate some of the pressure of choosing a learning rate and learning rate schedule.

Early Stopping:-

It is a form of regularization used to avoid overfitting when training a learner with an iterative method, such as gradient descent. Such methods update the learner so as to make it better fit the training data with each iteration. Up to a point, this improves the learner's performance on data outside of the training set. Past that point, however, improving the learner's fit to the training data comes at the expense of increased generalization error. Early stopping rules provide guidance as to how many iterations can be run before the learner begins to over-fit. Early stopping rules have been employed in many different machine learning methods, with varying amounts of theoretical foundation.

Weight Decay:-

Generally a wd = 0.00001 works pretty well. However, the folks at fastai have been a little conservative in this respect. Hence the default value of weight decay in fastai is actually 0.00001 . The reason to choose this value is because if you have too much weight decay, then no matter how much you train, the model never quite fits well enough whereas if you have too little weight decay, you can still train well, you just have to stop a little bit early.

N folds:-Here N folds is defined as the number of times the model will do training Iteration in order to average CV loss.

TRAINING FUNCTION:-

MULTIFOLD TRAINING FUNCTION = N * SINGLEFOLD TRAINING FUNCTION

SINGLE FOLD TRAINING FUNCTION :-

It follows the pipeline as given below. This is a generalized pytorch training function pipeline

- Pre-Processing raw train and test data.
- Converting it into tensor format.
- Defining Neural net model function, optimizer, scheduler, early stopping and loss.
- Iterating over epochs and training the model by passing the defined parameters by using Train and validation function.
- Saving the Validation and Train Loss in a list while training.
- Separating Minimum Losses in a list called as Best Losses.
- After iterations end it moves to evaluate the Train model on Test set by using Inference func.
- Once that is done the function return out of fold predictions and Test set predictions

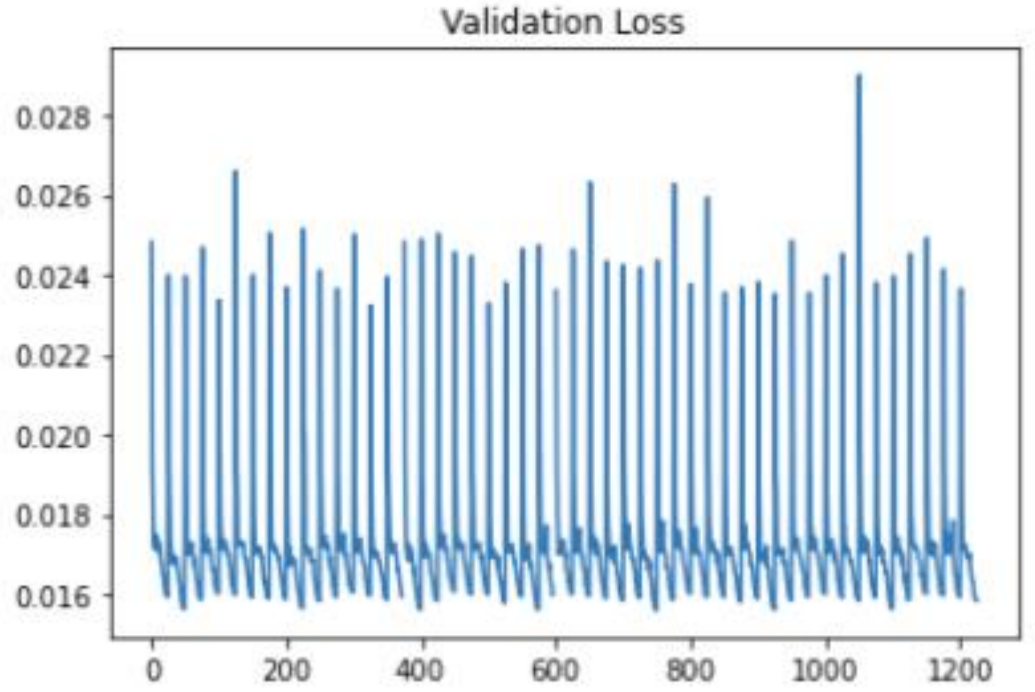
Here Number of Seed is used to repeat the steps given above in order to minimize the CV Log loss and to average it on multiple seeds.

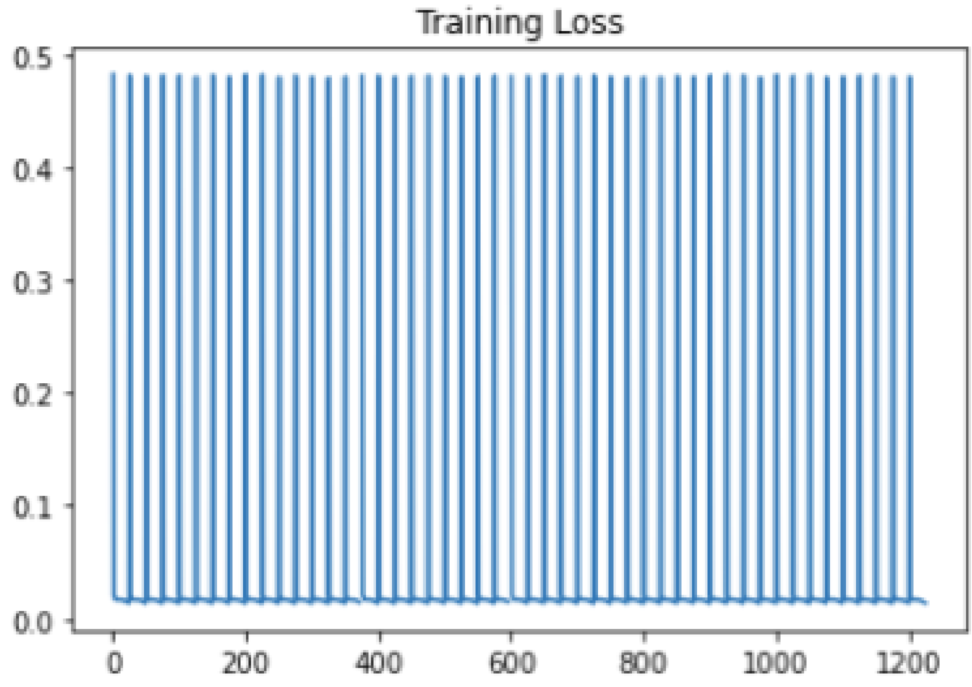
CALCULATING CV LOG LOSS:-

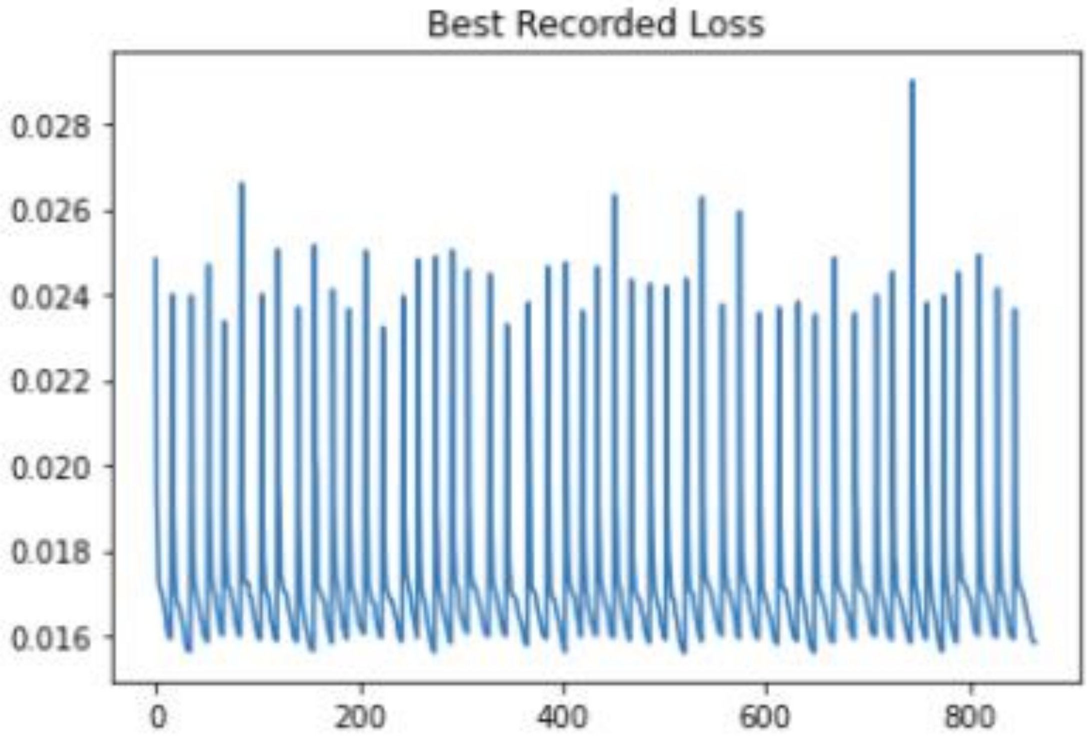
This is the loss function used in (multinomial) logistic regression and extensions of it such as neural networks, defined as the negative log-likelihood of a logistic model that returns y_pred probabilities for its training data y_true. The log loss is only defined for two or more labels. For a single sample with true label yt in {0,1} and estimated probability yp that yt = 1, the log loss is

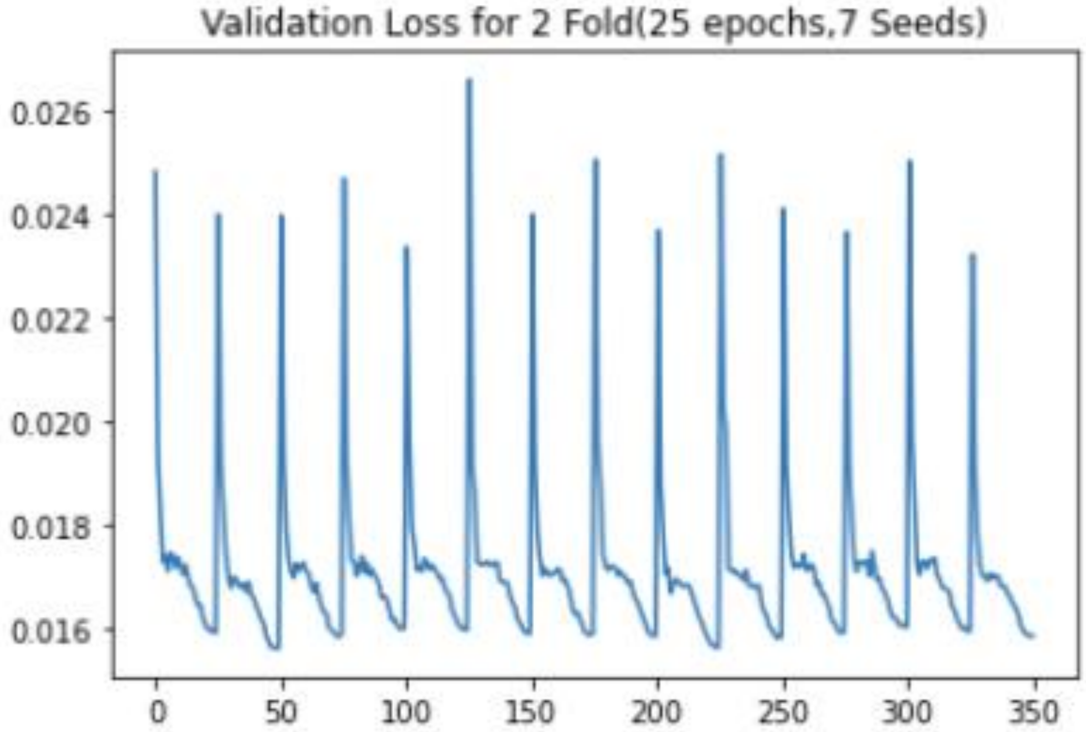
$-\log P(yt|yp) = -(yt \log(yp) + (1 - yt) \log(1 - yp))$

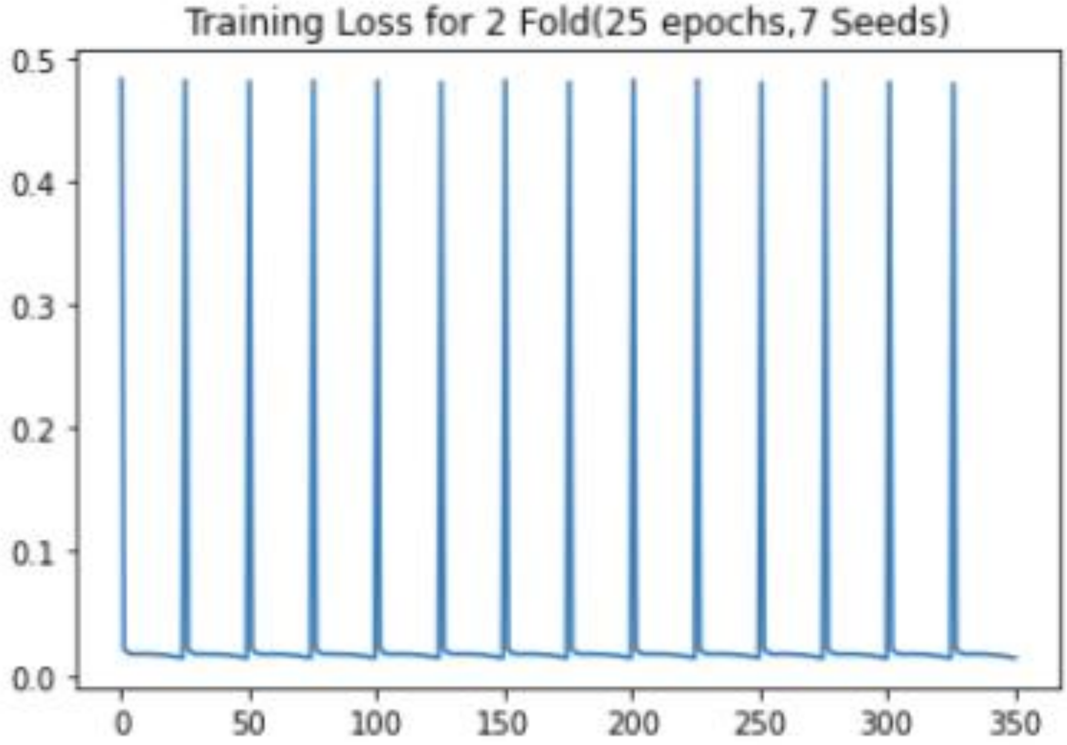
LEARNING AND VALIDATION CURVES:- (Loss vs (Seed*Epoch*Folds))

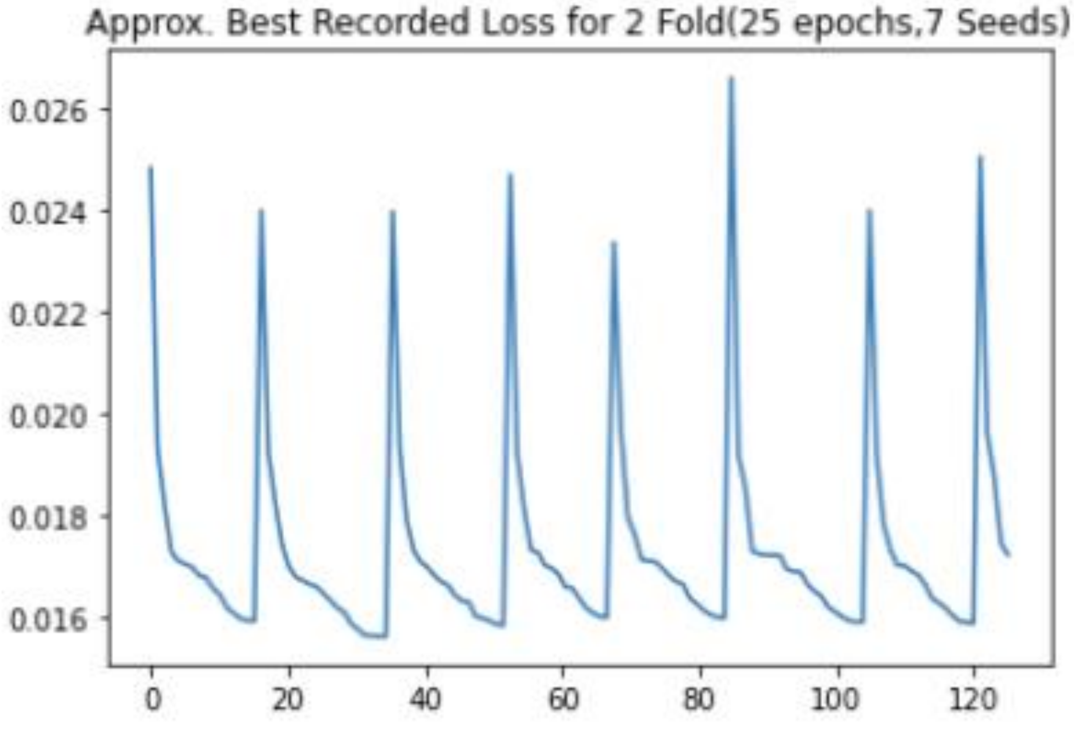





Best Recorded Loss
0.028
0.026
0.024
0.022
0.020
0.018
0.016
0
200
400
600
800


Validation Loss for 2 Fold(25 epochs,7 Seeds)
0.026
0.024
0.022
0.020
0.018
0.016
0
50
100
150
200
250
300
350


Training Loss for 2 Fold(25 epochs,7 Seeds)
0.5
0.4
0.3
0.2
0.1
0.0
0
50
100
150
200
250
300
350


Approx. Best Recorded Loss for 2 Fold(25 epochs,7 Seeds)
0.026
0.024
0.022
0.020
0.018
0.016
0
20
40
60
80
100
120

GENERATING PREDICTION FILE:-

Here we will merge the prediction and the test target and align that data with the proper heading into the submission.csv

SUBMITTING AND OBTAINING SCORE:-

Here For every `sig_id` you will be predicting the probability that the sample had a positive response for each `<MoA>` target. For NN `sig_id` rows and MM `<MoA>` targets, you will be making N×MN×M predictions. Submissions are scored by the log loss:

$$\text{score} = -\frac{1}{M}\sum_{m=1}^{M}\frac{1}{N}\sum_{i=1}^{N}\left[y_{i,m}\log(\hat{y}_{i,m}) + (1-y_{i,m})\log(1-\hat{y}_{i,m})\right]$$

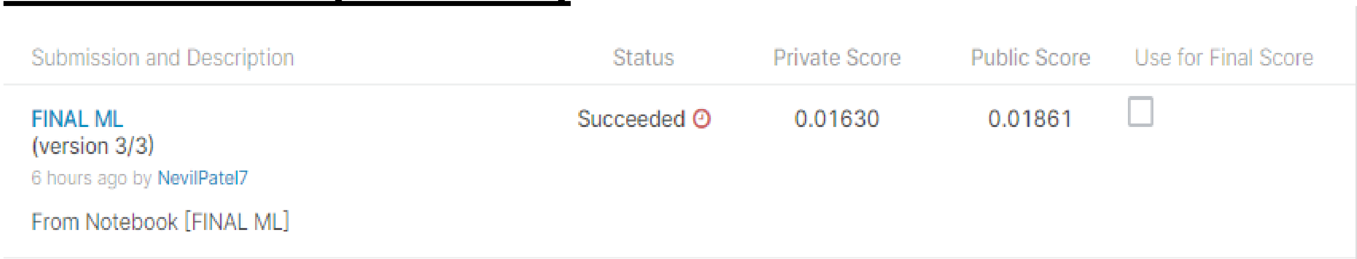
where:

- NN is the number of `sig_id` observations in the test data (i=1,…,Ni=1,…,N)
- MM is the number of scored MoA targets (m=1,…,Mm=1,…,M)
- y^i,my^i,m is the predicted probability of a positive `MoA` response for a `sig_id`
- yi,myi,m is the ground truth, 1 for a positive response, 0 otherwise
- log()log() is the natural (base e) logarithm

Note: the actual submitted predicted probabilities are replaced with max(min(p,1−10−15),10−15)max(min(p,1−10−15),10−15). A smaller log loss is better.

Number of Folds	CV Log Loss
1	0.1323
3	0.0737
4	0.0625
5	0.0239
7	0.01456
9	0.05627

FINAL SCORE(Folds = 7)



Submission and Description	Status	Private Score	Public Score	Use for Final Score
FINAL ML (version 3/3) 6 hours ago by NevilPatel7 From Notebook [FINAL ML]	Succeeded	0.01630	0.01861	□

REFERENCES:-

https://data36.com/pandas-tutorial-1-basics-reading-data-files-dataframes-data-selection/

https://pytorch.org/docs/stable/generated/torch.nn.BCEWithLogitsLoss.html

https://pytorch.org/docs/stable/optim.html

https://link.springer.com/referenceworkentry/10.1007%2F978-1-4419-9863-7_233#:~:text=Definition,the%20training%20set%20is%20derived.

https://datascience.stackexchange.com/questions/40275/multi-class-neural-networks-different-features

https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks/

https://www.geeksforgeeks.org/python-pandas-get_dummies-method/#:~:text=get_dummies()%20is%20used%20for,into%20dummy%20or%20indicator%20variables

https://www.kaggle.com/

CODE:-

MoA DATA VISUALIZATION:-

```
import numpy as np  # linear algebra
import pandas as pd  # data processing, CSV file I/O (e.g. pd.read_csv)
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import plotly.express as px
import random
import seaborn as sns
from colorama import Fore, Back, Style

y_ = Fore.YELLOW
r_ = Fore.RED
g_ = Fore.GREEN
b_ = Fore.BLUE
m_ = Fore.MAGENTA
sr_ = Style.RESET_ALL

def plotCounterPlot(feature):
    plt.figure(dpi=90)
    sns.countplot(train_data[feature])
    counts = train_data[feature].value_counts()
    for i in range(len(counts)):
        print(f"{b_}count of {counts.index[i]} is: {r_}{counts.values[i]}")


def plotDistribution(feature, color):
    plt.figure(figsize=(15,7))
    plt.subplot(121)
    sns.distplot(train_data[feature],color=color)
    plt.subplot(122)
    sns.violinplot(train_data[feature])
    print("{}Max value of {} is: {} {:.2f} \n{}Min value of {} is: {} {:.2f}\n{}Mean of {} is: {}{:.2f}\n{}Standard Deviation of {} is:{}{:.2f}"\

.format(y_,feature,r_,train_data[feature].max(),g_,feature,r_,train_data[feature].min(),b_,feature,r_,train_data[feature].mean(),m_,feature,r_,train_data[feature].std()))
# Input data files are available in the Kaggle/input/lish-moa

train_data = pd.read_csv('Kaggle/input/lish-moa/train_features.csv')
train_target = pd.read_csv('Kaggle/input/lish-moa/train_targets_scored.csv')

print('train_data shape is {}'.format(train_data.shape))
print("train_target shape is {}".format(train_target.shape))

#From Above , it seems number of data is 2E4, which is avergage number. The main things to note
#here is that: a) Number of features is 876, which is quite large. b) Number of labels for single data
```

```
#prediction is 207, which is also above average.


train_data, test_data , train_target, test_target  =
train_test_split(train_data,train_target, test_size=0.2)


print("sample_train data\n")
print(train_data.head())
print("")


NGTypeFeature = sum(train_data.columns.to_series().str.contains('g-') == True )
NCTypeFeature = sum(train_data.columns.to_series().str.contains('c-') == True )

print("GENE_TypeFeature = {} \nCELL_TypeFeature = {}".format(NGTypeFeature ,
NCTypeFeature))

plotCounterPlot('cp_type')
plt.title("CP_TYPE")
# plt.show()
print("")

plotCounterPlot('cp_time')
plt.title("CP_TIME")
# plt.show()
print("")

plotCounterPlot('cp_dose')
plt.title("CP_DOSE")
# plt.show()
print("")

print(train_data.describe())

print("Checking if train_data is in Panda or other format:
{}".format((type(train_data))))

print(train_data.head())

NDataRow , NDataCol = train_data.shape
print("Number Of Sample on which testing is done = {}  \nNumber of Feature which are
recorded = {}".format(NDataRow,NDataCol))

df = train_target.iloc[:,1:].sum(axis=0).sort_values(ascending=True)[-50:]
fig = px.bar(x=df.values,y = df.index,color=df.values)
fig.show()

plotDistribution("g-0" , "blue")
plt.title("G-0")

plotDistribution("g-12" , "blue")
plt.title("G-12")

plotDistribution("g-20" , "blue")
```

```
plt.title("G-20")

plotDistribution("g-40" , "blue")
plt.title("G-40")

plotDistribution("g-60" , "blue")
plt.title("G-60")

plotDistribution("c-30" , "blue")
plt.title("C-30")

plotDistribution("c-60" , "blue")
plt.title("C-60")

plotDistribution("c-99" , "blue")
plt.title("C-99")
plt.show()

minValOfFeature = min(train_data.min(axis = 1))
maxValOfFeature = max (train_data.iloc[: , 4:].max(axis = 1))
print("min Feature data = {} , maxvalOfFeature = {}".format(minValOfFeature ,
maxValOfFeature))
```

MoA Machine Learning Pipeline CODE:-

final-ml (1)

December 15, 2020

[74]:
```
# This Python 3 environment comes with many helpful analytics libraries␣
↪installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/
↪docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list␣
↪all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that␣
↪gets preserved as output when you create a version using "Save & Run All"
# You can also write temporary files to /kaggle/temp/, but they won't be saved␣
↪outside of the current session
```

```
/kaggle/input/iterativestratification/.travis.yml
/kaggle/input/iterativestratification/setup.cfg
/kaggle/input/iterativestratification/LICENSE
/kaggle/input/iterativestratification/.gitignore
/kaggle/input/iterativestratification/README.md
/kaggle/input/iterativestratification/setup.py
/kaggle/input/iterativestratification/tests/test_ml_stratifiers.py
/kaggle/input/iterativestratification/tests/__init__.py
/kaggle/input/iterativestratification/iterstrat/ml_stratifiers.py
/kaggle/input/iterativestratification/iterstrat/__init__.py
/kaggle/input/lish-moa/train_targets_scored.csv
/kaggle/input/lish-moa/sample_submission.csv
/kaggle/input/lish-moa/train_drug.csv
/kaggle/input/lish-moa/train_targets_nonscored.csv
```

```
/kaggle/input/lish-moa/train_features.csv
/kaggle/input/lish-moa/test_features.csv
```

[75]:

```
import sys
sys.path.append('../input/iterativestratification')
from iterstrat.ml_stratifiers import MultilabelStratifiedKFold

import numpy as np
import random
import pandas as pd
import matplotlib.pyplot as plt
import os
import copy
import seaborn as sns

from sklearn import preprocessing
from sklearn.metrics import log_loss
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA

import torch
import torch.nn as nn
import torch.nn.functional as F
import torch.optim as optim

import warnings
warnings.filterwarnings('ignore')

from sklearn.preprocessing import QuantileTransformer
```

1 Reading the Data Files

[76]:

```
os.listdir('../input/lish-moa')

train_features = pd.read_csv('../input/lish-moa/train_features.csv')
train_targets_scored = pd.read_csv('../input/lish-moa/train_targets_scored.csv')
train_targets_nonscored = pd.read_csv('../input/lish-moa/
 ↪train_targets_nonscored.csv')

test_features = pd.read_csv('../input/lish-moa/test_features.csv')
sample_submission = pd.read_csv('../input/lish-moa/sample_submission.csv')
```

2 *RankGauss on Train & Test Data Set*

[77]:

```
GENES = [col for col in train_features.columns if col.startswith('g-')]
CELLS = [col for col in train_features.columns if col.startswith('c-')]



for col in (GENES + CELLS):

    transformer = QuantileTransformer(n_quantiles=100,random_state=0,␣
 ↪output_distribution="normal")
    vec_len = len(train_features[col].values)
    vec_len_test = len(test_features[col].values)
    raw_vec = train_features[col].values.reshape(vec_len, 1)
    transformer.fit(raw_vec)

    train_features[col] = transformer.transform(raw_vec).reshape(1, vec_len)[0]
    test_features[col] = transformer.transform(test_features[col].values.
 ↪reshape(vec_len_test, 1)).reshape(1, vec_len_test)[0]
```

3 PCA + Normal Features

[78]:

```
def seed_everything(seed=42):
    random.seed(seed)
    os.environ['PYTHONHASHSEED'] = str(seed)
    np.random.seed(seed)
    torch.manual_seed(seed)
    torch.cuda.manual_seed(seed)
    torch.backends.cudnn.deterministic = True

seed_everything(seed=42)
```

[79]:

```
n_comp = 600  #<--Update

data = pd.concat([pd.DataFrame(train_features[GENES]), pd.
 ↪DataFrame(test_features[GENES])])
data2 = (PCA(n_components=n_comp, random_state=42).fit_transform(data[GENES]))
train2 = data2[:train_features.shape[0]]; test2 = data2[-test_features.shape[0]:
 ↪]

train2 = pd.DataFrame(train2, columns=[f'pca_G-{i}' for i in range(n_comp)])
test2 = pd.DataFrame(test2, columns=[f'pca_G-{i}' for i in range(n_comp)])

# drop_cols = [f'c-{i}' for i in range(n_comp,len(GENES))]
train_features = pd.concat((train_features, train2), axis=1)
```

```
test_features = pd.concat((test_features, test2), axis=1)
```

[80]:

```
n_comp = 50  #<--Update

data = pd.concat([pd.DataFrame(train_features[CELLS]), pd.
 ↪DataFrame(test_features[CELLS])])
data2 = (PCA(n_components=n_comp, random_state=42).fit_transform(data[CELLS]))
train2 = data2[:train_features.shape[0]]; test2 = data2[-test_features.shape[0]:
 ↪]

train2 = pd.DataFrame(train2, columns=[f'pca_C-{i}' for i in range(n_comp)])
test2 = pd.DataFrame(test2, columns=[f'pca_C-{i}' for i in range(n_comp)])

# drop_cols = [f'c-{i}' for i in range(n_comp,len(CELLS))]
train_features = pd.concat((train_features, train2), axis=1)
test_features = pd.concat((test_features, test2), axis=1)
```

[81]:

```
train_features.shape
```

[81]:

```
(23814, 1526)
```

4 BEST FEATURE SELECTION(VARIANCE ENCODING)

[82]:

```
from sklearn.feature_selection import VarianceThreshold


var_thresh = VarianceThreshold(0.8)  #<-- Update
data = train_features.append(test_features)
data_transformed = var_thresh.fit_transform(data.iloc[:, 4:])

train_features_transformed = data_transformed[ : train_features.shape[0]]
test_features_transformed = data_transformed[-test_features.shape[0] : ]


train_features = pd.
 ↪DataFrame(train_features[['sig_id','cp_type','cp_time','cp_dose']].values.
 ↪reshape(-1, 4),\
                              columns=['sig_id','cp_type','cp_time','cp_dose'])

train_features = pd.concat([train_features, pd.
 ↪DataFrame(train_features_transformed)], axis=1)


test_features = pd.
 ↪DataFrame(test_features[['sig_id','cp_type','cp_time','cp_dose']].values.
 ↪reshape(-1, 4),\
                              columns=['sig_id','cp_type','cp_time','cp_dose'])

test_features = pd.concat([test_features, pd.
 ↪DataFrame(test_features_transformed)], axis=1)


train = train_features.merge(train_targets_scored, on='sig_id')
train = train[train['cp_type']!='ctl_vehicle'].reset_index(drop=True)
test = test_features[test_features['cp_type']!='ctl_vehicle'].
 ↪reset_index(drop=True)

target = train[train_targets_scored.columns]

train = train.drop('cp_type', axis=1)
test = test.drop('cp_type', axis=1)
target_cols = target.drop('sig_id', axis=1).columns.values.tolist()

train_features.shape
```

[82]:

```
(23814, 1040)
```

5 CV FOLDS

[83]:

```
folds = train.copy()

mskf = MultilabelStratifiedKFold(n_splits=7)

for f, (t_idx, v_idx) in enumerate(mskf.split(X=train, y=target)):
    folds.loc[v_idx, 'kfold'] = int(f)

folds['kfold'] = folds['kfold'].astype(int)


print(train.shape)
print(folds.shape)
print(test.shape)
print(target.shape)
print(sample_submission.shape)
folds
```

```
(21948, 1245)
(21948, 1246)
(3624, 1039)
(21948, 207)
(3982, 207)
```

```
[83]:             sig_id cp_time cp_dose         0         1         2         3  \
      0      id_000644bb2      24      D1  1.134849  0.907687 -0.416385 -0.966814
      1      id_000779bfc      72      D1  0.119282  0.681738  0.272399  0.080113
      2      id_000a6266a      48      D1  0.779973  0.946463  1.425350 -0.132928
      3      id_0015fd391      48      D1 -0.734910 -0.274641 -0.438509  0.759097
      4      id_001626bd3      72      D2 -0.452718 -0.477513  0.972316  0.970731
      …               …     …       …       …         …         …        …
      21943  id_fff8c2444      72      D1  0.237856 -1.228203  0.218376 -0.365976
      21944  id_fffb1ceed      24      D2  0.209361 -0.022389 -0.235888 -0.796989
      21945  id_fffb70c0c      24      D2 -1.911021  0.587228 -0.588417  1.296405
      21946  id_fffcb9e7c      24      D1  0.816407  0.417618  0.431631  0.300617
      21947  id_ffffdd77b      72      D1 -1.243096  1.567730 -0.269573  1.083636

                    4         5         6  …  trpv_agonist  trpv_antagonist  \
      0     -0.254723 -1.017473 -1.364787  …             0                0
      1      1.205169  0.686517  0.313396  …             0                0
      2     -0.006122  1.492493  0.235577  …             0                0
      3      2.346330 -0.858153 -2.288417  …             0                0
      4      1.463427 -0.869555 -0.375501  …             0                0
      …           …         …         …  …           …              …
      21943 -0.330177  0.569243 -0.150978  …             0                0
      21944 -0.674009  0.919312  0.735603  …             0                0
      21945 -1.002640  0.850589 -0.304313  …             0                0
      21946  1.070346 -0.024189  0.048942  …             0                0
      21947 -0.511235 -2.099634 -1.622462  …             0                0

             tubulin_inhibitor  tyrosine_kinase_inhibitor  \
      0                      0                          0
      1                      0                          0
      2                      0                          0
      3                      0                          0
      4                      0                          0
      …                    …                          …
      21943                  0                          0
      21944                  0                          0
      21945                  0                          0
      21946                  0                          0
      21947                  0                          0

             ubiquitin_specific_protease_inhibitor  vegfr_inhibitor  vitamin_b  \
      0                                          0                0          0
      1                                          0                0          0
      2                                          0                0          0
      3                                          0                0          0
      4                                          0                0          0
      …                                        …                …        …
      21943                                      0                0          0
```

```
21944                                      0               0          0
21945                                      0               0          0
21946                                      0               0          0
21947                                      0               0          0

       vitamin_d_receptor_agonist  wnt_inhibitor  kfold
0                               0              0      5
1                               0              0      0
2                               0              0      6
3                               0              0      0
4                               0              0      4
…                           …            …    …
21943                           0              0      5
21944                           0              0      1
21945                           0              0      5
21946                           0              0      1
21947                           0              0      6

[21948 rows x 1246 columns]
```

6 DEFINING CLASSES(DATASET,TRAIN,VALID,INFERENCE)

[84]:
```python
class MoADataset:
    def __init__(self, features, targets):
        self.features = features
        self.targets = targets

    def __len__(self):
        return (self.features.shape[0])

    def __getitem__(self, idx):
        dct = {
            'x' : torch.tensor(self.features[idx, :], dtype=torch.float),
            'y' : torch.tensor(self.targets[idx, :], dtype=torch.float)
        }
        return dct

class TestDataset:
    def __init__(self, features):
        self.features = features

    def __len__(self):
        return (self.features.shape[0])

    def __getitem__(self, idx):
        dct = {
            'x' : torch.tensor(self.features[idx, :], dtype=torch.float)
        }
        return dct

def train_fn(model, optimizer, scheduler, loss_fn, dataloader, device):
    model.train()
    final_loss = 0

    for data in dataloader:
        optimizer.zero_grad()
        inputs, targets = data['x'].to(device), data['y'].to(device)
#         print(inputs.shape)
        outputs = model(inputs)
        loss = loss_fn(outputs, targets)
        loss.backward()
        optimizer.step()
        scheduler.step()

        final_loss += loss.item()

    final_loss /= len(dataloader)

    return final_loss

def valid_fn(model, loss_fn, dataloader, device):
    model.eval()
    final_loss = 0
    valid_preds = []

    for data in dataloader:
        inputs, targets = data['x'].to(device), data['y'].to(device)
        outputs = model(inputs)
        loss = loss_fn(outputs, targets)

        final_loss += loss.item()
        valid_preds.append(outputs.sigmoid().detach().cpu().numpy())

    final_loss /= len(dataloader)
    valid_preds = np.concatenate(valid_preds)

    return final_loss, valid_preds

def inference_fn(model, dataloader, device):
    model.eval()
    preds = []

    for data in dataloader:
        inputs = data['x'].to(device)

        with torch.no_grad():
            outputs = model(inputs)

        preds.append(outputs.sigmoid().detach().cpu().numpy())

    preds = np.concatenate(preds)

    return preds
```

7 MODEL

[85]:

```
class Model(nn.Module):      # <-- Update
    def __init__(self, num_features, num_targets, hidden_size):
        super(Model, self).__init__()
        self.batch_norm1 = nn.BatchNorm1d(num_features)
        self.dense1 = nn.utils.weight_norm(nn.Linear(num_features, hidden_size))

        self.batch_norm2 = nn.BatchNorm1d(hidden_size)
        self.dropout2 = nn.Dropout(0.4)
        self.dense2 = nn.utils.weight_norm(nn.Linear(hidden_size, hidden_size))

        self.batch_norm3 = nn.BatchNorm1d(hidden_size)
        self.dropout3 = nn.Dropout(0.4)
        self.dense3 = nn.utils.weight_norm(nn.Linear(hidden_size, num_targets))

    def forward(self, x):
        x = self.batch_norm1(x)
        x = F.relu(self.dense1(x))

        x = self.batch_norm2(x)
        x = self.dropout2(x)
        x = F.relu(self.dense2(x))

        x = self.batch_norm3(x)
        x = self.dropout3(x)
        x = self.dense3(x)

        return x
```

8 PREPROCESSING DATA

[86]:
```
def process_data(data):
    data = pd.get_dummies(data, columns=['cp_time','cp_dose'])
    return data

feature_cols = [c for c in process_data(folds).columns if c not in target_cols]
feature_cols = [c for c in feature_cols if c not in ['kfold','sig_id']]
len(feature_cols)
```

[86]:
```
1041
```

9 HYPERPARAMETERS

[87]:
```
DEVICE = ('cuda' if torch.cuda.is_available() else 'cpu')
EPOCHS = 25
BATCH_SIZE = 128
LEARNING_RATE = 1e-3
WEIGHT_DECAY = 1e-5
NFOLDS = 7              #<-- Update
EARLY_STOPPING_STEPS = 10
EARLY_STOP = False

num_features=len(feature_cols)
num_targets=len(target_cols)
hidden_size=2048
```

[88]:
```
# to plot validation loss
train_loss_ = []
# to plot the training loss
valid_loss_ = []
# to plot the best recorded loss
best_loss_ = []
```

10 DEFINING SINGLE AND MULTIFOLD TRAINING FUNCTION

[89]:
```
def run_training(fold, seed):
    global train_loss_
    global valid_loss_
    global best_loss_

    seed_everything(seed)

    train = process_data(folds)
    test_ = process_data(test)
    trn_idx = train[train['kfold'] != fold].index
    val_idx = train[train['kfold'] == fold].index

    train_df = train[train['kfold'] != fold].reset_index(drop=True)
    valid_df = train[train['kfold'] == fold].reset_index(drop=True)

    x_train, y_train  = train_df[feature_cols].values, train_df[target_cols].
↪values
    x_valid, y_valid =  valid_df[feature_cols].values, valid_df[target_cols].
↪values

    train_dataset = MoADataset(x_train, y_train)
    valid_dataset = MoADataset(x_valid, y_valid)
    trainloader = torch.utils.data.DataLoader(train_dataset,␣
↪batch_size=BATCH_SIZE, shuffle=True)
    validloader = torch.utils.data.DataLoader(valid_dataset,␣
↪batch_size=BATCH_SIZE, shuffle=False)

    model = Model(
        num_features=num_features,
        num_targets=num_targets,
        hidden_size=hidden_size,
    )

    model.to(DEVICE)
    optimizer = torch.optim.Adam(model.parameters(), lr=LEARNING_RATE,␣
↪weight_decay=WEIGHT_DECAY)
    scheduler = optim.lr_scheduler.OneCycleLR(optimizer=optimizer, pct_start=0.
↪1, div_factor=1e3,
                                              max_lr=1e-2, epochs=EPOCHS,␣
↪steps_per_epoch=len(trainloader))

    loss_fn = nn.BCEWithLogitsLoss()

    early_stopping_steps = EARLY_STOPPING_STEPS
    early_step = 0

    oof = np.zeros((len(train), target.iloc[:, 1:].shape[1]))
    best_loss = np.inf

    for epoch in range(EPOCHS):

        train_loss = train_fn(model, optimizer,scheduler, loss_fn, trainloader,␣
↪DEVICE)
        train_loss_.append(train_loss)
        print(f"FOLD: {fold}, EPOCH: {epoch}, train_loss: {train_loss}")
        valid_loss, valid_preds = valid_fn(model, loss_fn, validloader, DEVICE)
        valid_loss_.append(valid_loss)
        print(f"FOLD: {fold}, EPOCH: {epoch}, valid_loss: {valid_loss}")

        if valid_loss < best_loss:

            best_loss = valid_loss
            best_loss_.append(best_loss)
            oof[val_idx] = valid_preds
            torch.save(model.state_dict(), f"FOLD{fold}_.pth")

        elif(EARLY_STOP == True):

            early_step += 1
            if (early_step >= early_stopping_steps):
                break


    #--------------------- PREDICTION---------------------
    x_test = test_[feature_cols].values
    testdataset = TestDataset(x_test)
    testloader = torch.utils.data.DataLoader(testdataset, batch_size=BATCH_SIZE, shuffle=False)

    model = Model(
        num_features=num_features,
        num_targets=num_targets,
        hidden_size=hidden_size,

    )

    model.load_state_dict(torch.load(f"FOLD{fold}_.pth"))
    model.to(DEVICE)

    predictions = np.zeros((len(test_), target.iloc[:, 1:].shape[1]))
    predictions = inference_fn(model, testloader, DEVICE)

    return oof, predictions


def run_k_fold(NFOLDS, seed):
    oof = np.zeros((len(train), len(target_cols)))
    predictions = np.zeros((len(test), len(target_cols)))

    for fold in range(NFOLDS):
        oof_, pred_ = run_training(fold, seed)
        predictions += pred_ / NFOLDS
        oof += oof_

    return oof, predictions
```

11 TRAINING AND TESTING

[90]:

```
# Averaging on multiple SEEDS

SEED = [0, 1, 2, 3 ,4, 5 ,6]  #<-- Update
oof = np.zeros((len(train), len(target_cols)))
predictions = np.zeros((len(test), len(target_cols)))

for seed in SEED:

    oof_, predictions_ = run_k_fold(NFOLDS, seed)
    oof += oof_ / len(SEED)
    predictions += predictions_ / len(SEED)

train[target_cols] = oof
test[target_cols] = predictions
```

```
FOLD: 0, EPOCH: 0, train_loss: 0.48232868185811706
FOLD: 0, EPOCH: 0, valid_loss: 0.024814292937517166
FOLD: 0, EPOCH: 1, train_loss: 0.02107173926672157
FOLD: 0, EPOCH: 1, valid_loss: 0.019315823763608932
FOLD: 0, EPOCH: 2, train_loss: 0.018996933551163088
FOLD: 0, EPOCH: 2, valid_loss: 0.018227280303835868
FOLD: 0, EPOCH: 3, train_loss: 0.01764812731013006
FOLD: 0, EPOCH: 3, valid_loss: 0.017300846874713897
FOLD: 0, EPOCH: 4, train_loss: 0.017051006686322544
FOLD: 0, EPOCH: 4, valid_loss: 0.017407792396843435
FOLD: 0, EPOCH: 5, train_loss: 0.01694557220250571
FOLD: 0, EPOCH: 5, valid_loss: 0.017122614867985247
FOLD: 0, EPOCH: 6, train_loss: 0.016944134151753113
FOLD: 0, EPOCH: 6, valid_loss: 0.017487628497183324
FOLD: 0, EPOCH: 7, train_loss: 0.01701711434997669
FOLD: 0, EPOCH: 7, valid_loss: 0.017396538332104684
FOLD: 0, EPOCH: 8, train_loss: 0.01705438644327477
FOLD: 0, EPOCH: 8, valid_loss: 0.01718995697796345
FOLD: 0, EPOCH: 9, train_loss: 0.017072569477517587
FOLD: 0, EPOCH: 9, valid_loss: 0.017394925542175768
FOLD: 0, EPOCH: 10, train_loss: 0.01707026385860581
FOLD: 0, EPOCH: 10, valid_loss: 0.017227504588663577
```

```
FOLD: 0, EPOCH: 11, train_loss: 0.017071871802869704
FOLD: 0, EPOCH: 11, valid_loss: 0.01705886472016573
FOLD: 0, EPOCH: 12, train_loss: 0.01694477510442134
FOLD: 0, EPOCH: 12, valid_loss: 0.017242863439023495
FOLD: 0, EPOCH: 13, train_loss: 0.016834985806929822
FOLD: 0, EPOCH: 13, valid_loss: 0.017002611085772515
FOLD: 0, EPOCH: 14, train_loss: 0.016744794343046997
FOLD: 0, EPOCH: 14, valid_loss: 0.01683291345834732
FOLD: 0, EPOCH: 15, train_loss: 0.01654451344871805
FOLD: 0, EPOCH: 15, valid_loss: 0.016795877888798712
FOLD: 0, EPOCH: 16, train_loss: 0.016415288507127437
FOLD: 0, EPOCH: 16, valid_loss: 0.016584612652659415
FOLD: 0, EPOCH: 17, train_loss: 0.016158162020653687
FOLD: 0, EPOCH: 17, valid_loss: 0.016459527127444744
FOLD: 0, EPOCH: 18, train_loss: 0.015874763348830395
FOLD: 0, EPOCH: 18, valid_loss: 0.01646276954561472
FOLD: 0, EPOCH: 19, train_loss: 0.015568107081463143
FOLD: 0, EPOCH: 19, valid_loss: 0.016209116987884045
FOLD: 0, EPOCH: 20, train_loss: 0.01523009328437703
FOLD: 0, EPOCH: 20, valid_loss: 0.016090518794953823
FOLD: 0, EPOCH: 21, train_loss: 0.01476657761222854
FOLD: 0, EPOCH: 21, valid_loss: 0.015992887131869794
FOLD: 0, EPOCH: 22, train_loss: 0.01439099450956802
FOLD: 0, EPOCH: 22, valid_loss: 0.015996361672878264
FOLD: 0, EPOCH: 23, train_loss: 0.014070984020474412
FOLD: 0, EPOCH: 23, valid_loss: 0.015950537137687206
FOLD: 0, EPOCH: 24, train_loss: 0.01391883668344037
FOLD: 0, EPOCH: 24, valid_loss: 0.015938356928527356
FOLD: 1, EPOCH: 0, train_loss: 0.48164497100475695
FOLD: 1, EPOCH: 0, valid_loss: 0.023990048542618752
FOLD: 1, EPOCH: 1, train_loss: 0.02125996716168462
FOLD: 1, EPOCH: 1, valid_loss: 0.01925745338201523
FOLD: 1, EPOCH: 2, train_loss: 0.018982065341048907
FOLD: 1, EPOCH: 2, valid_loss: 0.01826371282339096
FOLD: 1, EPOCH: 3, train_loss: 0.017765485656251308
FOLD: 1, EPOCH: 3, valid_loss: 0.017435449585318567
FOLD: 1, EPOCH: 4, train_loss: 0.017055268855575397
FOLD: 1, EPOCH: 4, valid_loss: 0.01700600393116474
FOLD: 1, EPOCH: 5, train_loss: 0.016884893774479427
FOLD: 1, EPOCH: 5, valid_loss: 0.01680383525788784
FOLD: 1, EPOCH: 6, train_loss: 0.01696788412550477
FOLD: 1, EPOCH: 6, valid_loss: 0.01700858797878027
FOLD: 1, EPOCH: 7, train_loss: 0.01697442078722172
FOLD: 1, EPOCH: 7, valid_loss: 0.01698469549417496
FOLD: 1, EPOCH: 8, train_loss: 0.017057943442950443
FOLD: 1, EPOCH: 8, valid_loss: 0.01684837684035301
FOLD: 1, EPOCH: 9, train_loss: 0.01707357317082533
FOLD: 1, EPOCH: 9, valid_loss: 0.01681532885879278
```

```
FOLD: 1, EPOCH: 10, train_loss: 0.01701134249415933
FOLD: 1, EPOCH: 10, valid_loss: 0.016884175315499305
FOLD: 1, EPOCH: 11, train_loss: 0.01702892197435405
FOLD: 1, EPOCH: 11, valid_loss: 0.016736076734960078
FOLD: 1, EPOCH: 12, train_loss: 0.016938764728656432
FOLD: 1, EPOCH: 12, valid_loss: 0.016910760439932348
FOLD: 1, EPOCH: 13, train_loss: 0.01684195919557899
FOLD: 1, EPOCH: 13, valid_loss: 0.01666232731193304
FOLD: 1, EPOCH: 14, train_loss: 0.016781124767853695
FOLD: 1, EPOCH: 14, valid_loss: 0.016613694317638875
FOLD: 1, EPOCH: 15, train_loss: 0.01656384022944436
FOLD: 1, EPOCH: 15, valid_loss: 0.016481956504285335
FOLD: 1, EPOCH: 16, train_loss: 0.01639245954171127
FOLD: 1, EPOCH: 16, valid_loss: 0.016348120048642157
FOLD: 1, EPOCH: 17, train_loss: 0.01613922528669137
FOLD: 1, EPOCH: 17, valid_loss: 0.016200663335621358
FOLD: 1, EPOCH: 18, train_loss: 0.015877455656676472
FOLD: 1, EPOCH: 18, valid_loss: 0.016118471100926398
FOLD: 1, EPOCH: 19, train_loss: 0.015539707844050564
FOLD: 1, EPOCH: 19, valid_loss: 0.01588365714997053
FOLD: 1, EPOCH: 20, train_loss: 0.015128381315897516
FOLD: 1, EPOCH: 20, valid_loss: 0.015780030451714992
FOLD: 1, EPOCH: 21, train_loss: 0.014706411139190603
FOLD: 1, EPOCH: 21, valid_loss: 0.015661385916173456
FOLD: 1, EPOCH: 22, train_loss: 0.014258321748450905
FOLD: 1, EPOCH: 22, valid_loss: 0.015657323598861694
FOLD: 1, EPOCH: 23, train_loss: 0.013933191392697445
FOLD: 1, EPOCH: 23, valid_loss: 0.015643304996192455
FOLD: 1, EPOCH: 24, train_loss: 0.01374531992185278
FOLD: 1, EPOCH: 24, valid_loss: 0.015640568807721137
FOLD: 2, EPOCH: 0, train_loss: 0.4808451424093068
FOLD: 2, EPOCH: 0, valid_loss: 0.02396875739097595З
FOLD: 2, EPOCH: 1, train_loss: 0.021134131657732588
FOLD: 2, EPOCH: 1, valid_loss: 0.019332114979624747
FOLD: 2, EPOCH: 2, train_loss: 0.019023318502570497
FOLD: 2, EPOCH: 2, valid_loss: 0.01792460974305868
FOLD: 2, EPOCH: 3, train_loss: 0.017661144095314604
FOLD: 2, EPOCH: 3, valid_loss: 0.017341603264212607
FOLD: 2, EPOCH: 4, train_loss: 0.017108005756626323
FOLD: 2, EPOCH: 4, valid_loss: 0.017110489681363104
FOLD: 2, EPOCH: 5, train_loss: 0.016865184917082998
FOLD: 2, EPOCH: 5, valid_loss: 0.017000625021755696
FOLD: 2, EPOCH: 6, train_loss: 0.016894768467363045
FOLD: 2, EPOCH: 6, valid_loss: 0.017267115898430347
FOLD: 2, EPOCH: 7, train_loss: 0.017013532413663914
FOLD: 2, EPOCH: 7, valid_loss: 0.017131545171141623
FOLD: 2, EPOCH: 8, train_loss: 0.017032435018138416
FOLD: 2, EPOCH: 8, valid_loss: 0.017225769944489003
```

```
FOLD: 2, EPOCH: 9, train_loss: 0.017014865966222318
FOLD: 2, EPOCH: 9, valid_loss: 0.017286537401378154
FOLD: 2, EPOCH: 10, train_loss: 0.017077522225627282
FOLD: 2, EPOCH: 10, valid_loss: 0.017111664712429045
FOLD: 2, EPOCH: 11, train_loss: 0.01701057618673967
FOLD: 2, EPOCH: 11, valid_loss: 0.017076589316129685
FOLD: 2, EPOCH: 12, train_loss: 0.016921554780786947
FOLD: 2, EPOCH: 12, valid_loss: 0.01684307150542736
FOLD: 2, EPOCH: 13, train_loss: 0.0168493794416692
FOLD: 2, EPOCH: 13, valid_loss: 0.016715701147913934
FOLD: 2, EPOCH: 14, train_loss: 0.01669748318159864
FOLD: 2, EPOCH: 14, valid_loss: 0.016887223273515703
FOLD: 2, EPOCH: 15, train_loss: 0.01657572821999083
FOLD: 2, EPOCH: 15, valid_loss: 0.016643097177147865
FOLD: 2, EPOCH: 16, train_loss: 0.016343392761183434
FOLD: 2, EPOCH: 16, valid_loss: 0.016451832316815854
FOLD: 2, EPOCH: 17, train_loss: 0.016159619394765824
FOLD: 2, EPOCH: 17, valid_loss: 0.016334693729877472
FOLD: 2, EPOCH: 18, train_loss: 0.015878291211414094
FOLD: 2, EPOCH: 18, valid_loss: 0.01631740767508745
FOLD: 2, EPOCH: 19, train_loss: 0.015540706069797886
FOLD: 2, EPOCH: 19, valid_loss: 0.01603298556059599
FOLD: 2, EPOCH: 20, train_loss: 0.015120962412939185
FOLD: 2, EPOCH: 20, valid_loss: 0.015998328886926174
FOLD: 2, EPOCH: 21, train_loss: 0.014728535422865225
FOLD: 2, EPOCH: 21, valid_loss: 0.01594638582319021
FOLD: 2, EPOCH: 22, train_loss: 0.014307463738028289
FOLD: 2, EPOCH: 22, valid_loss: 0.015881385058164596
FOLD: 2, EPOCH: 23, train_loss: 0.01394795793659833
FOLD: 2, EPOCH: 23, valid_loss: 0.015857660584151743
FOLD: 2, EPOCH: 24, train_loss: 0.013801744549858327
FOLD: 2, EPOCH: 24, valid_loss: 0.015909500159323214
FOLD: 3, EPOCH: 0, train_loss: 0.481176372135983
FOLD: 3, EPOCH: 0, valid_loss: 0.02468633271753788
FOLD: 3, EPOCH: 1, train_loss: 0.021344500891610878
FOLD: 3, EPOCH: 1, valid_loss: 0.019207972064614297
FOLD: 3, EPOCH: 2, train_loss: 0.01890843418635884
FOLD: 3, EPOCH: 2, valid_loss: 0.01813852533698082
FOLD: 3, EPOCH: 3, train_loss: 0.017711779001314622
FOLD: 3, EPOCH: 3, valid_loss: 0.017333767376840115
FOLD: 3, EPOCH: 4, train_loss: 0.01706478454578085
FOLD: 3, EPOCH: 4, valid_loss: 0.017291972562670708
FOLD: 3, EPOCH: 5, train_loss: 0.01691005484131323
FOLD: 3, EPOCH: 5, valid_loss: 0.01703678574413061
FOLD: 3, EPOCH: 6, train_loss: 0.016993613102707732
FOLD: 3, EPOCH: 6, valid_loss: 0.017195160500705242
FOLD: 3, EPOCH: 7, train_loss: 0.016973940034707386
FOLD: 3, EPOCH: 7, valid_loss: 0.01741372115910053
```

```
FOLD: 3, EPOCH: 8, train_loss: 0.017041414481948833
FOLD: 3, EPOCH: 8, valid_loss: 0.017105118930339815
FOLD: 3, EPOCH: 9, train_loss: 0.017077530715234424
FOLD: 3, EPOCH: 9, valid_loss: 0.01728488862514496
FOLD: 3, EPOCH: 10, train_loss: 0.017103633475901724
FOLD: 3, EPOCH: 10, valid_loss: 0.01704591266810894
FOLD: 3, EPOCH: 11, train_loss: 0.01693209360803471
FOLD: 3, EPOCH: 11, valid_loss: 0.01716302376240492
FOLD: 3, EPOCH: 12, train_loss: 0.016991714471761062
FOLD: 3, EPOCH: 12, valid_loss: 0.017050273418426513
FOLD: 3, EPOCH: 13, train_loss: 0.016831964716872795
FOLD: 3, EPOCH: 13, valid_loss: 0.01698146477341652
FOLD: 3, EPOCH: 14, train_loss: 0.01675136979086464
FOLD: 3, EPOCH: 14, valid_loss: 0.01685590226203203
FOLD: 3, EPOCH: 15, train_loss: 0.016602338158658574
FOLD: 3, EPOCH: 15, valid_loss: 0.016609069257974625
FOLD: 3, EPOCH: 16, train_loss: 0.016363444643057123
FOLD: 3, EPOCH: 16, valid_loss: 0.01663355268537998
FOLD: 3, EPOCH: 17, train_loss: 0.0161432096050406
FOLD: 3, EPOCH: 17, valid_loss: 0.016587978713214397
FOLD: 3, EPOCH: 18, train_loss: 0.015875170343131028
FOLD: 3, EPOCH: 18, valid_loss: 0.016382760144770147
FOLD: 3, EPOCH: 19, train_loss: 0.015504951519118685
FOLD: 3, EPOCH: 19, valid_loss: 0.016193571649491787
FOLD: 3, EPOCH: 20, train_loss: 0.01509320682079411
FOLD: 3, EPOCH: 20, valid_loss: 0.01621102396398783
FOLD: 3, EPOCH: 21, train_loss: 0.014665798987356983
FOLD: 3, EPOCH: 21, valid_loss: 0.016099652200937273
FOLD: 3, EPOCH: 22, train_loss: 0.014204028810114682
FOLD: 3, EPOCH: 22, valid_loss: 0.01602823715656996
FOLD: 3, EPOCH: 23, train_loss: 0.013907317097495202
FOLD: 3, EPOCH: 23, valid_loss: 0.016014298871159554
FOLD: 3, EPOCH: 24, train_loss: 0.013694521850671899
FOLD: 3, EPOCH: 24, valid_loss: 0.016025093011558057
FOLD: 4, EPOCH: 0, train_loss: 0.4813230415286661
FOLD: 4, EPOCH: 0, valid_loss: 0.023361404612660407
FOLD: 4, EPOCH: 1, train_loss: 0.021278197528655025
FOLD: 4, EPOCH: 1, valid_loss: 0.01972547896206379
FOLD: 4, EPOCH: 2, train_loss: 0.018912846615322592
FOLD: 4, EPOCH: 2, valid_loss: 0.01798312544822693
FOLD: 4, EPOCH: 3, train_loss: 0.017604556549213776
FOLD: 4, EPOCH: 3, valid_loss: 0.017639932930469514
FOLD: 4, EPOCH: 4, train_loss: 0.017013342613924523
FOLD: 4, EPOCH: 4, valid_loss: 0.017146721184253692
FOLD: 4, EPOCH: 5, train_loss: 0.016852759002220063
FOLD: 4, EPOCH: 5, valid_loss: 0.017177169546484947
FOLD: 4, EPOCH: 6, train_loss: 0.0169189307065744
FOLD: 4, EPOCH: 6, valid_loss: 0.017123880200088024
```

```
FOLD: 4, EPOCH: 7, train_loss: 0.01695225622859739
FOLD: 4, EPOCH: 7, valid_loss: 0.01736565284430981
FOLD: 4, EPOCH: 8, train_loss: 0.01704889212158464
FOLD: 4, EPOCH: 8, valid_loss: 0.017267959751188756
FOLD: 4, EPOCH: 9, train_loss: 0.01705487996821298
FOLD: 4, EPOCH: 9, valid_loss: 0.01715393878519535
FOLD: 4, EPOCH: 10, train_loss: 0.017005288682016385
FOLD: 4, EPOCH: 10, valid_loss: 0.017248418480157852
FOLD: 4, EPOCH: 11, train_loss: 0.016980446925779588
FOLD: 4, EPOCH: 11, valid_loss: 0.017156862281262876
FOLD: 4, EPOCH: 12, train_loss: 0.01694923479642187
FOLD: 4, EPOCH: 12, valid_loss: 0.017103648111224175
FOLD: 4, EPOCH: 13, train_loss: 0.016831221128655535
FOLD: 4, EPOCH: 13, valid_loss: 0.016972102522850037
FOLD: 4, EPOCH: 14, train_loss: 0.016671811832355804
FOLD: 4, EPOCH: 14, valid_loss: 0.016997361853718758
FOLD: 4, EPOCH: 15, train_loss: 0.016549416310882487
FOLD: 4, EPOCH: 15, valid_loss: 0.016826039515435695
FOLD: 4, EPOCH: 16, train_loss: 0.016385519610983983
FOLD: 4, EPOCH: 16, valid_loss: 0.016713106743991374
FOLD: 4, EPOCH: 17, train_loss: 0.01613765553298856
FOLD: 4, EPOCH: 17, valid_loss: 0.01666247211396694
FOLD: 4, EPOCH: 18, train_loss: 0.015869157775273535
FOLD: 4, EPOCH: 18, valid_loss: 0.01638146284967661
FOLD: 4, EPOCH: 19, train_loss: 0.015529219504623186
FOLD: 4, EPOCH: 19, valid_loss: 0.016273619271814824
FOLD: 4, EPOCH: 20, train_loss: 0.015117206325640484
FOLD: 4, EPOCH: 20, valid_loss: 0.016149713955819607
FOLD: 4, EPOCH: 21, train_loss: 0.014739847729350029
FOLD: 4, EPOCH: 21, valid_loss: 0.01606719069182873
FOLD: 4, EPOCH: 22, train_loss: 0.01432079795215811
FOLD: 4, EPOCH: 22, valid_loss: 0.01601383153349161
FOLD: 4, EPOCH: 23, train_loss: 0.013987775803321884
FOLD: 4, EPOCH: 23, valid_loss: 0.015997431837022305
FOLD: 4, EPOCH: 24, train_loss: 0.013855091346820602
FOLD: 4, EPOCH: 24, valid_loss: 0.015997861847281455
FOLD: 5, EPOCH: 0, train_loss: 0.4801074617928794
FOLD: 5, EPOCH: 0, valid_loss: 0.02659213535487652
FOLD: 5, EPOCH: 1, train_loss: 0.02116788867987743
FOLD: 5, EPOCH: 1, valid_loss: 0.019204589501023293
FOLD: 5, EPOCH: 2, train_loss: 0.018973836833674484
FOLD: 5, EPOCH: 2, valid_loss: 0.018565854728221892
FOLD: 5, EPOCH: 3, train_loss: 0.01763743113371588
FOLD: 5, EPOCH: 3, valid_loss: 0.017317701652646064
FOLD: 5, EPOCH: 4, train_loss: 0.01701721434361067
FOLD: 5, EPOCH: 4, valid_loss: 0.017259297780692576
FOLD: 5, EPOCH: 5, train_loss: 0.016925979633720552
FOLD: 5, EPOCH: 5, valid_loss: 0.01723521910607815
```

```
FOLD: 5, EPOCH: 6, train_loss: 0.016948121080339766
FOLD: 5, EPOCH: 6, valid_loss: 0.017282073237001896
FOLD: 5, EPOCH: 7, train_loss: 0.016973019412839088
FOLD: 5, EPOCH: 7, valid_loss: 0.017305919714272023
FOLD: 5, EPOCH: 8, train_loss: 0.017002767091300212
FOLD: 5, EPOCH: 8, valid_loss: 0.017232264019548894
FOLD: 5, EPOCH: 9, train_loss: 0.017048870226000847
FOLD: 5, EPOCH: 9, valid_loss: 0.01723006770014763
FOLD: 5, EPOCH: 10, train_loss: 0.0170576323305263
FOLD: 5, EPOCH: 10, valid_loss: 0.017289252318441868
FOLD: 5, EPOCH: 11, train_loss: 0.016938964658904643
FOLD: 5, EPOCH: 11, valid_loss: 0.017286139875650405
FOLD: 5, EPOCH: 12, train_loss: 0.01697021892585722
FOLD: 5, EPOCH: 12, valid_loss: 0.016958149187266826
FOLD: 5, EPOCH: 13, train_loss: 0.016872969519372293
FOLD: 5, EPOCH: 13, valid_loss: 0.016912103816866874
FOLD: 5, EPOCH: 14, train_loss: 0.01667163887244909
FOLD: 5, EPOCH: 14, valid_loss: 0.016898549012839794
FOLD: 5, EPOCH: 15, train_loss: 0.016594011482282154
FOLD: 5, EPOCH: 15, valid_loss: 0.01690175097435713
FOLD: 5, EPOCH: 16, train_loss: 0.01637874620960278
FOLD: 5, EPOCH: 16, valid_loss: 0.016659300178289413
FOLD: 5, EPOCH: 17, train_loss: 0.016122987430517367
FOLD: 5, EPOCH: 17, valid_loss: 0.01652503363788128
FOLD: 5, EPOCH: 18, train_loss: 0.015850790288476718
FOLD: 5, EPOCH: 18, valid_loss: 0.0164246866106987
FOLD: 5, EPOCH: 19, train_loss: 0.015524352406512718
FOLD: 5, EPOCH: 19, valid_loss: 0.01621575381606817
FOLD: 5, EPOCH: 20, train_loss: 0.015152986484522723
FOLD: 5, EPOCH: 20, valid_loss: 0.01612159512937069
FOLD: 5, EPOCH: 21, train_loss: 0.014702875553598616
FOLD: 5, EPOCH: 21, valid_loss: 0.016026458367705346
FOLD: 5, EPOCH: 22, train_loss: 0.014284653259682006
FOLD: 5, EPOCH: 22, valid_loss: 0.015947826839983463
FOLD: 5, EPOCH: 23, train_loss: 0.01396009923421404
FOLD: 5, EPOCH: 23, valid_loss: 0.015928554013371468
FOLD: 5, EPOCH: 24, train_loss: 0.013782165950911791
FOLD: 5, EPOCH: 24, valid_loss: 0.015924964286386966
FOLD: 6, EPOCH: 0, train_loss: 0.48175223892694025
FOLD: 6, EPOCH: 0, valid_loss: 0.02399221919476986
FOLD: 6, EPOCH: 1, train_loss: 0.02126073577523637
FOLD: 6, EPOCH: 1, valid_loss: 0.019207300655543803
FOLD: 6, EPOCH: 2, train_loss: 0.018938960640558174
FOLD: 6, EPOCH: 2, valid_loss: 0.017848056741058826
FOLD: 6, EPOCH: 3, train_loss: 0.0177518091208878
FOLD: 6, EPOCH: 3, valid_loss: 0.017337964326143266
FOLD: 6, EPOCH: 4, train_loss: 0.017037777375860683
FOLD: 6, EPOCH: 4, valid_loss: 0.017036644108593464
```

```
FOLD: 6, EPOCH: 5, train_loss: 0.016927622043255236
FOLD: 6, EPOCH: 5, valid_loss: 0.017172663025557994
FOLD: 6, EPOCH: 6, train_loss: 0.016937246951623026
FOLD: 6, EPOCH: 6, valid_loss: 0.017102854177355766
FOLD: 6, EPOCH: 7, train_loss: 0.016956501355280682
FOLD: 6, EPOCH: 7, valid_loss: 0.017034680284559726
FOLD: 6, EPOCH: 8, train_loss: 0.01699756999455747
FOLD: 6, EPOCH: 8, valid_loss: 0.01706769809126854
FOLD: 6, EPOCH: 9, train_loss: 0.017067188118286683
FOLD: 6, EPOCH: 9, valid_loss: 0.01709349926561117
FOLD: 6, EPOCH: 10, train_loss: 0.01708656894106443
FOLD: 6, EPOCH: 10, valid_loss: 0.017170765399932862
FOLD: 6, EPOCH: 11, train_loss: 0.017027468680321765
FOLD: 6, EPOCH: 11, valid_loss: 0.0171413404494524
FOLD: 6, EPOCH: 12, train_loss: 0.016959985889190315
FOLD: 6, EPOCH: 12, valid_loss: 0.016926286593079567
FOLD: 6, EPOCH: 13, train_loss: 0.016871846496399973
FOLD: 6, EPOCH: 13, valid_loss: 0.016968220211565493
FOLD: 6, EPOCH: 14, train_loss: 0.016736460906662504
FOLD: 6, EPOCH: 14, valid_loss: 0.016844660341739655
FOLD: 6, EPOCH: 15, train_loss: 0.016604949758217042
FOLD: 6, EPOCH: 15, valid_loss: 0.016662905737757683
FOLD: 6, EPOCH: 16, train_loss: 0.01640641698170276
FOLD: 6, EPOCH: 16, valid_loss: 0.016386416777968406
FOLD: 6, EPOCH: 17, train_loss: 0.016158468343418875
FOLD: 6, EPOCH: 17, valid_loss: 0.01629565317183733
FOLD: 6, EPOCH: 18, train_loss: 0.01595688689195988
FOLD: 6, EPOCH: 18, valid_loss: 0.016329005546867847
FOLD: 6, EPOCH: 19, train_loss: 0.015578754214557256
FOLD: 6, EPOCH: 19, valid_loss: 0.01619295608252287
FOLD: 6, EPOCH: 20, train_loss: 0.015243950629366093
FOLD: 6, EPOCH: 20, valid_loss: 0.016043546721339227
FOLD: 6, EPOCH: 21, train_loss: 0.014772943835578808
FOLD: 6, EPOCH: 21, valid_loss: 0.015935490280389784
FOLD: 6, EPOCH: 22, train_loss: 0.014381752178377035
FOLD: 6, EPOCH: 22, valid_loss: 0.01591996818780899
FOLD: 6, EPOCH: 23, train_loss: 0.014021862133526478
FOLD: 6, EPOCH: 23, valid_loss: 0.015891979187726973
FOLD: 6, EPOCH: 24, train_loss: 0.01387392923387946
FOLD: 6, EPOCH: 24, valid_loss: 0.01591839961707592
FOLD: 0, EPOCH: 0, train_loss: 0.4803331242557488
FOLD: 0, EPOCH: 0, valid_loss: 0.025046857595443724
FOLD: 0, EPOCH: 1, train_loss: 0.02133326626503143
FOLD: 0, EPOCH: 1, valid_loss: 0.01961749292910099
FOLD: 0, EPOCH: 2, train_loss: 0.01894205640114489
FOLD: 0, EPOCH: 2, valid_loss: 0.018781046718358993
FOLD: 0, EPOCH: 3, train_loss: 0.017718616391525787
FOLD: 0, EPOCH: 3, valid_loss: 0.01746027521789074
```

```
FOLD: 0, EPOCH: 4, train_loss: 0.017017558546496087
FOLD: 0, EPOCH: 4, valid_loss: 0.017243310138583184
FOLD: 0, EPOCH: 5, train_loss: 0.01691255159676075
FOLD: 0, EPOCH: 5, valid_loss: 0.017142893485724925
FOLD: 0, EPOCH: 6, train_loss: 0.016904713262860874
FOLD: 0, EPOCH: 6, valid_loss: 0.017205427289009093
FOLD: 0, EPOCH: 7, train_loss: 0.017051309087396074
FOLD: 0, EPOCH: 7, valid_loss: 0.017223815768957138
FOLD: 0, EPOCH: 8, train_loss: 0.017043509355866585
FOLD: 0, EPOCH: 8, valid_loss: 0.01718041129410267
FOLD: 0, EPOCH: 9, train_loss: 0.01699416229159248
FOLD: 0, EPOCH: 9, valid_loss: 0.017043114863336085
FOLD: 0, EPOCH: 10, train_loss: 0.017022683120453033
FOLD: 0, EPOCH: 10, valid_loss: 0.0171242094039917
FOLD: 0, EPOCH: 11, train_loss: 0.016903570460371015
FOLD: 0, EPOCH: 11, valid_loss: 0.017022613845765592
FOLD: 0, EPOCH: 12, train_loss: 0.016885192906420654
FOLD: 0, EPOCH: 12, valid_loss: 0.017008679769933224
FOLD: 0, EPOCH: 13, train_loss: 0.016840744016020478
FOLD: 0, EPOCH: 13, valid_loss: 0.016891661994159222
FOLD: 0, EPOCH: 14, train_loss: 0.016712504819187582
FOLD: 0, EPOCH: 14, valid_loss: 0.0169240253046155
FOLD: 0, EPOCH: 15, train_loss: 0.016542553990369753
FOLD: 0, EPOCH: 15, valid_loss: 0.016720291152596475
FOLD: 0, EPOCH: 16, train_loss: 0.01637849164809905
FOLD: 0, EPOCH: 16, valid_loss: 0.016571029387414456
FOLD: 0, EPOCH: 17, train_loss: 0.01614289770067549
FOLD: 0, EPOCH: 17, valid_loss: 0.016507640704512595
FOLD: 0, EPOCH: 18, train_loss: 0.015900612884790312
FOLD: 0, EPOCH: 18, valid_loss: 0.016362338475883007
FOLD: 0, EPOCH: 19, train_loss: 0.015511510557919538
FOLD: 0, EPOCH: 19, valid_loss: 0.01621892809867859
FOLD: 0, EPOCH: 20, train_loss: 0.015142968475666582
FOLD: 0, EPOCH: 20, valid_loss: 0.016109126806259155
FOLD: 0, EPOCH: 21, train_loss: 0.014714944797257582
FOLD: 0, EPOCH: 21, valid_loss: 0.01595396425575018
FOLD: 0, EPOCH: 22, train_loss: 0.014313563907227548
FOLD: 0, EPOCH: 22, valid_loss: 0.015912441238760948
FOLD: 0, EPOCH: 23, train_loss: 0.013995921860138575
FOLD: 0, EPOCH: 23, valid_loss: 0.015884631648659706
FOLD: 0, EPOCH: 24, train_loss: 0.013780083925444253
FOLD: 0, EPOCH: 24, valid_loss: 0.015870806351304053
FOLD: 1, EPOCH: 0, train_loss: 0.48204327367094096
FOLD: 1, EPOCH: 0, valid_loss: 0.02368339627981186
FOLD: 1, EPOCH: 1, train_loss: 0.021045229201312778
FOLD: 1, EPOCH: 1, valid_loss: 0.01882676236331463
FOLD: 1, EPOCH: 2, train_loss: 0.01890543195716783
FOLD: 1, EPOCH: 2, valid_loss: 0.017788169234991075
```

```
FOLD: 1, EPOCH: 3, train_loss: 0.017802364566699178
FOLD: 1, EPOCH: 3, valid_loss: 0.01707376629114151
FOLD: 1, EPOCH: 4, train_loss: 0.017018821204499324
FOLD: 1, EPOCH: 4, valid_loss: 0.017183937691152097
FOLD: 1, EPOCH: 5, train_loss: 0.016922987180248816
FOLD: 1, EPOCH: 5, valid_loss: 0.01670518606901169
FOLD: 1, EPOCH: 6, train_loss: 0.01701606502186279
FOLD: 1, EPOCH: 6, valid_loss: 0.016856636814773084
FOLD: 1, EPOCH: 7, train_loss: 0.017036790167596065
FOLD: 1, EPOCH: 7, valid_loss: 0.016965620815753937
FOLD: 1, EPOCH: 8, train_loss: 0.01704240492133259
FOLD: 1, EPOCH: 8, valid_loss: 0.01694462526589632
FOLD: 1, EPOCH: 9, train_loss: 0.01713644253837616
FOLD: 1, EPOCH: 9, valid_loss: 0.016860792376101018
FOLD: 1, EPOCH: 10, train_loss: 0.017078801736134252
FOLD: 1, EPOCH: 10, valid_loss: 0.016825353875756264
FOLD: 1, EPOCH: 11, train_loss: 0.01704934801348821
FOLD: 1, EPOCH: 11, valid_loss: 0.016839721724390983
FOLD: 1, EPOCH: 12, train_loss: 0.01691074367789995
FOLD: 1, EPOCH: 12, valid_loss: 0.016858120150864124
FOLD: 1, EPOCH: 13, train_loss: 0.01687577176129534
FOLD: 1, EPOCH: 13, valid_loss: 0.016760229282081127
FOLD: 1, EPOCH: 14, train_loss: 0.01669906580909377
FOLD: 1, EPOCH: 14, valid_loss: 0.01660395871847868
FOLD: 1, EPOCH: 15, train_loss: 0.016584746463566412
FOLD: 1, EPOCH: 15, valid_loss: 0.01644287206232548
FOLD: 1, EPOCH: 16, train_loss: 0.016366250255480917
FOLD: 1, EPOCH: 16, valid_loss: 0.016288681253790856
FOLD: 1, EPOCH: 17, train_loss: 0.016180622581468553
FOLD: 1, EPOCH: 17, valid_loss: 0.01617266032844782
FOLD: 1, EPOCH: 18, train_loss: 0.01588758551927448
FOLD: 1, EPOCH: 18, valid_loss: 0.016058966629207135
FOLD: 1, EPOCH: 19, train_loss: 0.015506867788174526
FOLD: 1, EPOCH: 19, valid_loss: 0.015915762893855573
FOLD: 1, EPOCH: 20, train_loss: 0.015124451140968167
FOLD: 1, EPOCH: 20, valid_loss: 0.01582153473049402
FOLD: 1, EPOCH: 21, train_loss: 0.014753893537282133
FOLD: 1, EPOCH: 21, valid_loss: 0.01571882378309965
FOLD: 1, EPOCH: 22, train_loss: 0.014270615358488494
FOLD: 1, EPOCH: 22, valid_loss: 0.01569646816700697
FOLD: 1, EPOCH: 23, train_loss: 0.01396188179829291
FOLD: 1, EPOCH: 23, valid_loss: 0.01565823160111904
FOLD: 1, EPOCH: 24, train_loss: 0.013796968829064142
FOLD: 1, EPOCH: 24, valid_loss: 0.015650779753923417
FOLD: 2, EPOCH: 0, train_loss: 0.4820510152169839
FOLD: 2, EPOCH: 0, valid_loss: 0.025146274119615553
FOLD: 2, EPOCH: 1, train_loss: 0.020991972882021852
FOLD: 2, EPOCH: 1, valid_loss: 0.02022985689342022
```

```
FOLD: 2, EPOCH: 2, train_loss: 0.018908680815781866
FOLD: 2, EPOCH: 2, valid_loss: 0.019835590980947018
FOLD: 2, EPOCH: 3, train_loss: 0.017685525543785014
FOLD: 2, EPOCH: 3, valid_loss: 0.017194286547601224
FOLD: 2, EPOCH: 4, train_loss: 0.016952463254636646
FOLD: 2, EPOCH: 4, valid_loss: 0.01713101364672184
FOLD: 2, EPOCH: 5, train_loss: 0.01701431269092219
FOLD: 2, EPOCH: 5, valid_loss: 0.01715358067303896
FOLD: 2, EPOCH: 6, train_loss: 0.01689903807452544
FOLD: 2, EPOCH: 6, valid_loss: 0.01707582127302885
FOLD: 2, EPOCH: 7, train_loss: 0.01703832465775159
FOLD: 2, EPOCH: 7, valid_loss: 0.017079290710389616
FOLD: 2, EPOCH: 8, train_loss: 0.017020210978530702
FOLD: 2, EPOCH: 8, valid_loss: 0.017016660645604132
FOLD: 2, EPOCH: 9, train_loss: 0.01708731966942143
FOLD: 2, EPOCH: 9, valid_loss: 0.016953468322753906
FOLD: 2, EPOCH: 10, train_loss: 0.01706284455650923
FOLD: 2, EPOCH: 10, valid_loss: 0.017143521867692472
FOLD: 2, EPOCH: 11, train_loss: 0.01700388161832986
FOLD: 2, EPOCH: 11, valid_loss: 0.01687226928770542
FOLD: 2, EPOCH: 12, train_loss: 0.016934767656889903
FOLD: 2, EPOCH: 12, valid_loss: 0.016820230446755888
FOLD: 2, EPOCH: 13, train_loss: 0.016846961481179916
FOLD: 2, EPOCH: 13, valid_loss: 0.01684346217662096
FOLD: 2, EPOCH: 14, train_loss: 0.016770995874591424
FOLD: 2, EPOCH: 14, valid_loss: 0.01680222399532795
FOLD: 2, EPOCH: 15, train_loss: 0.01662287802821925
FOLD: 2, EPOCH: 15, valid_loss: 0.016849438324570656
FOLD: 2, EPOCH: 16, train_loss: 0.01643636502756351
FOLD: 2, EPOCH: 16, valid_loss: 0.01653080329298973
FOLD: 2, EPOCH: 17, train_loss: 0.01618100652078382
FOLD: 2, EPOCH: 17, valid_loss: 0.016350412815809248
FOLD: 2, EPOCH: 18, train_loss: 0.015904268972119506
FOLD: 2, EPOCH: 18, valid_loss: 0.016287979371845722
FOLD: 2, EPOCH: 19, train_loss: 0.015616927834443088
FOLD: 2, EPOCH: 19, valid_loss: 0.016090837828814982
FOLD: 2, EPOCH: 20, train_loss: 0.0152122767097285
FOLD: 2, EPOCH: 20, valid_loss: 0.015998590104281903
FOLD: 2, EPOCH: 21, train_loss: 0.01480706524560038
FOLD: 2, EPOCH: 21, valid_loss: 0.015953749231994153
FOLD: 2, EPOCH: 22, train_loss: 0.014412246847233804
FOLD: 2, EPOCH: 22, valid_loss: 0.015839036479592325
FOLD: 2, EPOCH: 23, train_loss: 0.01412516778499699
FOLD: 2, EPOCH: 23, valid_loss: 0.0158663235232234
FOLD: 2, EPOCH: 24, train_loss: 0.013946458146128118
FOLD: 2, EPOCH: 24, valid_loss: 0.01584696341305971
FOLD: 3, EPOCH: 0, train_loss: 0.47982344677557753
FOLD: 3, EPOCH: 0, valid_loss: 0.024105850383639337
```

```
FOLD: 3, EPOCH: 1, train_loss: 0.021277950633139836
FOLD: 3, EPOCH: 1, valid_loss: 0.019170698374509812
FOLD: 3, EPOCH: 2, train_loss: 0.019299962589529908
FOLD: 3, EPOCH: 2, valid_loss: 0.01814370892941952
FOLD: 3, EPOCH: 3, train_loss: 0.017696060994196506
FOLD: 3, EPOCH: 3, valid_loss: 0.01747499793767929
FOLD: 3, EPOCH: 4, train_loss: 0.017028058714353714
FOLD: 3, EPOCH: 4, valid_loss: 0.017192985378205777
FOLD: 3, EPOCH: 5, train_loss: 0.016821597348011676
FOLD: 3, EPOCH: 5, valid_loss: 0.01727366302162409
FOLD: 3, EPOCH: 6, train_loss: 0.016910698480245206
FOLD: 3, EPOCH: 6, valid_loss: 0.017198177985846996
FOLD: 3, EPOCH: 7, train_loss: 0.017019016193035915
FOLD: 3, EPOCH: 7, valid_loss: 0.017244680263102056
FOLD: 3, EPOCH: 8, train_loss: 0.016994160289565723
FOLD: 3, EPOCH: 8, valid_loss: 0.017449271269142627
FOLD: 3, EPOCH: 9, train_loss: 0.017052228125382443
FOLD: 3, EPOCH: 9, valid_loss: 0.017136994525790216
FOLD: 3, EPOCH: 10, train_loss: 0.01701360912423353
FOLD: 3, EPOCH: 10, valid_loss: 0.017155649289488793
FOLD: 3, EPOCH: 11, train_loss: 0.016940214196029976
FOLD: 3, EPOCH: 11, valid_loss: 0.017255095578730107
FOLD: 3, EPOCH: 12, train_loss: 0.016930743868202983
FOLD: 3, EPOCH: 12, valid_loss: 0.017167340964078903
FOLD: 3, EPOCH: 13, train_loss: 0.016862368394880475
FOLD: 3, EPOCH: 13, valid_loss: 0.01684935811907053
FOLD: 3, EPOCH: 14, train_loss: 0.016706393017959432
FOLD: 3, EPOCH: 14, valid_loss: 0.016909665390849114
FOLD: 3, EPOCH: 15, train_loss: 0.016541230023464783
FOLD: 3, EPOCH: 15, valid_loss: 0.016688522174954413
FOLD: 3, EPOCH: 16, train_loss: 0.016348672352832595
FOLD: 3, EPOCH: 16, valid_loss: 0.01668941643089056
FOLD: 3, EPOCH: 17, train_loss: 0.01611566392793542
FOLD: 3, EPOCH: 17, valid_loss: 0.01654868442565203
FOLD: 3, EPOCH: 18, train_loss: 0.01583788720067261
FOLD: 3, EPOCH: 18, valid_loss: 0.01644157472997904
FOLD: 3, EPOCH: 19, train_loss: 0.01548277423241917
FOLD: 3, EPOCH: 19, valid_loss: 0.01618418760597706
FOLD: 3, EPOCH: 20, train_loss: 0.015120573322839883
FOLD: 3, EPOCH: 20, valid_loss: 0.01614241797477007
FOLD: 3, EPOCH: 21, train_loss: 0.014647711030378634
FOLD: 3, EPOCH: 21, valid_loss: 0.01604886870831251
FOLD: 3, EPOCH: 22, train_loss: 0.014210126939274016
FOLD: 3, EPOCH: 22, valid_loss: 0.015950331464409828
FOLD: 3, EPOCH: 23, train_loss: 0.013818974733403344
FOLD: 3, EPOCH: 23, valid_loss: 0.01593074545264244
FOLD: 3, EPOCH: 24, train_loss: 0.013652523447360312
FOLD: 3, EPOCH: 24, valid_loss: 0.015954088270664215
```

```
FOLD: 4, EPOCH: 0, train_loss: 0.4809735416479054
FOLD: 4, EPOCH: 0, valid_loss: 0.023650598973035813
FOLD: 4, EPOCH: 1, train_loss: 0.02104735931977123
FOLD: 4, EPOCH: 1, valid_loss: 0.019190338030457495
FOLD: 4, EPOCH: 2, train_loss: 0.018841004702357615
FOLD: 4, EPOCH: 2, valid_loss: 0.018162243217229843
FOLD: 4, EPOCH: 3, train_loss: 0.01767152202550043
FOLD: 4, EPOCH: 3, valid_loss: 0.017353024035692215
FOLD: 4, EPOCH: 4, train_loss: 0.017011754309796557
FOLD: 4, EPOCH: 4, valid_loss: 0.01711425706744194
FOLD: 4, EPOCH: 5, train_loss: 0.016891671952848533
FOLD: 4, EPOCH: 5, valid_loss: 0.017297650203108788
FOLD: 4, EPOCH: 6, train_loss: 0.01693720040450088
FOLD: 4, EPOCH: 6, valid_loss: 0.01726415805518627
FOLD: 4, EPOCH: 7, train_loss: 0.016959801557011343
FOLD: 4, EPOCH: 7, valid_loss: 0.017308461591601373
FOLD: 4, EPOCH: 8, train_loss: 0.01701120998026157
FOLD: 4, EPOCH: 8, valid_loss: 0.01730369236320257
FOLD: 4, EPOCH: 9, train_loss: 0.01695489654076748
FOLD: 4, EPOCH: 9, valid_loss: 0.017030922695994377
FOLD: 4, EPOCH: 10, train_loss: 0.01698900807370134
FOLD: 4, EPOCH: 10, valid_loss: 0.017513072416186333
FOLD: 4, EPOCH: 11, train_loss: 0.01694696874288069
FOLD: 4, EPOCH: 11, valid_loss: 0.01724404811859131
FOLD: 4, EPOCH: 12, train_loss: 0.016868208592035333
FOLD: 4, EPOCH: 12, valid_loss: 0.0169587005674839
FOLD: 4, EPOCH: 13, train_loss: 0.016773396133616263
FOLD: 4, EPOCH: 13, valid_loss: 0.016989801116287707
FOLD: 4, EPOCH: 14, train_loss: 0.01669156621051889
FOLD: 4, EPOCH: 14, valid_loss: 0.016778020858764647
FOLD: 4, EPOCH: 15, train_loss: 0.016519598707201936
FOLD: 4, EPOCH: 15, valid_loss: 0.01673418764024973
FOLD: 4, EPOCH: 16, train_loss: 0.016378539747425487
FOLD: 4, EPOCH: 16, valid_loss: 0.016595640517771244
FOLD: 4, EPOCH: 17, train_loss: 0.016091123958225963
FOLD: 4, EPOCH: 17, valid_loss: 0.01643480721861124
FOLD: 4, EPOCH: 18, train_loss: 0.01579013694280467
FOLD: 4, EPOCH: 18, valid_loss: 0.016253395415842534
FOLD: 4, EPOCH: 19, train_loss: 0.015510907878174263
FOLD: 4, EPOCH: 19, valid_loss: 0.016244264096021654
FOLD: 4, EPOCH: 20, train_loss: 0.015096983327498648
FOLD: 4, EPOCH: 20, valid_loss: 0.016174677312374115
FOLD: 4, EPOCH: 21, train_loss: 0.01465426764267237
FOLD: 4, EPOCH: 21, valid_loss: 0.01608598105609417
FOLD: 4, EPOCH: 22, train_loss: 0.014201008923807922
FOLD: 4, EPOCH: 22, valid_loss: 0.016082317344844343
FOLD: 4, EPOCH: 23, train_loss: 0.013850487451873669
FOLD: 4, EPOCH: 23, valid_loss: 0.016049367859959602
```

```
FOLD: 4, EPOCH: 24, train_loss: 0.01369883362077126
FOLD: 4, EPOCH: 24, valid_loss: 0.0160575357824564
FOLD: 5, EPOCH: 0, train_loss: 0.48029334204537527
FOLD: 5, EPOCH: 0, valid_loss: 0.025014126598834993
FOLD: 5, EPOCH: 1, train_loss: 0.02117114745992787
FOLD: 5, EPOCH: 1, valid_loss: 0.01971410498023033
FOLD: 5, EPOCH: 2, train_loss: 0.018955072236000275
FOLD: 5, EPOCH: 2, valid_loss: 0.01852488540112972
FOLD: 5, EPOCH: 3, train_loss: 0.017742974632958167
FOLD: 5, EPOCH: 3, valid_loss: 0.017251971177756786
FOLD: 5, EPOCH: 4, train_loss: 0.017016697934746337
FOLD: 5, EPOCH: 4, valid_loss: 0.01705134455114603
FOLD: 5, EPOCH: 5, train_loss: 0.016837000365362686
FOLD: 5, EPOCH: 5, valid_loss: 0.01731704343110323
FOLD: 5, EPOCH: 6, train_loss: 0.01692285353862712
FOLD: 5, EPOCH: 6, valid_loss: 0.017304558493196964
FOLD: 5, EPOCH: 7, train_loss: 0.01704342267951187
FOLD: 5, EPOCH: 7, valid_loss: 0.01717522121965885
FOLD: 5, EPOCH: 8, train_loss: 0.01698063169510997
FOLD: 5, EPOCH: 8, valid_loss: 0.017330071814358236
FOLD: 5, EPOCH: 9, train_loss: 0.017046458043512843
FOLD: 5, EPOCH: 9, valid_loss: 0.017309673093259335
FOLD: 5, EPOCH: 10, train_loss: 0.016983210020476862
FOLD: 5, EPOCH: 10, valid_loss: 0.017380312271416187
FOLD: 5, EPOCH: 11, train_loss: 0.016989375933447258
FOLD: 5, EPOCH: 11, valid_loss: 0.017053677439689635
FOLD: 5, EPOCH: 12, train_loss: 0.0168447280809486
FOLD: 5, EPOCH: 12, valid_loss: 0.016961749345064163
FOLD: 5, EPOCH: 13, train_loss: 0.01682296094997805
FOLD: 5, EPOCH: 13, valid_loss: 0.016820741519331932
FOLD: 5, EPOCH: 14, train_loss: 0.01673129364391979
FOLD: 5, EPOCH: 14, valid_loss: 0.016782965026795865
FOLD: 5, EPOCH: 15, train_loss: 0.016579824430095095
FOLD: 5, EPOCH: 15, valid_loss: 0.016765388697385787
FOLD: 5, EPOCH: 16, train_loss: 0.016326379252686388
FOLD: 5, EPOCH: 16, valid_loss: 0.01671287514269352
FOLD: 5, EPOCH: 17, train_loss: 0.016158953613164474
FOLD: 5, EPOCH: 17, valid_loss: 0.016432282850146293
FOLD: 5, EPOCH: 18, train_loss: 0.015841717952165473
FOLD: 5, EPOCH: 18, valid_loss: 0.01636404249817133
FOLD: 5, EPOCH: 19, train_loss: 0.015482104091974749
FOLD: 5, EPOCH: 19, valid_loss: 0.01620681505650282
FOLD: 5, EPOCH: 20, train_loss: 0.015064914064595893
FOLD: 5, EPOCH: 20, valid_loss: 0.01610106658190489
FOLD: 5, EPOCH: 21, train_loss: 0.014674901892589468
FOLD: 5, EPOCH: 21, valid_loss: 0.016018796302378176
FOLD: 5, EPOCH: 22, train_loss: 0.01425523020732565
FOLD: 5, EPOCH: 22, valid_loss: 0.016033078879117965
```

```
FOLD: 5, EPOCH: 23, train_loss: 0.013923346166353242
FOLD: 5, EPOCH: 23, valid_loss: 0.015973819121718406
FOLD: 5, EPOCH: 24, train_loss: 0.013731136335199382
FOLD: 5, EPOCH: 24, valid_loss: 0.015963666029274463
FOLD: 6, EPOCH: 0, train_loss: 0.4790377524890462
FOLD: 6, EPOCH: 0, valid_loss: 0.023225703313946723
FOLD: 6, EPOCH: 1, train_loss: 0.021285045397829036
FOLD: 6, EPOCH: 1, valid_loss: 0.019108701795339585
FOLD: 6, EPOCH: 2, train_loss: 0.0189985391449462
FOLD: 6, EPOCH: 2, valid_loss: 0.017832795046269893
FOLD: 6, EPOCH: 3, train_loss: 0.017612096777527918
FOLD: 6, EPOCH: 3, valid_loss: 0.017242547310888767
FOLD: 6, EPOCH: 4, train_loss: 0.01709171023447903
FOLD: 6, EPOCH: 4, valid_loss: 0.01704309355467558
FOLD: 6, EPOCH: 5, train_loss: 0.01686613460001694
FOLD: 6, EPOCH: 5, valid_loss: 0.01701408553868532
FOLD: 6, EPOCH: 6, train_loss: 0.016994241834142988
FOLD: 6, EPOCH: 6, valid_loss: 0.01693613074719906
FOLD: 6, EPOCH: 7, train_loss: 0.01705680584826437
FOLD: 6, EPOCH: 7, valid_loss: 0.017081439830362796
FOLD: 6, EPOCH: 8, train_loss: 0.017034839122828577
FOLD: 6, EPOCH: 8, valid_loss: 0.01699220985174179
FOLD: 6, EPOCH: 9, train_loss: 0.01705682033761626
FOLD: 6, EPOCH: 9, valid_loss: 0.01700487956404686
FOLD: 6, EPOCH: 10, train_loss: 0.017008632402486946
FOLD: 6, EPOCH: 10, valid_loss: 0.017047495916485786
FOLD: 6, EPOCH: 11, train_loss: 0.016993061804092255
FOLD: 6, EPOCH: 11, valid_loss: 0.01685798816382885
FOLD: 6, EPOCH: 12, train_loss: 0.016903324306112567
FOLD: 6, EPOCH: 12, valid_loss: 0.016831180714070797
FOLD: 6, EPOCH: 13, train_loss: 0.01685638147957471
FOLD: 6, EPOCH: 13, valid_loss: 0.01682867631316185
FOLD: 6, EPOCH: 14, train_loss: 0.01669775856164645
FOLD: 6, EPOCH: 14, valid_loss: 0.016688687801361082
FOLD: 6, EPOCH: 15, train_loss: 0.016558770242394234
FOLD: 6, EPOCH: 15, valid_loss: 0.0166323122382164
FOLD: 6, EPOCH: 16, train_loss: 0.016438793308627443
FOLD: 6, EPOCH: 16, valid_loss: 0.016489243917167187
FOLD: 6, EPOCH: 17, train_loss: 0.016128870886655485
FOLD: 6, EPOCH: 17, valid_loss: 0.016444510705769064
FOLD: 6, EPOCH: 18, train_loss: 0.01590047376927267
FOLD: 6, EPOCH: 18, valid_loss: 0.016326313503086567
FOLD: 6, EPOCH: 19, train_loss: 0.015509134671669833
FOLD: 6, EPOCH: 19, valid_loss: 0.016117842011153696
FOLD: 6, EPOCH: 20, train_loss: 0.015127125088454915
FOLD: 6, EPOCH: 20, valid_loss: 0.015999867022037505
FOLD: 6, EPOCH: 21, train_loss: 0.014713663474789687
FOLD: 6, EPOCH: 21, valid_loss: 0.015923480950295926
```

```
FOLD: 6, EPOCH: 22, train_loss: 0.014294243062890712
FOLD: 6, EPOCH: 22, valid_loss: 0.015890509411692618
FOLD: 6, EPOCH: 23, train_loss: 0.013950457511355683
FOLD: 6, EPOCH: 23, valid_loss: 0.015896229930222036
FOLD: 6, EPOCH: 24, train_loss: 0.013774669311028354
FOLD: 6, EPOCH: 24, valid_loss: 0.015872968845069408
FOLD: 0, EPOCH: 0, train_loss: 0.48029174221058685
FOLD: 0, EPOCH: 0, valid_loss: 0.023946089521050453
FOLD: 0, EPOCH: 1, train_loss: 0.021173448206818834
FOLD: 0, EPOCH: 1, valid_loss: 0.01922344237565994
FOLD: 0, EPOCH: 2, train_loss: 0.019142101746888792
FOLD: 0, EPOCH: 2, valid_loss: 0.018406013026833534
FOLD: 0, EPOCH: 3, train_loss: 0.017672036014193176
FOLD: 0, EPOCH: 3, valid_loss: 0.017559948079288006
FOLD: 0, EPOCH: 4, train_loss: 0.017050023532795663
FOLD: 0, EPOCH: 4, valid_loss: 0.01729796651750803
FOLD: 0, EPOCH: 5, train_loss: 0.016908327928509843
FOLD: 0, EPOCH: 5, valid_loss: 0.017251189537346362
FOLD: 0, EPOCH: 6, train_loss: 0.01693144932287891
FOLD: 0, EPOCH: 6, valid_loss: 0.017039718329906462
FOLD: 0, EPOCH: 7, train_loss: 0.01697117228339724
FOLD: 0, EPOCH: 7, valid_loss: 0.017202776595950126
FOLD: 0, EPOCH: 8, train_loss: 0.01698530502110517
FOLD: 0, EPOCH: 8, valid_loss: 0.017240624614059924
FOLD: 0, EPOCH: 9, train_loss: 0.017020767814397406
FOLD: 0, EPOCH: 9, valid_loss: 0.017177668400108814
FOLD: 0, EPOCH: 10, train_loss: 0.01697178364281549
FOLD: 0, EPOCH: 10, valid_loss: 0.017272931598126887
FOLD: 0, EPOCH: 11, train_loss: 0.016960023471540738
FOLD: 0, EPOCH: 11, valid_loss: 0.017081333771348
FOLD: 0, EPOCH: 12, train_loss: 0.016869093722277354
FOLD: 0, EPOCH: 12, valid_loss: 0.017013381868600845
FOLD: 0, EPOCH: 13, train_loss: 0.016795190659408668
FOLD: 0, EPOCH: 13, valid_loss: 0.017021454088389874
FOLD: 0, EPOCH: 14, train_loss: 0.01668734685042683
FOLD: 0, EPOCH: 14, valid_loss: 0.016802685372531413
FOLD: 0, EPOCH: 15, train_loss: 0.016564986636849487
FOLD: 0, EPOCH: 15, valid_loss: 0.016594892628490926
FOLD: 0, EPOCH: 16, train_loss: 0.016364550255999272
FOLD: 0, EPOCH: 16, valid_loss: 0.016647035628557204
FOLD: 0, EPOCH: 17, train_loss: 0.0161791332953033
FOLD: 0, EPOCH: 17, valid_loss: 0.01646868985146284
FOLD: 0, EPOCH: 18, train_loss: 0.01586357123997747
FOLD: 0, EPOCH: 18, valid_loss: 0.016324181593954563
FOLD: 0, EPOCH: 19, train_loss: 0.015503082449744347
FOLD: 0, EPOCH: 19, valid_loss: 0.016219929270446302
FOLD: 0, EPOCH: 20, train_loss: 0.015137650522397083
FOLD: 0, EPOCH: 20, valid_loss: 0.01607202023267746
```

```
FOLD: 0, EPOCH: 21, train_loss: 0.014708368937630638
FOLD: 0, EPOCH: 21, valid_loss: 0.01595073163509369
FOLD: 0, EPOCH: 22, train_loss: nan
FOLD: 0, EPOCH: 22, valid_loss: nan
FOLD: 0, EPOCH: 23, train_loss: nan
FOLD: 0, EPOCH: 23, valid_loss: nan
FOLD: 0, EPOCH: 24, train_loss: nan
FOLD: 0, EPOCH: 24, valid_loss: nan
FOLD: 1, EPOCH: 0, train_loss: 0.4808223478901548
FOLD: 1, EPOCH: 0, valid_loss: 0.024812344163656234
FOLD: 1, EPOCH: 1, train_loss: 0.02115540918545658
FOLD: 1, EPOCH: 1, valid_loss: 0.019366511553525926
FOLD: 1, EPOCH: 2, train_loss: 0.01899107520272132
FOLD: 1, EPOCH: 2, valid_loss: 0.017867696993052958
FOLD: 1, EPOCH: 3, train_loss: 0.01775231577303945
FOLD: 1, EPOCH: 3, valid_loss: 0.017101817913353443
FOLD: 1, EPOCH: 4, train_loss: 0.017102348162051365
FOLD: 1, EPOCH: 4, valid_loss: 0.017038892880082132
FOLD: 1, EPOCH: 5, train_loss: 0.016938545741140842
FOLD: 1, EPOCH: 5, valid_loss: 0.01686871040612459
FOLD: 1, EPOCH: 6, train_loss: 0.0169917470743867
FOLD: 1, EPOCH: 6, valid_loss: 0.017256692871451377
FOLD: 1, EPOCH: 7, train_loss: 0.01705340986602566
FOLD: 1, EPOCH: 7, valid_loss: 0.017041142247617245
FOLD: 1, EPOCH: 8, train_loss: 0.017085544441883663
FOLD: 1, EPOCH: 8, valid_loss: 0.016932440251111985
FOLD: 1, EPOCH: 9, train_loss: 0.017071962375573967
FOLD: 1, EPOCH: 9, valid_loss: 0.017152712158858777
FOLD: 1, EPOCH: 10, train_loss: 0.016998695196951328
FOLD: 1, EPOCH: 10, valid_loss: 0.01681532636284828
FOLD: 1, EPOCH: 11, train_loss: 0.016969327510771702
FOLD: 1, EPOCH: 11, valid_loss: 0.016960955411195754
FOLD: 1, EPOCH: 12, train_loss: 0.01699229403316569
FOLD: 1, EPOCH: 12, valid_loss: 0.016877349540591238
FOLD: 1, EPOCH: 13, train_loss: 0.01691540795042604
FOLD: 1, EPOCH: 13, valid_loss: 0.016731238812208175
FOLD: 1, EPOCH: 14, train_loss: 0.016766769881201845
FOLD: 1, EPOCH: 14, valid_loss: 0.016514836959540844
FOLD: 1, EPOCH: 15, train_loss: 0.016568072013506272
FOLD: 1, EPOCH: 15, valid_loss: 0.016649024412035942
FOLD: 1, EPOCH: 16, train_loss: 0.016402926042574605
FOLD: 1, EPOCH: 16, valid_loss: 0.01629891209304333
FOLD: 1, EPOCH: 17, train_loss: 0.016123528028426527
FOLD: 1, EPOCH: 17, valid_loss: 0.01619778372347355
FOLD: 1, EPOCH: 18, train_loss: 0.01588142815293098
FOLD: 1, EPOCH: 18, valid_loss: 0.01607178956270218
FOLD: 1, EPOCH: 19, train_loss: 0.015564221806418733
FOLD: 1, EPOCH: 19, valid_loss: 0.015902577675879003
```

```
FOLD: 1, EPOCH: 20, train_loss: 0.015190921856888704
FOLD: 1, EPOCH: 20, valid_loss: 0.01582213941961527
FOLD: 1, EPOCH: 21, train_loss: 0.014766930564477736
FOLD: 1, EPOCH: 21, valid_loss: 0.015693071708083154
FOLD: 1, EPOCH: 22, train_loss: 0.014325511407385878
FOLD: 1, EPOCH: 22, valid_loss: 0.01564443428069353
FOLD: 1, EPOCH: 23, train_loss: 0.013968967224414252
FOLD: 1, EPOCH: 23, valid_loss: 0.015647099018096924
FOLD: 1, EPOCH: 24, train_loss: 0.013801887251284659
FOLD: 1, EPOCH: 24, valid_loss: 0.015616458244621753
FOLD: 2, EPOCH: 0, train_loss: 0.4808703261066456
FOLD: 2, EPOCH: 0, valid_loss: 0.024880244582891464
FOLD: 2, EPOCH: 1, train_loss: 0.02114043458282542
FOLD: 2, EPOCH: 1, valid_loss: 0.018960779719054698
FOLD: 2, EPOCH: 2, train_loss: 0.019015085028142344
FOLD: 2, EPOCH: 2, valid_loss: 0.018345214426517487
FOLD: 2, EPOCH: 3, train_loss: 0.017704928433205806
FOLD: 2, EPOCH: 3, valid_loss: 0.017417233884334565
FOLD: 2, EPOCH: 4, train_loss: 0.017057205670747626
FOLD: 2, EPOCH: 4, valid_loss: 0.017037723585963248
FOLD: 2, EPOCH: 5, train_loss: 0.016898751011764516
FOLD: 2, EPOCH: 5, valid_loss: 0.0169193072989583
FOLD: 2, EPOCH: 6, train_loss: 0.016926351035026467
FOLD: 2, EPOCH: 6, valid_loss: 0.017285697720944883
FOLD: 2, EPOCH: 7, train_loss: 0.016959929883125283
FOLD: 2, EPOCH: 7, valid_loss: 0.017161825001239778
FOLD: 2, EPOCH: 8, train_loss: 0.017022290457116097
FOLD: 2, EPOCH: 8, valid_loss: 0.01714812472462654
FOLD: 2, EPOCH: 9, train_loss: 0.016977866528695134
FOLD: 2, EPOCH: 9, valid_loss: 0.017125286757946015
FOLD: 2, EPOCH: 10, train_loss: 0.017052315264230684
FOLD: 2, EPOCH: 10, valid_loss: 0.017075384967029095
FOLD: 2, EPOCH: 11, train_loss: 0.016994265529016655
FOLD: 2, EPOCH: 11, valid_loss: 0.01701240710914135
FOLD: 2, EPOCH: 12, train_loss: 0.016901641197147824
FOLD: 2, EPOCH: 12, valid_loss: 0.017000651471316814
FOLD: 2, EPOCH: 13, train_loss: 0.016834127387272663
FOLD: 2, EPOCH: 13, valid_loss: 0.01689414996653795
FOLD: 2, EPOCH: 14, train_loss: 0.016715232789719186
FOLD: 2, EPOCH: 14, valid_loss: 0.01663567315787077
FOLD: 2, EPOCH: 15, train_loss: 0.01658406560340909
FOLD: 2, EPOCH: 15, valid_loss: 0.016509251743555067
FOLD: 2, EPOCH: 16, train_loss: 0.01629678507139083
FOLD: 2, EPOCH: 16, valid_loss: 0.016470525078475476
FOLD: 2, EPOCH: 17, train_loss: 0.016083555847254336
FOLD: 2, EPOCH: 17, valid_loss: 0.016295129656791685
FOLD: 2, EPOCH: 18, train_loss: 0.015821544852639948
FOLD: 2, EPOCH: 18, valid_loss: 0.016250660717487334
```

```
FOLD: 2, EPOCH: 19, train_loss: 0.015491206192595213
FOLD: 2, EPOCH: 19, valid_loss: 0.01606551095843315
FOLD: 2, EPOCH: 20, train_loss: 0.015117641506703939
FOLD: 2, EPOCH: 20, valid_loss: 0.01594421487301588
FOLD: 2, EPOCH: 21, train_loss: 0.014662557731376213
FOLD: 2, EPOCH: 21, valid_loss: 0.015954803302884103
FOLD: 2, EPOCH: 22, train_loss: 0.014225465942452959
FOLD: 2, EPOCH: 22, valid_loss: 0.015855884552001952
FOLD: 2, EPOCH: 23, train_loss: 0.013846278817830036
FOLD: 2, EPOCH: 23, valid_loss: 0.015838617049157618
FOLD: 2, EPOCH: 24, train_loss: 0.013679176854083732
FOLD: 2, EPOCH: 24, valid_loss: 0.015848777256906033
FOLD: 3, EPOCH: 0, train_loss: 0.48022048844366655
FOLD: 3, EPOCH: 0, valid_loss: 0.025026120617985724
FOLD: 3, EPOCH: 1, train_loss: 0.021201959323315395
FOLD: 3, EPOCH: 1, valid_loss: 0.01982655070722103
FOLD: 3, EPOCH: 2, train_loss: 0.018992610649541528
FOLD: 3, EPOCH: 2, valid_loss: 0.018130761012434958
FOLD: 3, EPOCH: 3, train_loss: 0.017696029481281635
FOLD: 3, EPOCH: 3, valid_loss: 0.01772117231041193
FOLD: 3, EPOCH: 4, train_loss: 0.017066594827793487
FOLD: 3, EPOCH: 4, valid_loss: 0.017235054858028887
FOLD: 3, EPOCH: 5, train_loss: 0.016899037612031917
FOLD: 3, EPOCH: 5, valid_loss: 0.017260446585714816
FOLD: 3, EPOCH: 6, train_loss: 0.016944109633260845
FOLD: 3, EPOCH: 6, valid_loss: 0.01716387253254652
FOLD: 3, EPOCH: 7, train_loss: 0.016942557213561877
FOLD: 3, EPOCH: 7, valid_loss: 0.01727043852210045
FOLD: 3, EPOCH: 8, train_loss: 0.017021945519309467
FOLD: 3, EPOCH: 8, valid_loss: 0.017490527667105196
FOLD: 3, EPOCH: 9, train_loss: 0.016970047708220626
FOLD: 3, EPOCH: 9, valid_loss: 0.017314641773700713
FOLD: 3, EPOCH: 10, train_loss: 0.01703660032984351
FOLD: 3, EPOCH: 10, valid_loss: 0.017168014980852603
FOLD: 3, EPOCH: 11, train_loss: 0.017008252251817256
FOLD: 3, EPOCH: 11, valid_loss: 0.017206613384187223
FOLD: 3, EPOCH: 12, train_loss: 0.016971083319917018
FOLD: 3, EPOCH: 12, valid_loss: 0.017085697539150714
FOLD: 3, EPOCH: 13, train_loss: 0.016848590167961558
FOLD: 3, EPOCH: 13, valid_loss: 0.017087050676345826
FOLD: 3, EPOCH: 14, train_loss: 0.016769131949665596
FOLD: 3, EPOCH: 14, valid_loss: 0.016855510249733925
FOLD: 3, EPOCH: 15, train_loss: 0.01655696764537672
FOLD: 3, EPOCH: 15, valid_loss: 0.016870710514485836
FOLD: 3, EPOCH: 16, train_loss: 0.016422529375421352
FOLD: 3, EPOCH: 16, valid_loss: 0.01657623626291752
FOLD: 3, EPOCH: 17, train_loss: 0.0162216602400148
FOLD: 3, EPOCH: 17, valid_loss: 0.016479881852865218
```

```
FOLD: 3, EPOCH: 18, train_loss: 0.01590132404777671
FOLD: 3, EPOCH: 18, valid_loss: 0.016390889473259448
FOLD: 3, EPOCH: 19, train_loss: 0.015590166609708954
FOLD: 3, EPOCH: 19, valid_loss: 0.016327331252396108
FOLD: 3, EPOCH: 20, train_loss: 0.015208811834961379
FOLD: 3, EPOCH: 20, valid_loss: 0.01625014327466488
FOLD: 3, EPOCH: 21, train_loss: 0.01479180298466869
FOLD: 3, EPOCH: 21, valid_loss: 0.016149142421782017
FOLD: 3, EPOCH: 22, train_loss: 0.014347572183730652
FOLD: 3, EPOCH: 22, valid_loss: 0.016116663217544555
FOLD: 3, EPOCH: 23, train_loss: 0.014029051367269486
FOLD: 3, EPOCH: 23, valid_loss: 0.016064374931156634
FOLD: 3, EPOCH: 24, train_loss: 0.013847806738043318
FOLD: 3, EPOCH: 24, valid_loss: 0.016081955023109912
FOLD: 4, EPOCH: 0, train_loss: 0.48073040288524566
FOLD: 4, EPOCH: 0, valid_loss: 0.024559471607208252
FOLD: 4, EPOCH: 1, train_loss: 0.021161683942793177
FOLD: 4, EPOCH: 1, valid_loss: 0.019294266253709794
FOLD: 4, EPOCH: 2, train_loss: 0.019046939808089716
FOLD: 4, EPOCH: 2, valid_loss: 0.018380920886993408
FOLD: 4, EPOCH: 3, train_loss: 0.017732687237463435
FOLD: 4, EPOCH: 3, valid_loss: 0.017370018064975738
FOLD: 4, EPOCH: 4, train_loss: 0.01706467647434903
FOLD: 4, EPOCH: 4, valid_loss: 0.017319457903504372
FOLD: 4, EPOCH: 5, train_loss: 0.016941019378247716
FOLD: 4, EPOCH: 5, valid_loss: 0.017407640255987643
FOLD: 4, EPOCH: 6, train_loss: 0.016957040122222333
FOLD: 4, EPOCH: 6, valid_loss: 0.01714132346212864
FOLD: 4, EPOCH: 7, train_loss: 0.01699092178320398
FOLD: 4, EPOCH: 7, valid_loss: 0.01713924393057823
FOLD: 4, EPOCH: 8, train_loss: 0.016979339063725098
FOLD: 4, EPOCH: 8, valid_loss: 0.0171297387778759
FOLD: 4, EPOCH: 9, train_loss: 0.01701943376133231
FOLD: 4, EPOCH: 9, valid_loss: 0.017131614424288275
FOLD: 4, EPOCH: 10, train_loss: 0.017023648559844413
FOLD: 4, EPOCH: 10, valid_loss: 0.017262357771396637
FOLD: 4, EPOCH: 11, train_loss: 0.0169220929775311
FOLD: 4, EPOCH: 11, valid_loss: 0.017273542881011964
FOLD: 4, EPOCH: 12, train_loss: 0.016883852783920003
FOLD: 4, EPOCH: 12, valid_loss: 0.01705174420028925
FOLD: 4, EPOCH: 13, train_loss: 0.016790613475046595
FOLD: 4, EPOCH: 13, valid_loss: 0.016909937635064126
FOLD: 4, EPOCH: 14, train_loss: 0.016687514095687542
FOLD: 4, EPOCH: 14, valid_loss: 0.016800466887652875
FOLD: 4, EPOCH: 15, train_loss: 0.016523402038530834
FOLD: 4, EPOCH: 15, valid_loss: 0.016744550913572312
FOLD: 4, EPOCH: 16, train_loss: 0.016377402710265852
FOLD: 4, EPOCH: 16, valid_loss: 0.01666022006422281
```

```
FOLD: 4, EPOCH: 17, train_loss: 0.016137564662514495
FOLD: 4, EPOCH: 17, valid_loss: 0.016513872742652893
FOLD: 4, EPOCH: 18, train_loss: 0.01591468625860352
FOLD: 4, EPOCH: 18, valid_loss: 0.01633517015725374
FOLD: 4, EPOCH: 19, train_loss: 0.01553330555887652
FOLD: 4, EPOCH: 19, valid_loss: 0.01624809794127941
FOLD: 4, EPOCH: 20, train_loss: 0.015102015953941816
FOLD: 4, EPOCH: 20, valid_loss: 0.01617756385356188
FOLD: 4, EPOCH: 21, train_loss: 0.014682901319952644
FOLD: 4, EPOCH: 21, valid_loss: 0.016047953367233275
FOLD: 4, EPOCH: 22, train_loss: 0.014287001706761162
FOLD: 4, EPOCH: 22, valid_loss: 0.016043743304908275
FOLD: 4, EPOCH: 23, train_loss: 0.01392691993617079
FOLD: 4, EPOCH: 23, valid_loss: 0.015999454706907272
FOLD: 4, EPOCH: 24, train_loss: 0.01376476760345454
FOLD: 4, EPOCH: 24, valid_loss: 0.015994838885962964
FOLD: 5, EPOCH: 0, train_loss: 0.48116053166944966
FOLD: 5, EPOCH: 0, valid_loss: 0.024474337995052338
FOLD: 5, EPOCH: 1, train_loss: 0.021242725276419905
FOLD: 5, EPOCH: 1, valid_loss: 0.019221568256616594
FOLD: 5, EPOCH: 2, train_loss: 0.01915504473584647
FOLD: 5, EPOCH: 2, valid_loss: 0.018231398835778238
FOLD: 5, EPOCH: 3, train_loss: 0.017711482251531817
FOLD: 5, EPOCH: 3, valid_loss: 0.01741752218455076
FOLD: 5, EPOCH: 4, train_loss: 0.017096172359322204
FOLD: 5, EPOCH: 4, valid_loss: 0.01714259386062622
FOLD: 5, EPOCH: 5, train_loss: 0.016921192578667282
FOLD: 5, EPOCH: 5, valid_loss: 0.01710904445499182
FOLD: 5, EPOCH: 6, train_loss: 0.01693627389692733
FOLD: 5, EPOCH: 6, valid_loss: 0.017178003080189228
FOLD: 5, EPOCH: 7, train_loss: 0.017021464893505686
FOLD: 5, EPOCH: 7, valid_loss: 0.017241457104682924
FOLD: 5, EPOCH: 8, train_loss: 0.01703550340290986
FOLD: 5, EPOCH: 8, valid_loss: 0.01723160222172737
FOLD: 5, EPOCH: 9, train_loss: 0.017027757504359396
FOLD: 5, EPOCH: 9, valid_loss: 0.01724115464836359
FOLD: 5, EPOCH: 10, train_loss: 0.016995188368319654
FOLD: 5, EPOCH: 10, valid_loss: 0.017287720255553723
FOLD: 5, EPOCH: 11, train_loss: 0.017002050581128417
FOLD: 5, EPOCH: 11, valid_loss: 0.01717138335108757
FOLD: 5, EPOCH: 12, train_loss: 0.01684502909454156
FOLD: 5, EPOCH: 12, valid_loss: 0.016990964524447916
FOLD: 5, EPOCH: 13, train_loss: 0.016880481332844616
FOLD: 5, EPOCH: 13, valid_loss: 0.016883419826626778
FOLD: 5, EPOCH: 14, train_loss: 0.016661927103996277
FOLD: 5, EPOCH: 14, valid_loss: 0.016981395594775676
FOLD: 5, EPOCH: 15, train_loss: 0.016550192013889752
FOLD: 5, EPOCH: 15, valid_loss: 0.016791253946721554
```

```
FOLD: 5, EPOCH: 16, train_loss: 0.016411791815340113
FOLD: 5, EPOCH: 16, valid_loss: 0.01667249407619238
FOLD: 5, EPOCH: 17, train_loss: 0.01615234107083204
FOLD: 5, EPOCH: 17, valid_loss: 0.016503089368343354
FOLD: 5, EPOCH: 18, train_loss: 0.01582860075520212
FOLD: 5, EPOCH: 18, valid_loss: 0.016374737471342087
FOLD: 5, EPOCH: 19, train_loss: 0.015570045372813332
FOLD: 5, EPOCH: 19, valid_loss: 0.01625288952142
FOLD: 5, EPOCH: 20, train_loss: 0.015160326092016129
FOLD: 5, EPOCH: 20, valid_loss: 0.016124554686248304
FOLD: 5, EPOCH: 21, train_loss: 0.014715300087316507
FOLD: 5, EPOCH: 21, valid_loss: 0.016048859022557736
FOLD: 5, EPOCH: 22, train_loss: 0.014300465564794687
FOLD: 5, EPOCH: 22, valid_loss: 0.016021119095385075
FOLD: 5, EPOCH: 23, train_loss: 0.01396394257440048
FOLD: 5, EPOCH: 23, valid_loss: 0.016000584326684476
FOLD: 5, EPOCH: 24, train_loss: 0.01380636921266512
FOLD: 5, EPOCH: 24, valid_loss: 0.016009703017771243
FOLD: 6, EPOCH: 0, train_loss: 0.48069661813883147
FOLD: 6, EPOCH: 0, valid_loss: 0.023286369815468787
FOLD: 6, EPOCH: 1, train_loss: 0.021443880803021443
FOLD: 6, EPOCH: 1, valid_loss: 0.020143647193908692
FOLD: 6, EPOCH: 2, train_loss: 0.019038551822811566
FOLD: 6, EPOCH: 2, valid_loss: 0.017763585150241852
FOLD: 6, EPOCH: 3, train_loss: 0.017684928318929104
FOLD: 6, EPOCH: 3, valid_loss: 0.017424733750522135
FOLD: 6, EPOCH: 4, train_loss: 0.017191975529552723
FOLD: 6, EPOCH: 4, valid_loss: 0.017044149041175842
FOLD: 6, EPOCH: 5, train_loss: 0.016916145279141915
FOLD: 6, EPOCH: 5, valid_loss: 0.016967807821929456
FOLD: 6, EPOCH: 6, train_loss: 0.016988575445855556
FOLD: 6, EPOCH: 6, valid_loss: 0.016934320889413358
FOLD: 6, EPOCH: 7, train_loss: 0.016994822681659742
FOLD: 6, EPOCH: 7, valid_loss: 0.017128295488655566
FOLD: 6, EPOCH: 8, train_loss: 0.017072013585644516
FOLD: 6, EPOCH: 8, valid_loss: 0.016926967017352582
FOLD: 6, EPOCH: 9, train_loss: 0.01701972638668657
FOLD: 6, EPOCH: 9, valid_loss: 0.016970829851925374
FOLD: 6, EPOCH: 10, train_loss: 0.017025553253891112
FOLD: 6, EPOCH: 10, valid_loss: 0.017021398656070233
FOLD: 6, EPOCH: 11, train_loss: 0.017016349360346794
FOLD: 6, EPOCH: 11, valid_loss: 0.016874165870249272
FOLD: 6, EPOCH: 12, train_loss: 0.016953294932031307
FOLD: 6, EPOCH: 12, valid_loss: 0.016836782544851304
FOLD: 6, EPOCH: 13, train_loss: 0.016824706109101268
FOLD: 6, EPOCH: 13, valid_loss: 0.01670189969241619
FOLD: 6, EPOCH: 14, train_loss: 0.01671939504769992
FOLD: 6, EPOCH: 14, valid_loss: 0.016921364329755308
```

```
FOLD: 6, EPOCH: 15, train_loss: 0.01661087765808211
FOLD: 6, EPOCH: 15, valid_loss: 0.01651741288602352
FOLD: 6, EPOCH: 16, train_loss: 0.016376040806220907
FOLD: 6, EPOCH: 16, valid_loss: 0.01645413354039192
FOLD: 6, EPOCH: 17, train_loss: 0.016144621933550655
FOLD: 6, EPOCH: 17, valid_loss: 0.01634560141712427
FOLD: 6, EPOCH: 18, train_loss: 0.015917793657554656
FOLD: 6, EPOCH: 18, valid_loss: 0.01631132557988167
FOLD: 6, EPOCH: 19, train_loss: 0.015620657908064979
FOLD: 6, EPOCH: 19, valid_loss: 0.016172146275639534
FOLD: 6, EPOCH: 20, train_loss: 0.01522642927130266
FOLD: 6, EPOCH: 20, valid_loss: 0.016033906787633896
FOLD: 6, EPOCH: 21, train_loss: 0.014799048896042668
FOLD: 6, EPOCH: 21, valid_loss: 0.015908198803663252
FOLD: 6, EPOCH: 22, train_loss: 0.014393271599598483
FOLD: 6, EPOCH: 22, valid_loss: 0.015832398273050784
FOLD: 6, EPOCH: 23, train_loss: 0.014091360864235836
FOLD: 6, EPOCH: 23, valid_loss: 0.015793926268815994
FOLD: 6, EPOCH: 24, train_loss: 0.013875825140549213
FOLD: 6, EPOCH: 24, valid_loss: 0.015786710567772387
FOLD: 0, EPOCH: 0, train_loss: 0.4801790922322646
FOLD: 0, EPOCH: 0, valid_loss: 0.023799920082092287
FOLD: 0, EPOCH: 1, train_loss: 0.021241431257554462
FOLD: 0, EPOCH: 1, valid_loss: 0.02137688025832176
FOLD: 0, EPOCH: 2, train_loss: 0.018939682789349637
FOLD: 0, EPOCH: 2, valid_loss: 0.018614307641983033
FOLD: 0, EPOCH: 3, train_loss: 0.01759881474913991
FOLD: 0, EPOCH: 3, valid_loss: 0.017246431857347488
FOLD: 0, EPOCH: 4, train_loss: 0.017055970166816193
FOLD: 0, EPOCH: 4, valid_loss: 0.01707864422351122
FOLD: 0, EPOCH: 5, train_loss: 0.016846038426468972
FOLD: 0, EPOCH: 5, valid_loss: 0.017118188627064226
FOLD: 0, EPOCH: 6, train_loss: 0.01691555954059776
FOLD: 0, EPOCH: 6, valid_loss: 0.01712479542940855
FOLD: 0, EPOCH: 7, train_loss: 0.01697511139142067
FOLD: 0, EPOCH: 7, valid_loss: 0.017056205458939076
FOLD: 0, EPOCH: 8, train_loss: 0.01699636621261332
FOLD: 0, EPOCH: 8, valid_loss: 0.01707579880952835
FOLD: 0, EPOCH: 9, train_loss: 0.01703557145914861
FOLD: 0, EPOCH: 9, valid_loss: 0.017248324863612653
FOLD: 0, EPOCH: 10, train_loss: 0.017015571718668045
FOLD: 0, EPOCH: 10, valid_loss: 0.01702596791088581
FOLD: 0, EPOCH: 11, train_loss: 0.016915575518798666
FOLD: 0, EPOCH: 11, valid_loss: 0.01709539569914341
FOLD: 0, EPOCH: 12, train_loss: 0.016893773523419083
FOLD: 0, EPOCH: 12, valid_loss: 0.016936002783477306
FOLD: 0, EPOCH: 13, train_loss: 0.016803724894008668
FOLD: 0, EPOCH: 13, valid_loss: 0.016830314248800278
```

```
FOLD: 0, EPOCH: 14, train_loss: 0.0167257884504641
FOLD: 0, EPOCH: 14, valid_loss: 0.01683069221675396
FOLD: 0, EPOCH: 15, train_loss: 0.016524724016080097
FOLD: 0, EPOCH: 15, valid_loss: 0.016618845649063588
FOLD: 0, EPOCH: 16, train_loss: 0.016327559941632
FOLD: 0, EPOCH: 16, valid_loss: 0.016541500575840473
FOLD: 0, EPOCH: 17, train_loss: 0.016116418913767045
FOLD: 0, EPOCH: 17, valid_loss: 0.016416672617197037
FOLD: 0, EPOCH: 18, train_loss: 0.015829356412599686
FOLD: 0, EPOCH: 18, valid_loss: 0.016321938820183277
FOLD: 0, EPOCH: 19, train_loss: 0.015514894166872614
FOLD: 0, EPOCH: 19, valid_loss: 0.016183889545500277
FOLD: 0, EPOCH: 20, train_loss: 0.01512317218165211
FOLD: 0, EPOCH: 20, valid_loss: 0.01606907844543457
FOLD: 0, EPOCH: 21, train_loss: 0.014639224185525965
FOLD: 0, EPOCH: 21, valid_loss: 0.016012586280703546
FOLD: 0, EPOCH: 22, train_loss: 0.0141888755772795
FOLD: 0, EPOCH: 22, valid_loss: 0.015959781520068647
FOLD: 0, EPOCH: 23, train_loss: 0.013825368871088742
FOLD: 0, EPOCH: 23, valid_loss: 0.0159478060901165
FOLD: 0, EPOCH: 24, train_loss: 0.013696176931262016
FOLD: 0, EPOCH: 24, valid_loss: 0.015940846726298332
FOLD: 1, EPOCH: 0, train_loss: 0.480586217209494
FOLD: 1, EPOCH: 0, valid_loss: 0.024636338278651238
FOLD: 1, EPOCH: 1, train_loss: 0.021250638578619276
FOLD: 1, EPOCH: 1, valid_loss: 0.02024312488734722
FOLD: 1, EPOCH: 2, train_loss: 0.01927307751789993
FOLD: 1, EPOCH: 2, valid_loss: 0.018100381195545197
FOLD: 1, EPOCH: 3, train_loss: 0.017879087858054102
FOLD: 1, EPOCH: 3, valid_loss: 0.01707018718123436
FOLD: 1, EPOCH: 4, train_loss: 0.01713164656923539
FOLD: 1, EPOCH: 4, valid_loss: 0.016970947869122028
FOLD: 1, EPOCH: 5, train_loss: 0.016985075694008343
FOLD: 1, EPOCH: 5, valid_loss: 0.01687811132520437
FOLD: 1, EPOCH: 6, train_loss: 0.0169581462420067
FOLD: 1, EPOCH: 6, valid_loss: 0.017028269171714783
FOLD: 1, EPOCH: 7, train_loss: 0.0170263366418935
FOLD: 1, EPOCH: 7, valid_loss: 0.017032364681363106
FOLD: 1, EPOCH: 8, train_loss: 0.01709956949462696
FOLD: 1, EPOCH: 8, valid_loss: 0.016859473325312138
FOLD: 1, EPOCH: 9, train_loss: 0.017098259516567193
FOLD: 1, EPOCH: 9, valid_loss: 0.016962896473705767
FOLD: 1, EPOCH: 10, train_loss: 0.017093646004289188
FOLD: 1, EPOCH: 10, valid_loss: 0.017253718450665475
FOLD: 1, EPOCH: 11, train_loss: 0.017038007501234003
FOLD: 1, EPOCH: 11, valid_loss: 0.016818967461586
FOLD: 1, EPOCH: 12, train_loss: 0.016945465740297927
FOLD: 1, EPOCH: 12, valid_loss: 0.016678593531250955
```

```
FOLD: 1, EPOCH: 13, train_loss: 0.016876455339394053
FOLD: 1, EPOCH: 13, valid_loss: 0.016753439530730246
FOLD: 1, EPOCH: 14, train_loss: 0.016757801346176743
FOLD: 1, EPOCH: 14, valid_loss: 0.016574453078210354
FOLD: 1, EPOCH: 15, train_loss: 0.016665755904146602
FOLD: 1, EPOCH: 15, valid_loss: 0.016427539624273776
FOLD: 1, EPOCH: 16, train_loss: 0.016440928147054043
FOLD: 1, EPOCH: 16, valid_loss: 0.016373175419867037
FOLD: 1, EPOCH: 17, train_loss: 0.01624157854362207
FOLD: 1, EPOCH: 17, valid_loss: 0.016199312694370745
FOLD: 1, EPOCH: 18, train_loss: 0.01599305000815059
FOLD: 1, EPOCH: 18, valid_loss: 0.016111699417233467
FOLD: 1, EPOCH: 19, train_loss: 0.015669279003224405
FOLD: 1, EPOCH: 19, valid_loss: 0.015879976823925973
FOLD: 1, EPOCH: 20, train_loss: 0.015247915833428199
FOLD: 1, EPOCH: 20, valid_loss: 0.0158562221378088
FOLD: 1, EPOCH: 21, train_loss: 0.014799268745190027
FOLD: 1, EPOCH: 21, valid_loss: 0.015675219930708407
FOLD: 1, EPOCH: 22, train_loss: 0.014388745422570072
FOLD: 1, EPOCH: 22, valid_loss: 0.015622003600001336
FOLD: 1, EPOCH: 23, train_loss: 0.01407485866450331
FOLD: 1, EPOCH: 23, valid_loss: 0.015628491677343846
FOLD: 1, EPOCH: 24, train_loss: 0.013886934450390388
FOLD: 1, EPOCH: 24, valid_loss: 0.015635842867195607
FOLD: 2, EPOCH: 0, train_loss: 0.4810004333908461
FOLD: 2, EPOCH: 0, valid_loss: 0.024735444933176042
FOLD: 2, EPOCH: 1, train_loss: 0.021137115260472104
FOLD: 2, EPOCH: 1, valid_loss: 0.019903352856636046
FOLD: 2, EPOCH: 2, train_loss: 0.018872500365587318
FOLD: 2, EPOCH: 2, valid_loss: 0.0180094014108181
FOLD: 2, EPOCH: 3, train_loss: 0.017690133879936877
FOLD: 2, EPOCH: 3, valid_loss: 0.01744071990251541
FOLD: 2, EPOCH: 4, train_loss: 0.017006052467896014
FOLD: 2, EPOCH: 4, valid_loss: 0.017193730734288693
FOLD: 2, EPOCH: 5, train_loss: 0.016920497184809372
FOLD: 2, EPOCH: 5, valid_loss: 0.01709017377346754
FOLD: 2, EPOCH: 6, train_loss: 0.016989171112171648
FOLD: 2, EPOCH: 6, valid_loss: 0.01732622690498829
FOLD: 2, EPOCH: 7, train_loss: 0.017020373757244373
FOLD: 2, EPOCH: 7, valid_loss: 0.01702039673924446
FOLD: 2, EPOCH: 8, train_loss: 0.017053119996625954
FOLD: 2, EPOCH: 8, valid_loss: 0.01704077523201704
FOLD: 2, EPOCH: 9, train_loss: 0.01702507071475796
FOLD: 2, EPOCH: 9, valid_loss: 0.01705407686531544
FOLD: 2, EPOCH: 10, train_loss: 0.0170092820474992
FOLD: 2, EPOCH: 10, valid_loss: 0.017718366943299777
FOLD: 2, EPOCH: 11, train_loss: 0.016954233635495714
FOLD: 2, EPOCH: 11, valid_loss: 0.01696766585111618
```

```
FOLD: 2, EPOCH: 12, train_loss: 0.01688324576433824
FOLD: 2, EPOCH: 12, valid_loss: 0.016922266930341722
FOLD: 2, EPOCH: 13, train_loss: 0.01679924172254241
FOLD: 2, EPOCH: 13, valid_loss: 0.016853889711201192
FOLD: 2, EPOCH: 14, train_loss: 0.016683703573534683
FOLD: 2, EPOCH: 14, valid_loss: 0.01664580937474966
FOLD: 2, EPOCH: 15, train_loss: 0.016548771886345074
FOLD: 2, EPOCH: 15, valid_loss: 0.016753433123230934
FOLD: 2, EPOCH: 16, train_loss: 0.016376529047329003
FOLD: 2, EPOCH: 16, valid_loss: 0.016501575261354446
FOLD: 2, EPOCH: 17, train_loss: 0.016117891163698266
FOLD: 2, EPOCH: 17, valid_loss: 0.016333647333085538
FOLD: 2, EPOCH: 18, train_loss: 0.015830700786239434
FOLD: 2, EPOCH: 18, valid_loss: 0.01626410823315382
FOLD: 2, EPOCH: 19, train_loss: 0.015528523895357336
FOLD: 2, EPOCH: 19, valid_loss: 0.016095304638147356
FOLD: 2, EPOCH: 20, train_loss: 0.01520731286810977
FOLD: 2, EPOCH: 20, valid_loss: 0.01598656512796879
FOLD: 2, EPOCH: 21, train_loss: nan
FOLD: 2, EPOCH: 21, valid_loss: nan
FOLD: 2, EPOCH: 22, train_loss: nan
FOLD: 2, EPOCH: 22, valid_loss: nan
FOLD: 2, EPOCH: 23, train_loss: nan
FOLD: 2, EPOCH: 23, valid_loss: nan
FOLD: 2, EPOCH: 24, train_loss: nan
FOLD: 2, EPOCH: 24, valid_loss: nan
FOLD: 3, EPOCH: 0, train_loss: 0.48026267402380907
FOLD: 3, EPOCH: 0, valid_loss: 0.023603443577885626
FOLD: 3, EPOCH: 1, train_loss: 0.02127988941866119
FOLD: 3, EPOCH: 1, valid_loss: 0.0193124745041132
FOLD: 3, EPOCH: 2, train_loss: 0.018846164223085455
FOLD: 3, EPOCH: 2, valid_loss: 0.0180340064316988
FOLD: 3, EPOCH: 3, train_loss: 0.017616553808606807
FOLD: 3, EPOCH: 3, valid_loss: 0.01754094399511814
FOLD: 3, EPOCH: 4, train_loss: 0.017049466266113075
FOLD: 3, EPOCH: 4, valid_loss: 0.017003291696310045
FOLD: 3, EPOCH: 5, train_loss: 0.01687854209116527
FOLD: 3, EPOCH: 5, valid_loss: 0.017155957855284214
FOLD: 3, EPOCH: 6, train_loss: 0.016899211560280954
FOLD: 3, EPOCH: 6, valid_loss: 0.01712133940309286
FOLD: 3, EPOCH: 7, train_loss: 0.016994704302324325
FOLD: 3, EPOCH: 7, valid_loss: 0.017170774266123772
FOLD: 3, EPOCH: 8, train_loss: 0.017052414560956613
FOLD: 3, EPOCH: 8, valid_loss: 0.017301854491233826
FOLD: 3, EPOCH: 9, train_loss: 0.017029430045663905
FOLD: 3, EPOCH: 9, valid_loss: 0.017286727353930473
FOLD: 3, EPOCH: 10, train_loss: 0.01709546070850017
FOLD: 3, EPOCH: 10, valid_loss: 0.01736246459186077
```

```
FOLD: 3, EPOCH: 11, train_loss: 0.017066966444507344
FOLD: 3, EPOCH: 11, valid_loss: 0.017027986720204353
FOLD: 3, EPOCH: 12, train_loss: 0.01691402067259258
FOLD: 3, EPOCH: 12, valid_loss: 0.01716131079941988
FOLD: 3, EPOCH: 13, train_loss: 0.01686450582453791
FOLD: 3, EPOCH: 13, valid_loss: 0.016957219429314135
FOLD: 3, EPOCH: 14, train_loss: 0.016771229991347204
FOLD: 3, EPOCH: 14, valid_loss: 0.017013451009988786
FOLD: 3, EPOCH: 15, train_loss: 0.016563913639204033
FOLD: 3, EPOCH: 15, valid_loss: 0.01666126072406769
FOLD: 3, EPOCH: 16, train_loss: 0.016382265721960942
FOLD: 3, EPOCH: 16, valid_loss: 0.01659337256103754
FOLD: 3, EPOCH: 17, train_loss: 0.01616077975933852
FOLD: 3, EPOCH: 17, valid_loss: 0.016529491730034353
FOLD: 3, EPOCH: 18, train_loss: 0.015928532383372995
FOLD: 3, EPOCH: 18, valid_loss: 0.016307784616947173
FOLD: 3, EPOCH: 19, train_loss: 0.015571470290016966
FOLD: 3, EPOCH: 19, valid_loss: 0.016290651448071004
FOLD: 3, EPOCH: 20, train_loss: 0.015162886044352638
FOLD: 3, EPOCH: 20, valid_loss: 0.01614811774343252
FOLD: 3, EPOCH: 21, train_loss: 0.014754361276622533
FOLD: 3, EPOCH: 21, valid_loss: 0.016083087250590326
FOLD: 3, EPOCH: 22, train_loss: 0.014374068697445652
FOLD: 3, EPOCH: 22, valid_loss: 0.01601563472300768
FOLD: 3, EPOCH: 23, train_loss: 0.014044325855769674
FOLD: 3, EPOCH: 23, valid_loss: 0.01602879587560892
FOLD: 3, EPOCH: 24, train_loss: 0.013845297096132421
FOLD: 3, EPOCH: 24, valid_loss: 0.016013798862695695
FOLD: 4, EPOCH: 0, train_loss: 0.480757159934867
FOLD: 4, EPOCH: 0, valid_loss: 0.024627946093678476
FOLD: 4, EPOCH: 1, train_loss: 0.021106504150951395
FOLD: 4, EPOCH: 1, valid_loss: 0.019006762430071832
FOLD: 4, EPOCH: 2, train_loss: 0.01920628222655885
FOLD: 4, EPOCH: 2, valid_loss: 0.018068747967481612
FOLD: 4, EPOCH: 3, train_loss: 0.017632073456687585
FOLD: 4, EPOCH: 3, valid_loss: 0.017374009639024735
FOLD: 4, EPOCH: 4, train_loss: 0.017114745788783036
FOLD: 4, EPOCH: 4, valid_loss: 0.01719771306961775
FOLD: 4, EPOCH: 5, train_loss: 0.01692720724358445
FOLD: 4, EPOCH: 5, valid_loss: 0.017051609046757222
FOLD: 4, EPOCH: 6, train_loss: 0.01696216263713277
FOLD: 4, EPOCH: 6, valid_loss: 0.01725705824792385
FOLD: 4, EPOCH: 7, train_loss: 0.017051527878510303
FOLD: 4, EPOCH: 7, valid_loss: 0.01725944221019745
FOLD: 4, EPOCH: 8, train_loss: 0.01705340800338051
FOLD: 4, EPOCH: 8, valid_loss: 0.01716303113847971
FOLD: 4, EPOCH: 9, train_loss: 0.017092429506940905
FOLD: 4, EPOCH: 9, valid_loss: 0.01714508727192879
```

```
FOLD: 4, EPOCH: 10, train_loss: 0.01699684414581782
FOLD: 4, EPOCH: 10, valid_loss: 0.017647585719823836
FOLD: 4, EPOCH: 11, train_loss: 0.01703640858509711
FOLD: 4, EPOCH: 11, valid_loss: 0.017148720137774943
FOLD: 4, EPOCH: 12, train_loss: 0.016899092800813875
FOLD: 4, EPOCH: 12, valid_loss: 0.01697548869997263
FOLD: 4, EPOCH: 13, train_loss: 0.016896867148932956
FOLD: 4, EPOCH: 13, valid_loss: 0.01697954084724188
FOLD: 4, EPOCH: 14, train_loss: 0.016770316237190955
FOLD: 4, EPOCH: 14, valid_loss: 0.016836495585739613
FOLD: 4, EPOCH: 15, train_loss: 0.016583956651338914
FOLD: 4, EPOCH: 15, valid_loss: 0.01671934552490711
FOLD: 4, EPOCH: 16, train_loss: 0.016441541730242523
FOLD: 4, EPOCH: 16, valid_loss: 0.016640160344541074
FOLD: 4, EPOCH: 17, train_loss: 0.016216889213846654
FOLD: 4, EPOCH: 17, valid_loss: 0.016638977155089378
FOLD: 4, EPOCH: 18, train_loss: 0.015947292007657946
FOLD: 4, EPOCH: 18, valid_loss: 0.016456627659499644
FOLD: 4, EPOCH: 19, train_loss: 0.015645589109282106
FOLD: 4, EPOCH: 19, valid_loss: 0.016213668063282966
FOLD: 4, EPOCH: 20, train_loss: 0.015247807046081745
FOLD: 4, EPOCH: 20, valid_loss: 0.01622691411525011
FOLD: 4, EPOCH: 21, train_loss: 0.014824620916881935
FOLD: 4, EPOCH: 21, valid_loss: 0.016027042753994466
FOLD: 4, EPOCH: 22, train_loss: 0.014433468774366541
FOLD: 4, EPOCH: 22, valid_loss: 0.01601828943938017
FOLD: 4, EPOCH: 23, train_loss: 0.014115711845153448
FOLD: 4, EPOCH: 23, valid_loss: 0.01597691584378481
FOLD: 4, EPOCH: 24, train_loss: 0.013958170845293674
FOLD: 4, EPOCH: 24, valid_loss: 0.015961052402853965
FOLD: 5, EPOCH: 0, train_loss: 0.4817004786664937
FOLD: 5, EPOCH: 0, valid_loss: 0.02631576284766197
FOLD: 5, EPOCH: 1, train_loss: 0.021211522219537877
FOLD: 5, EPOCH: 1, valid_loss: 0.019231923446059226
FOLD: 5, EPOCH: 2, train_loss: 0.018846051723119758
FOLD: 5, EPOCH: 2, valid_loss: 0.01855659395456314
FOLD: 5, EPOCH: 3, train_loss: 0.018088463237699196
FOLD: 5, EPOCH: 3, valid_loss: 0.01740754470229149
FOLD: 5, EPOCH: 4, train_loss: 0.017123783019935192
FOLD: 5, EPOCH: 4, valid_loss: 0.017156889513134956
FOLD: 5, EPOCH: 5, train_loss: 0.016929619540111955
FOLD: 5, EPOCH: 5, valid_loss: 0.017211403474211694
FOLD: 5, EPOCH: 6, train_loss: 0.017017919152062768
FOLD: 5, EPOCH: 6, valid_loss: 0.01724058374762535
FOLD: 5, EPOCH: 7, train_loss: 0.017011427897072974
FOLD: 5, EPOCH: 7, valid_loss: 0.017427316606044768
FOLD: 5, EPOCH: 8, train_loss: 0.01702644994013569
FOLD: 5, EPOCH: 8, valid_loss: 0.017102975957095624
```

```
FOLD: 5, EPOCH: 9, train_loss: 0.017091242767566323
FOLD: 5, EPOCH: 9, valid_loss: 0.017263008318841457
FOLD: 5, EPOCH: 10, train_loss: 0.01695686904396735
FOLD: 5, EPOCH: 10, valid_loss: 0.016991139352321626
FOLD: 5, EPOCH: 11, train_loss: 0.01695215034925816
FOLD: 5, EPOCH: 11, valid_loss: 0.017281469367444516
FOLD: 5, EPOCH: 12, train_loss: 0.016969179506508673
FOLD: 5, EPOCH: 12, valid_loss: 0.01711196843534708
FOLD: 5, EPOCH: 13, train_loss: 0.016829287772681438
FOLD: 5, EPOCH: 13, valid_loss: 0.017019381299614906
FOLD: 5, EPOCH: 14, train_loss: 0.01675671997929917
FOLD: 5, EPOCH: 14, valid_loss: 0.01700835958123207
FOLD: 5, EPOCH: 15, train_loss: 0.01658959294586968
FOLD: 5, EPOCH: 15, valid_loss: 0.016737351715564727
FOLD: 5, EPOCH: 16, train_loss: 0.01636157222852415
FOLD: 5, EPOCH: 16, valid_loss: 0.016615087687969207
FOLD: 5, EPOCH: 17, train_loss: 0.01614290984588213
FOLD: 5, EPOCH: 17, valid_loss: 0.0165150111913681
FOLD: 5, EPOCH: 18, train_loss: 0.015906118692792192
FOLD: 5, EPOCH: 18, valid_loss: 0.016381737254559995
FOLD: 5, EPOCH: 19, train_loss: 0.01551560347988492
FOLD: 5, EPOCH: 19, valid_loss: 0.01622542444616556
FOLD: 5, EPOCH: 20, train_loss: 0.015144251217292685
FOLD: 5, EPOCH: 20, valid_loss: 0.016103848926723004
FOLD: 5, EPOCH: 21, train_loss: 0.01470786337518976
FOLD: 5, EPOCH: 21, valid_loss: 0.015999515987932682
FOLD: 5, EPOCH: 22, train_loss: 0.014268012654010941
FOLD: 5, EPOCH: 22, valid_loss: 0.015963018275797367
FOLD: 5, EPOCH: 23, train_loss: 0.013929354863203302
FOLD: 5, EPOCH: 23, valid_loss: 0.015894265100359916
FOLD: 5, EPOCH: 24, train_loss: 0.013738843371622822
FOLD: 5, EPOCH: 24, valid_loss: 0.01591346774250269
FOLD: 6, EPOCH: 0, train_loss: 0.480884503281745
FOLD: 6, EPOCH: 0, valid_loss: 0.024338292554020882
FOLD: 6, EPOCH: 1, train_loss: 0.02106337726977812
FOLD: 6, EPOCH: 1, valid_loss: 0.018769282922148703
FOLD: 6, EPOCH: 2, train_loss: 0.018885441776998593
FOLD: 6, EPOCH: 2, valid_loss: 0.017769076898694038
FOLD: 6, EPOCH: 3, train_loss: 0.017746354592647277
FOLD: 6, EPOCH: 3, valid_loss: 0.017378076314926147
FOLD: 6, EPOCH: 4, train_loss: 0.017054137013548493
FOLD: 6, EPOCH: 4, valid_loss: 0.017011654525995255
FOLD: 6, EPOCH: 5, train_loss: 0.016957850233480638
FOLD: 6, EPOCH: 5, valid_loss: 0.01698651019483805
FOLD: 6, EPOCH: 6, train_loss: 0.01696864509841009
FOLD: 6, EPOCH: 6, valid_loss: 0.017104379609227182
FOLD: 6, EPOCH: 7, train_loss: 0.017017859712141713
FOLD: 6, EPOCH: 7, valid_loss: 0.01716637209057808
```

```
FOLD: 6, EPOCH: 8, train_loss: 0.01703595353581873
FOLD: 6, EPOCH: 8, valid_loss: 0.01692531503736973
FOLD: 6, EPOCH: 9, train_loss: 0.017107198496355492
FOLD: 6, EPOCH: 9, valid_loss: 0.01702692996710539
FOLD: 6, EPOCH: 10, train_loss: 0.017118099817157197
FOLD: 6, EPOCH: 10, valid_loss: 0.01683626864105463
FOLD: 6, EPOCH: 11, train_loss: 0.016980562675871006
FOLD: 6, EPOCH: 11, valid_loss: 0.017214697115123273
FOLD: 6, EPOCH: 12, train_loss: 0.016894040768649303
FOLD: 6, EPOCH: 12, valid_loss: 0.016831023693084715
FOLD: 6, EPOCH: 13, train_loss: 0.016869858008663672
FOLD: 6, EPOCH: 13, valid_loss: 0.016787173599004744
FOLD: 6, EPOCH: 14, train_loss: 0.01673049947918475
FOLD: 6, EPOCH: 14, valid_loss: 0.016798882111907006
FOLD: 6, EPOCH: 15, train_loss: 0.016630030781993654
FOLD: 6, EPOCH: 15, valid_loss: 0.016552005223929883
FOLD: 6, EPOCH: 16, train_loss: 0.016441274453334664
FOLD: 6, EPOCH: 16, valid_loss: 0.016503336764872076
FOLD: 6, EPOCH: 17, train_loss: 0.016166910516465603
FOLD: 6, EPOCH: 17, valid_loss: 0.016459756940603257
FOLD: 6, EPOCH: 18, train_loss: 0.015884976941762732
FOLD: 6, EPOCH: 18, valid_loss: 0.016213380359113218
FOLD: 6, EPOCH: 19, train_loss: 0.015507560432413402
FOLD: 6, EPOCH: 19, valid_loss: 0.016112008318305015
FOLD: 6, EPOCH: 20, train_loss: 0.015167489375437604
FOLD: 6, EPOCH: 20, valid_loss: 0.015988050512969493
FOLD: 6, EPOCH: 21, train_loss: 0.01475349167475895
FOLD: 6, EPOCH: 21, valid_loss: 0.015977313555777074
FOLD: 6, EPOCH: 22, train_loss: 0.014358022490985134
FOLD: 6, EPOCH: 22, valid_loss: 0.01586295008659363
FOLD: 6, EPOCH: 23, train_loss: 0.013993104248225283
FOLD: 6, EPOCH: 23, valid_loss: 0.015844721645116806
FOLD: 6, EPOCH: 24, train_loss: 0.013837955100145065
FOLD: 6, EPOCH: 24, valid_loss: 0.015858473367989064
FOLD: 0, EPOCH: 0, train_loss: 0.4803789014845681
FOLD: 0, EPOCH: 0, valid_loss: 0.02423692189157009
FOLD: 0, EPOCH: 1, train_loss: 0.021213667784012905
FOLD: 0, EPOCH: 1, valid_loss: 0.019322006478905678
FOLD: 0, EPOCH: 2, train_loss: 0.018905291188077458
FOLD: 0, EPOCH: 2, valid_loss: 0.01823759004473686
FOLD: 0, EPOCH: 3, train_loss: 0.01768959251542886
FOLD: 0, EPOCH: 3, valid_loss: 0.01765403836965561
FOLD: 0, EPOCH: 4, train_loss: 0.017056693056864396
FOLD: 0, EPOCH: 4, valid_loss: 0.01711706191301346
FOLD: 0, EPOCH: 5, train_loss: 0.01693661812515486
FOLD: 0, EPOCH: 5, valid_loss: 0.017223269902169703
FOLD: 0, EPOCH: 6, train_loss: 0.016975289217012676
FOLD: 0, EPOCH: 6, valid_loss: 0.01722891677170992
```

```
FOLD: 0, EPOCH: 7, train_loss: 0.01704602686016738
FOLD: 0, EPOCH: 7, valid_loss: 0.017178979106247425
FOLD: 0, EPOCH: 8, train_loss: 0.016986528031375944
FOLD: 0, EPOCH: 8, valid_loss: 0.017752952799201013
FOLD: 0, EPOCH: 9, train_loss: 0.017105353273907487
FOLD: 0, EPOCH: 9, valid_loss: 0.017301602363586424
FOLD: 0, EPOCH: 10, train_loss: 0.017045353317544573
FOLD: 0, EPOCH: 10, valid_loss: 0.017214150168001652
FOLD: 0, EPOCH: 11, train_loss: 0.01701382112999757
FOLD: 0, EPOCH: 11, valid_loss: 0.017178190499544145
FOLD: 0, EPOCH: 12, train_loss: 0.01696837490715948
FOLD: 0, EPOCH: 12, valid_loss: 0.016944194100797175
FOLD: 0, EPOCH: 13, train_loss: 0.01680966264561934
FOLD: 0, EPOCH: 13, valid_loss: 0.01686282854527235
FOLD: 0, EPOCH: 14, train_loss: 0.01678817011542669
FOLD: 0, EPOCH: 14, valid_loss: 0.016908331513404845
FOLD: 0, EPOCH: 15, train_loss: 0.016546346665341025
FOLD: 0, EPOCH: 15, valid_loss: 0.016767378486692904
FOLD: 0, EPOCH: 16, train_loss: 0.01639360764703783
FOLD: 0, EPOCH: 16, valid_loss: 0.016654736511409283
FOLD: 0, EPOCH: 17, train_loss: 0.016151041172596872
FOLD: 0, EPOCH: 17, valid_loss: 0.016599762029945852
FOLD: 0, EPOCH: 18, train_loss: 0.015907456141700145
FOLD: 0, EPOCH: 18, valid_loss: 0.016315866373479367
FOLD: 0, EPOCH: 19, train_loss: 0.015527582201523846
FOLD: 0, EPOCH: 19, valid_loss: 0.016240798234939575
FOLD: 0, EPOCH: 20, train_loss: 0.015213290444847679
FOLD: 0, EPOCH: 20, valid_loss: 0.016115375123918058
FOLD: 0, EPOCH: 21, train_loss: 0.014738682524434157
FOLD: 0, EPOCH: 21, valid_loss: 0.015978024527430534
FOLD: 0, EPOCH: 22, train_loss: 0.01430662511689525
FOLD: 0, EPOCH: 22, valid_loss: 0.01596270512789488
FOLD: 0, EPOCH: 23, train_loss: 0.013993810545526394
FOLD: 0, EPOCH: 23, valid_loss: 0.015913964211940766
FOLD: 0, EPOCH: 24, train_loss: 0.013841592149213463
FOLD: 0, EPOCH: 24, valid_loss: 0.015915647447109223
FOLD: 1, EPOCH: 0, train_loss: 0.48101349116689496
FOLD: 1, EPOCH: 0, valid_loss: 0.024172501489520073
FOLD: 1, EPOCH: 1, train_loss: 0.02123550881891429
FOLD: 1, EPOCH: 1, valid_loss: 0.01894113153219223
FOLD: 1, EPOCH: 2, train_loss: 0.019279244463561343
FOLD: 1, EPOCH: 2, valid_loss: 0.017917823791503907
FOLD: 1, EPOCH: 3, train_loss: 0.0178671865122152
FOLD: 1, EPOCH: 3, valid_loss: 0.01764489892870188
FOLD: 1, EPOCH: 4, train_loss: 0.017200770040302456
FOLD: 1, EPOCH: 4, valid_loss: 0.017184388637542725
FOLD: 1, EPOCH: 5, train_loss: 0.016991214991426792
FOLD: 1, EPOCH: 5, valid_loss: 0.016893328316509724
```

```
FOLD: 1, EPOCH: 6, train_loss: 0.017024654755685605
FOLD: 1, EPOCH: 6, valid_loss: 0.01706224299967289
FOLD: 1, EPOCH: 7, train_loss: 0.017065310160166956
FOLD: 1, EPOCH: 7, valid_loss: 0.016957631297409535
FOLD: 1, EPOCH: 8, train_loss: 0.017077514832066435
FOLD: 1, EPOCH: 8, valid_loss: 0.01694926228374243
FOLD: 1, EPOCH: 9, train_loss: 0.01712383076447208
FOLD: 1, EPOCH: 9, valid_loss: 0.016873062886297703
FOLD: 1, EPOCH: 10, train_loss: 0.017075671839724187
FOLD: 1, EPOCH: 10, valid_loss: 0.016863671243190766
FOLD: 1, EPOCH: 11, train_loss: 0.01704928425907278
FOLD: 1, EPOCH: 11, valid_loss: 0.0169156401604414
FOLD: 1, EPOCH: 12, train_loss: 0.01696545260698617
FOLD: 1, EPOCH: 12, valid_loss: 0.01661474607884884
FOLD: 1, EPOCH: 13, train_loss: 0.016893328325886303
FOLD: 1, EPOCH: 13, valid_loss: 0.016834187880158426
FOLD: 1, EPOCH: 14, train_loss: 0.016773505123699604
FOLD: 1, EPOCH: 14, valid_loss: 0.016652309224009513
FOLD: 1, EPOCH: 15, train_loss: 0.01662328158232833
FOLD: 1, EPOCH: 15, valid_loss: 0.01643828384578228
FOLD: 1, EPOCH: 16, train_loss: 0.016419106036373952
FOLD: 1, EPOCH: 16, valid_loss: 0.016398021169006825
FOLD: 1, EPOCH: 17, train_loss: 0.016195265449532845
FOLD: 1, EPOCH: 17, valid_loss: 0.016187156736850738
FOLD: 1, EPOCH: 18, train_loss: 0.015947361730140486
FOLD: 1, EPOCH: 18, valid_loss: 0.016069946698844432
FOLD: 1, EPOCH: 19, train_loss: 0.01564666821437628
FOLD: 1, EPOCH: 19, valid_loss: 0.01585985131561756
FOLD: 1, EPOCH: 20, train_loss: 0.015180368261525825
FOLD: 1, EPOCH: 20, valid_loss: 0.015835633240640162
FOLD: 1, EPOCH: 21, train_loss: 0.014794529383989419
FOLD: 1, EPOCH: 21, valid_loss: 0.015689424350857736
FOLD: 1, EPOCH: 22, train_loss: 0.01430838613384435
FOLD: 1, EPOCH: 22, valid_loss: 0.015643067844212055
FOLD: 1, EPOCH: 23, train_loss: 0.01396854356134019
FOLD: 1, EPOCH: 23, valid_loss: 0.01561105415225029
FOLD: 1, EPOCH: 24, train_loss: 0.013783435577995518
FOLD: 1, EPOCH: 24, valid_loss: 0.015604428909718991
FOLD: 2, EPOCH: 0, train_loss: 0.48014792978611526
FOLD: 2, EPOCH: 0, valid_loss: 0.02435401551425457
FOLD: 2, EPOCH: 1, train_loss: 0.02105832357136976
FOLD: 2, EPOCH: 1, valid_loss: 0.019254092983901502
FOLD: 2, EPOCH: 2, train_loss: 0.019104479349592106
FOLD: 2, EPOCH: 2, valid_loss: 0.018195971697568893
FOLD: 2, EPOCH: 3, train_loss: 0.017622472572640904
FOLD: 2, EPOCH: 3, valid_loss: 0.01733066577464342
FOLD: 2, EPOCH: 4, train_loss: 0.017050120878179056
FOLD: 2, EPOCH: 4, valid_loss: 0.01707172330468893
```

```
FOLD: 2, EPOCH: 5, train_loss: 0.01693743825288249
FOLD: 2, EPOCH: 5, valid_loss: 0.01715570665895939
FOLD: 2, EPOCH: 6, train_loss: 0.01692075591509034
FOLD: 2, EPOCH: 6, valid_loss: 0.017090896926820278
FOLD: 2, EPOCH: 7, train_loss: 0.017006877505657624
FOLD: 2, EPOCH: 7, valid_loss: 0.017335540764033795
FOLD: 2, EPOCH: 8, train_loss: 0.017080097364223733
FOLD: 2, EPOCH: 8, valid_loss: 0.017831835672259332
FOLD: 2, EPOCH: 9, train_loss: 0.01706353019998998
FOLD: 2, EPOCH: 9, valid_loss: 0.01722012512385845
FOLD: 2, EPOCH: 10, train_loss: 0.01711494774640012
FOLD: 2, EPOCH: 10, valid_loss: 0.017193745411932467
FOLD: 2, EPOCH: 11, train_loss: 0.017005059627347253
FOLD: 2, EPOCH: 11, valid_loss: 0.017023539207875728
FOLD: 2, EPOCH: 12, train_loss: 0.01696367513033606
FOLD: 2, EPOCH: 12, valid_loss: 0.01700486596673727
FOLD: 2, EPOCH: 13, train_loss: 0.016892038969969263
FOLD: 2, EPOCH: 13, valid_loss: 0.01696793921291828
FOLD: 2, EPOCH: 14, train_loss: 0.01682609088440128
FOLD: 2, EPOCH: 14, valid_loss: 0.016659756898880006
FOLD: 2, EPOCH: 15, train_loss: 0.016689405668856336
FOLD: 2, EPOCH: 15, valid_loss: 0.01668788034468889
FOLD: 2, EPOCH: 16, train_loss: 0.01647828568127893
FOLD: 2, EPOCH: 16, valid_loss: 0.016628759540617467
FOLD: 2, EPOCH: 17, train_loss: 0.016226353852369754
FOLD: 2, EPOCH: 17, valid_loss: 0.01639806181192398
FOLD: 2, EPOCH: 18, train_loss: 0.015973712520382436
FOLD: 2, EPOCH: 18, valid_loss: 0.016335922293365
FOLD: 2, EPOCH: 19, train_loss: 0.015647757259913447
FOLD: 2, EPOCH: 19, valid_loss: 0.016092955954372882
FOLD: 2, EPOCH: 20, train_loss: 0.015257250099461906
FOLD: 2, EPOCH: 20, valid_loss: 0.016060215160250663
FOLD: 2, EPOCH: 21, train_loss: 0.014875564331404205
FOLD: 2, EPOCH: 21, valid_loss: 0.015929784849286078
FOLD: 2, EPOCH: 22, train_loss: 0.014479164655009905
FOLD: 2, EPOCH: 22, valid_loss: 0.015874796137213706
FOLD: 2, EPOCH: 23, train_loss: 0.014147493329380645
FOLD: 2, EPOCH: 23, valid_loss: 0.01589205976575613
FOLD: 2, EPOCH: 24, train_loss: 0.013996494902285184
FOLD: 2, EPOCH: 24, valid_loss: 0.01587852757424116
FOLD: 3, EPOCH: 0, train_loss: 0.47945041371648817
FOLD: 3, EPOCH: 0, valid_loss: 0.02626442037522793
FOLD: 3, EPOCH: 1, train_loss: 0.021603909428833293
FOLD: 3, EPOCH: 1, valid_loss: 0.01924003578722477
FOLD: 3, EPOCH: 2, train_loss: 0.019382180284936816
FOLD: 3, EPOCH: 2, valid_loss: 0.018028894886374474
FOLD: 3, EPOCH: 3, train_loss: 0.01791812566292732
FOLD: 3, EPOCH: 3, valid_loss: 0.017468085139989854
```

```
FOLD: 3, EPOCH: 4, train_loss: 0.01714832115867714
FOLD: 3, EPOCH: 4, valid_loss: 0.01729370392858982
FOLD: 3, EPOCH: 5, train_loss: 0.016904575876942298
FOLD: 3, EPOCH: 5, valid_loss: 0.017249271012842654
FOLD: 3, EPOCH: 6, train_loss: 0.01695691045297652
FOLD: 3, EPOCH: 6, valid_loss: 0.01717962071299553
FOLD: 3, EPOCH: 7, train_loss: 0.017023298750016964
FOLD: 3, EPOCH: 7, valid_loss: 0.017589910216629506
FOLD: 3, EPOCH: 8, train_loss: 0.017007523216307163
FOLD: 3, EPOCH: 8, valid_loss: 0.017225248031318186
FOLD: 3, EPOCH: 9, train_loss: 0.017037427813118816
FOLD: 3, EPOCH: 9, valid_loss: 0.017372040599584578
FOLD: 3, EPOCH: 10, train_loss: 0.017030746074152642
FOLD: 3, EPOCH: 10, valid_loss: 0.0171218204125762
FOLD: 3, EPOCH: 11, train_loss: 0.016926696156563403
FOLD: 3, EPOCH: 11, valid_loss: 0.01715603057295084
FOLD: 3, EPOCH: 12, train_loss: 0.016941717344329876
FOLD: 3, EPOCH: 12, valid_loss: 0.017057901732623577
FOLD: 3, EPOCH: 13, train_loss: 0.016847261107292306
FOLD: 3, EPOCH: 13, valid_loss: 0.01697424430400133
FOLD: 3, EPOCH: 14, train_loss: 0.01674241254574993
FOLD: 3, EPOCH: 14, valid_loss: 0.016829471848905087
FOLD: 3, EPOCH: 15, train_loss: 0.016574264421653585
FOLD: 3, EPOCH: 15, valid_loss: 0.016728610508143903
FOLD: 3, EPOCH: 16, train_loss: 0.01631410094294824
FOLD: 3, EPOCH: 16, valid_loss: 0.016631410494446754
FOLD: 3, EPOCH: 17, train_loss: 0.01611315851182151
FOLD: 3, EPOCH: 17, valid_loss: 0.016500789038836956
FOLD: 3, EPOCH: 18, train_loss: 0.0158379802252261
FOLD: 3, EPOCH: 18, valid_loss: 0.016370026879012584
FOLD: 3, EPOCH: 19, train_loss: 0.015546030827424152
FOLD: 3, EPOCH: 19, valid_loss: 0.016286748014390468
FOLD: 3, EPOCH: 20, train_loss: 0.01514429927227043
FOLD: 3, EPOCH: 20, valid_loss: 0.016149989739060403
FOLD: 3, EPOCH: 21, train_loss: 0.014649596217335487
FOLD: 3, EPOCH: 21, valid_loss: 0.01607527744024992
FOLD: 3, EPOCH: 22, train_loss: 0.014237544131877066
FOLD: 3, EPOCH: 22, valid_loss: 0.016049507446587084
FOLD: 3, EPOCH: 23, train_loss: 0.01387515122413027
FOLD: 3, EPOCH: 23, valid_loss: 0.016010844483971595
FOLD: 3, EPOCH: 24, train_loss: 0.01370518864924405
FOLD: 3, EPOCH: 24, valid_loss: 0.016019597686827182
FOLD: 4, EPOCH: 0, train_loss: 0.47979227096146465
FOLD: 4, EPOCH: 0, valid_loss: 0.023756136819720268
FOLD: 4, EPOCH: 1, train_loss: 0.021278515318725384
FOLD: 4, EPOCH: 1, valid_loss: 0.019250506684184075
FOLD: 4, EPOCH: 2, train_loss: 0.019127000072578185
FOLD: 4, EPOCH: 2, valid_loss: 0.017930823266506194
```

```
FOLD: 4, EPOCH: 3, train_loss: 0.017730015627786415
FOLD: 4, EPOCH: 3, valid_loss: 0.01740659847855568
FOLD: 4, EPOCH: 4, train_loss: 0.017123506740242444
FOLD: 4, EPOCH: 4, valid_loss: 0.01730248212814331
FOLD: 4, EPOCH: 5, train_loss: 0.016967353398347793
FOLD: 4, EPOCH: 5, valid_loss: 0.017070715390145777
FOLD: 4, EPOCH: 6, train_loss: 0.01697713235507206
FOLD: 4, EPOCH: 6, valid_loss: 0.017035534009337425
FOLD: 4, EPOCH: 7, train_loss: 0.017001673433498867
FOLD: 4, EPOCH: 7, valid_loss: 0.017108800485730172
FOLD: 4, EPOCH: 8, train_loss: 0.01702748972060932
FOLD: 4, EPOCH: 8, valid_loss: 0.017647182792425154
FOLD: 4, EPOCH: 9, train_loss: 0.01697438621424696
FOLD: 4, EPOCH: 9, valid_loss: 0.017248767614364623
FOLD: 4, EPOCH: 10, train_loss: 0.01699783480060952
FOLD: 4, EPOCH: 10, valid_loss: 0.017167230397462846
FOLD: 4, EPOCH: 11, train_loss: 0.016993913869215112
FOLD: 4, EPOCH: 11, valid_loss: 0.017022103890776633
FOLD: 4, EPOCH: 12, train_loss: 0.016960860850910347
FOLD: 4, EPOCH: 12, valid_loss: 0.017057210318744184
FOLD: 4, EPOCH: 13, train_loss: 0.01682827326462788
FOLD: 4, EPOCH: 13, valid_loss: 0.01704358119517565
FOLD: 4, EPOCH: 14, train_loss: 0.016684481348259513
FOLD: 4, EPOCH: 14, valid_loss: 0.016859576255083084
FOLD: 4, EPOCH: 15, train_loss: 0.016617069610071426
FOLD: 4, EPOCH: 15, valid_loss: 0.016796655617654324
FOLD: 4, EPOCH: 16, train_loss: 0.016375390610017745
FOLD: 4, EPOCH: 16, valid_loss: 0.016683779545128345
FOLD: 4, EPOCH: 17, train_loss: 0.01609957825747274
FOLD: 4, EPOCH: 17, valid_loss: 0.016607813462615015
FOLD: 4, EPOCH: 18, train_loss: 0.015846912267611545
FOLD: 4, EPOCH: 18, valid_loss: 0.016459699235856533
FOLD: 4, EPOCH: 19, train_loss: 0.015585360858513385
FOLD: 4, EPOCH: 19, valid_loss: 0.016277123317122458
FOLD: 4, EPOCH: 20, train_loss: 0.01511754502928784
FOLD: 4, EPOCH: 20, valid_loss: 0.016051533706486223
FOLD: 4, EPOCH: 21, train_loss: 0.014725327491760254
FOLD: 4, EPOCH: 21, valid_loss: 0.015974657274782658
FOLD: 4, EPOCH: 22, train_loss: 0.014294504194336683
FOLD: 4, EPOCH: 22, valid_loss: 0.01594057273119688
FOLD: 4, EPOCH: 23, train_loss: 0.01398122163020632
FOLD: 4, EPOCH: 23, valid_loss: 0.01594918567687273
FOLD: 4, EPOCH: 24, train_loss: 0.013797129675441859
FOLD: 4, EPOCH: 24, valid_loss: 0.01595120184123516
FOLD: 5, EPOCH: 0, train_loss: 0.47986524075675174
FOLD: 5, EPOCH: 0, valid_loss: 0.025934466272592546
FOLD: 5, EPOCH: 1, train_loss: 0.021193325190114326
FOLD: 5, EPOCH: 1, valid_loss: 0.01959357112646103
```

```
FOLD: 5, EPOCH: 2, train_loss: 0.019030482850360627
FOLD: 5, EPOCH: 2, valid_loss: 0.017986321672797203
FOLD: 5, EPOCH: 3, train_loss: 0.017664704193063333
FOLD: 5, EPOCH: 3, valid_loss: 0.017625335454940796
FOLD: 5, EPOCH: 4, train_loss: 0.016992292715375928
FOLD: 5, EPOCH: 4, valid_loss: 0.017307802066206933
FOLD: 5, EPOCH: 5, train_loss: 0.016923305223740283
FOLD: 5, EPOCH: 5, valid_loss: 0.017240221351385115
FOLD: 5, EPOCH: 6, train_loss: 0.016879166653823285
FOLD: 5, EPOCH: 6, valid_loss: 0.01718485090881586
FOLD: 5, EPOCH: 7, train_loss: 0.016993548062180175
FOLD: 5, EPOCH: 7, valid_loss: 0.017311646454036236
FOLD: 5, EPOCH: 8, train_loss: 0.01698109854747649
FOLD: 5, EPOCH: 8, valid_loss: 0.0170509385317564
FOLD: 5, EPOCH: 9, train_loss: 0.01701391229824144
FOLD: 5, EPOCH: 9, valid_loss: 0.017161188684403895
FOLD: 5, EPOCH: 10, train_loss: 0.017091054862149718
FOLD: 5, EPOCH: 10, valid_loss: 0.017138783372938633
FOLD: 5, EPOCH: 11, train_loss: 0.017005730692778722
FOLD: 5, EPOCH: 11, valid_loss: 0.017045263014733792
FOLD: 5, EPOCH: 12, train_loss: 0.016905066721952286
FOLD: 5, EPOCH: 12, valid_loss: 0.0170972016826272
FOLD: 5, EPOCH: 13, train_loss: 0.016883914194190178
FOLD: 5, EPOCH: 13, valid_loss: 0.016796216145157816
FOLD: 6, EPOCH: 7, train_loss: 0.017086753868783007
FOLD: 6, EPOCH: 7, valid_loss: 0.01713148854672909
FOLD: 6, EPOCH: 8, train_loss: 0.017160255753365505
FOLD: 6, EPOCH: 8, valid_loss: 0.017192012891173364
FOLD: 6, EPOCH: 9, train_loss: 0.017142811695075765
FOLD: 6, EPOCH: 9, valid_loss: 0.017136567868292332
FOLD: 6, EPOCH: 10, train_loss: 0.017066879514731518
FOLD: 6, EPOCH: 10, valid_loss: 0.01709876712411642
FOLD: 6, EPOCH: 11, train_loss: 0.017008107991851106
FOLD: 6, EPOCH: 11, valid_loss: 0.016995459124445916
FOLD: 6, EPOCH: 12, train_loss: 0.016932098777825328
FOLD: 6, EPOCH: 12, valid_loss: 0.016836671344935893
FOLD: 6, EPOCH: 13, train_loss: 0.016922563504503697
FOLD: 6, EPOCH: 13, valid_loss: 0.01679877907037735
FOLD: 6, EPOCH: 14, train_loss: 0.016749000601267734
FOLD: 6, EPOCH: 14, valid_loss: 0.01672162152826786
FOLD: 6, EPOCH: 15, train_loss: 0.016618913768150775
FOLD: 6, EPOCH: 15, valid_loss: 0.016513886414468288
FOLD: 6, EPOCH: 16, train_loss: 0.016409706850187715
FOLD: 6, EPOCH: 16, valid_loss: 0.01647498246282339
FOLD: 6, EPOCH: 17, train_loss: 0.01622363109597746
FOLD: 6, EPOCH: 17, valid_loss: 0.016452667117118836
FOLD: 6, EPOCH: 18, train_loss: 0.015966843446197154
FOLD: 6, EPOCH: 18, valid_loss: 0.016250338293612002
```

```
FOLD: 6, EPOCH: 19, train_loss: 0.015590189626681156
FOLD: 6, EPOCH: 19, valid_loss: 0.016108653880655764
FOLD: 6, EPOCH: 20, train_loss: 0.015300895765322406
FOLD: 6, EPOCH: 20, valid_loss: 0.01601420532912016
FOLD: 6, EPOCH: 21, train_loss: 0.014868240917519647
FOLD: 6, EPOCH: 21, valid_loss: 0.015887106619775295
FOLD: 6, EPOCH: 22, train_loss: 0.014464293929691218
FOLD: 6, EPOCH: 22, valid_loss: 0.015840716809034348
FOLD: 6, EPOCH: 23, train_loss: 0.01414437026900499
FOLD: 6, EPOCH: 23, valid_loss: 0.015820065699517726
FOLD: 6, EPOCH: 24, train_loss: 0.01400341711887697
FOLD: 6, EPOCH: 24, valid_loss: 0.01581827063113451
FOLD: 0, EPOCH: 0, train_loss: 0.4802097340247461
FOLD: 0, EPOCH: 0, valid_loss: 0.023677160665392875
FOLD: 0, EPOCH: 1, train_loss: 0.021078440037613014
FOLD: 0, EPOCH: 1, valid_loss: 0.019565429985523224
FOLD: 0, EPOCH: 2, train_loss: 0.018793020346740477
FOLD: 0, EPOCH: 2, valid_loss: 0.018063567280769348
FOLD: 0, EPOCH: 3, train_loss: 0.01760332249611819
FOLD: 0, EPOCH: 3, valid_loss: 0.017165017649531364
FOLD: 0, EPOCH: 4, train_loss: 0.017044735622598606
FOLD: 0, EPOCH: 4, valid_loss: 0.017316286116838456
FOLD: 0, EPOCH: 5, train_loss: 0.016907186988665134
FOLD: 0, EPOCH: 5, valid_loss: 0.017147769033908845
FOLD: 0, EPOCH: 6, train_loss: 0.01693680673381504
FOLD: 0, EPOCH: 6, valid_loss: 0.017706363871693612
FOLD: 0, EPOCH: 7, train_loss: 0.017056371028325995
FOLD: 0, EPOCH: 7, valid_loss: 0.01716181229799986
FOLD: 0, EPOCH: 8, train_loss: 0.01703475930785038
FOLD: 0, EPOCH: 8, valid_loss: 0.017273162826895713
FOLD: 0, EPOCH: 9, train_loss: 0.017022194606917247
FOLD: 0, EPOCH: 9, valid_loss: 0.017177169397473335
FOLD: 0, EPOCH: 10, train_loss: 0.01706383198651732
FOLD: 0, EPOCH: 10, valid_loss: 0.017104117423295973
FOLD: 0, EPOCH: 11, train_loss: 0.016971215415669948
FOLD: 0, EPOCH: 11, valid_loss: 0.017065220028162003
FOLD: 0, EPOCH: 12, train_loss: 0.01683125430147867
FOLD: 0, EPOCH: 12, valid_loss: 0.017060539871454238
FOLD: 0, EPOCH: 13, train_loss: 0.016818800472280605
FOLD: 0, EPOCH: 13, valid_loss: 0.016872598864138125
FOLD: 0, EPOCH: 14, train_loss: 0.016613634715021468
FOLD: 0, EPOCH: 14, valid_loss: 0.016721865385770796
FOLD: 0, EPOCH: 15, train_loss: 0.01655414677700218
FOLD: 0, EPOCH: 15, valid_loss: 0.01675972983241081
FOLD: 0, EPOCH: 16, train_loss: 0.016307498284039042
FOLD: 0, EPOCH: 16, valid_loss: 0.01665621720254421
FOLD: 0, EPOCH: 17, train_loss: 0.016115811247961458
FOLD: 0, EPOCH: 17, valid_loss: 0.016450380235910417
```

```
FOLD: 0, EPOCH: 18, train_loss: 0.015851454112400003
FOLD: 0, EPOCH: 18, valid_loss: 0.016329104974865914
FOLD: 0, EPOCH: 19, train_loss: 0.015483193948459463
FOLD: 0, EPOCH: 19, valid_loss: 0.01616803154349327
FOLD: 0, EPOCH: 20, train_loss: 0.015123226287058826
FOLD: 0, EPOCH: 20, valid_loss: 0.016120055578649043
FOLD: 0, EPOCH: 21, train_loss: 0.014706688812699448
FOLD: 0, EPOCH: 21, valid_loss: 0.015983118489384652
FOLD: 0, EPOCH: 22, train_loss: 0.014356004327636998
FOLD: 0, EPOCH: 22, valid_loss: 0.01591516487300396
FOLD: 0, EPOCH: 23, train_loss: 0.013996747379400293
FOLD: 0, EPOCH: 23, valid_loss: 0.015911887288093566
FOLD: 0, EPOCH: 24, train_loss: 0.013818384496634509
FOLD: 0, EPOCH: 24, valid_loss: 0.015888899154961108
FOLD: 1, EPOCH: 0, train_loss: 0.48106997964434883
FOLD: 1, EPOCH: 0, valid_loss: 0.02381311550736427
FOLD: 1, EPOCH: 1, train_loss: 0.02100737152152321
FOLD: 1, EPOCH: 1, valid_loss: 0.01894359424710274
FOLD: 1, EPOCH: 2, train_loss: 0.018991019120629952
FOLD: 1, EPOCH: 2, valid_loss: 0.01778986819088459
FOLD: 1, EPOCH: 3, train_loss: 0.01783733694579731
FOLD: 1, EPOCH: 3, valid_loss: 0.017048313170671462
FOLD: 1, EPOCH: 4, train_loss: 0.017075920236759447
FOLD: 1, EPOCH: 4, valid_loss: 0.016944882422685624
FOLD: 1, EPOCH: 5, train_loss: 0.01697570709574993
FOLD: 1, EPOCH: 5, valid_loss: 0.016836830750107765
FOLD: 1, EPOCH: 6, train_loss: 0.017017390737373406
FOLD: 1, EPOCH: 6, valid_loss: 0.016926807127892973
FOLD: 1, EPOCH: 7, train_loss: 0.017074870857961322
FOLD: 1, EPOCH: 7, valid_loss: 0.017026931643486024
FOLD: 1, EPOCH: 8, train_loss: 0.017133073380761813
FOLD: 1, EPOCH: 8, valid_loss: 0.01696266293525696
FOLD: 1, EPOCH: 9, train_loss: 0.017168587871960232
FOLD: 1, EPOCH: 9, valid_loss: 0.016913207545876503
FOLD: 1, EPOCH: 10, train_loss: 0.01701185171853523
FOLD: 1, EPOCH: 10, valid_loss: 0.016885814629495143
FOLD: 1, EPOCH: 11, train_loss: 0.017092899394025203
FOLD: 1, EPOCH: 11, valid_loss: 0.01721460271626711
FOLD: 1, EPOCH: 12, train_loss: 0.01703449873393085
FOLD: 1, EPOCH: 12, valid_loss: 0.016740203388035297
FOLD: 1, EPOCH: 13, train_loss: 0.016931443316798633
FOLD: 1, EPOCH: 13, valid_loss: 0.01686215043067932
FOLD: 1, EPOCH: 14, train_loss: 0.016834129522345503
FOLD: 1, EPOCH: 14, valid_loss: 0.016647883541882038
FOLD: 1, EPOCH: 15, train_loss: 0.01662638644706838
FOLD: 1, EPOCH: 15, valid_loss: 0.01666332546621561
FOLD: 1, EPOCH: 16, train_loss: 0.01644561321911763
FOLD: 1, EPOCH: 16, valid_loss: 0.01635868340730667
```

```
FOLD: 1, EPOCH: 17, train_loss: 0.016246650678416092
FOLD: 1, EPOCH: 17, valid_loss: 0.01627295434474945
FOLD: 1, EPOCH: 18, train_loss: 0.015977449208295264
FOLD: 1, EPOCH: 18, valid_loss: 0.01607346288859844
FOLD: 1, EPOCH: 19, train_loss: 0.015626401038599663
FOLD: 1, EPOCH: 19, valid_loss: 0.01592955969274044
FOLD: 1, EPOCH: 20, train_loss: 0.015255805770201343
FOLD: 1, EPOCH: 20, valid_loss: 0.015791633687913417
FOLD: 1, EPOCH: 21, train_loss: 0.014832297935575044
FOLD: 1, EPOCH: 21, valid_loss: 0.015695342496037483
FOLD: 1, EPOCH: 22, train_loss: 0.014482152711625408
FOLD: 1, EPOCH: 22, valid_loss: 0.015674432925879955
FOLD: 1, EPOCH: 23, train_loss: 0.014157214734171118
FOLD: 1, EPOCH: 23, valid_loss: 0.015648740381002425
FOLD: 1, EPOCH: 24, train_loss: 0.013973459797803641
FOLD: 1, EPOCH: 24, valid_loss: 0.01561506874859333
FOLD: 2, EPOCH: 0, train_loss: 0.4819955713793534
FOLD: 2, EPOCH: 0, valid_loss: 0.023524328619241714
FOLD: 2, EPOCH: 1, train_loss: 0.021199215799063243
FOLD: 2, EPOCH: 1, valid_loss: 0.019142170920968055
FOLD: 2, EPOCH: 2, train_loss: 0.018955170487364132
FOLD: 2, EPOCH: 2, valid_loss: 0.018457992933690548
FOLD: 2, EPOCH: 3, train_loss: 0.017673212990519546
FOLD: 2, EPOCH: 3, valid_loss: 0.01740675576031208
FOLD: 2, EPOCH: 4, train_loss: 0.01712116459067784
FOLD: 2, EPOCH: 4, valid_loss: 0.017029922045767306
FOLD: 2, EPOCH: 5, train_loss: 0.016862711121587933
FOLD: 2, EPOCH: 5, valid_loss: 0.01699919868260622
FOLD: 2, EPOCH: 6, train_loss: 0.01696647821488429
FOLD: 2, EPOCH: 6, valid_loss: 0.01706337958574295
FOLD: 2, EPOCH: 7, train_loss: 0.01709343477779505
FOLD: 2, EPOCH: 7, valid_loss: 0.01710910752415657
FOLD: 2, EPOCH: 8, train_loss: 0.01707884697180216
FOLD: 2, EPOCH: 8, valid_loss: 0.01704868797212839
FOLD: 2, EPOCH: 9, train_loss: 0.017021298890008408
FOLD: 2, EPOCH: 9, valid_loss: 0.017110474519431592
FOLD: 2, EPOCH: 10, train_loss: 0.01701112442529526
FOLD: 2, EPOCH: 10, valid_loss: 0.0169909892231226
FOLD: 2, EPOCH: 11, train_loss: 0.017000524080073347
FOLD: 2, EPOCH: 11, valid_loss: 0.01714823465794325
FOLD: 2, EPOCH: 12, train_loss: 0.01691506884130491
FOLD: 2, EPOCH: 12, valid_loss: 0.01684942726045847
FOLD: 2, EPOCH: 13, train_loss: 0.01680523764696859
FOLD: 2, EPOCH: 13, valid_loss: 0.01674946613609791
FOLD: 2, EPOCH: 14, train_loss: 0.01668912193858299
FOLD: 2, EPOCH: 14, valid_loss: 0.016844186931848526
FOLD: 2, EPOCH: 15, train_loss: 0.016561927552632735
FOLD: 2, EPOCH: 15, valid_loss: 0.016663097478449344
```

```
FOLD: 2, EPOCH: 16, train_loss: 0.016363194288343798
FOLD: 2, EPOCH: 16, valid_loss: 0.016588844023644925
FOLD: 2, EPOCH: 17, train_loss: 0.016161876825653777
FOLD: 2, EPOCH: 17, valid_loss: 0.016398345716297627
FOLD: 2, EPOCH: 18, train_loss: 0.01595033498696324
FOLD: 2, EPOCH: 18, valid_loss: 0.01629351604729891
FOLD: 2, EPOCH: 19, train_loss: 0.015550773818882145
FOLD: 2, EPOCH: 19, valid_loss: 0.016110017597675323
FOLD: 2, EPOCH: 20, train_loss: 0.01516550410614938
FOLD: 2, EPOCH: 20, valid_loss: 0.015996734388172628
FOLD: 2, EPOCH: 21, train_loss: 0.014722905406842426
FOLD: 2, EPOCH: 21, valid_loss: 0.01592323534190655
FOLD: 2, EPOCH: 22, train_loss: 0.014281904217185212
FOLD: 2, EPOCH: 22, valid_loss: 0.015870127454400064
FOLD: 2, EPOCH: 23, train_loss: 0.013950388720195715
FOLD: 2, EPOCH: 23, valid_loss: 0.015868438445031642
FOLD: 2, EPOCH: 24, train_loss: 0.013757832886866566
FOLD: 2, EPOCH: 24, valid_loss: 0.015854530818760395
FOLD: 3, EPOCH: 0, train_loss: 0.4814249072984165
FOLD: 3, EPOCH: 0, valid_loss: 0.02484494395554066
FOLD: 3, EPOCH: 1, train_loss: 0.021415635423899508
FOLD: 3, EPOCH: 1, valid_loss: 0.020806695595383645
FOLD: 3, EPOCH: 2, train_loss: 0.019223664393078307
FOLD: 3, EPOCH: 2, valid_loss: 0.01799988031387329
FOLD: 3, EPOCH: 3, train_loss: 0.01758199206040222
FOLD: 3, EPOCH: 3, valid_loss: 0.017245096415281297
FOLD: 3, EPOCH: 4, train_loss: 0.017021527691256432
FOLD: 3, EPOCH: 4, valid_loss: 0.017250209972262384
FOLD: 3, EPOCH: 5, train_loss: 0.01687715928621438
FOLD: 3, EPOCH: 5, valid_loss: 0.017191191613674165
FOLD: 3, EPOCH: 6, train_loss: 0.01687027285268315
FOLD: 3, EPOCH: 6, valid_loss: 0.017321761399507522
FOLD: 3, EPOCH: 7, train_loss: 0.016986550832940202
FOLD: 3, EPOCH: 7, valid_loss: 0.01711368203163147
FOLD: 3, EPOCH: 8, train_loss: 0.01697341636532829
FOLD: 3, EPOCH: 8, valid_loss: 0.017397280633449554
FOLD: 3, EPOCH: 9, train_loss: 0.016995796521960878
FOLD: 3, EPOCH: 9, valid_loss: 0.01716959122568369
FOLD: 3, EPOCH: 10, train_loss: 0.01690251320017641
FOLD: 3, EPOCH: 10, valid_loss: 0.017044662162661553
FOLD: 3, EPOCH: 11, train_loss: 0.01690786553001931
FOLD: 3, EPOCH: 11, valid_loss: 0.016988007910549642
FOLD: 3, EPOCH: 12, train_loss: 0.01686315806772636
FOLD: 3, EPOCH: 12, valid_loss: 0.01696857627481222
FOLD: 3, EPOCH: 13, train_loss: 0.01678199317230254
FOLD: 3, EPOCH: 13, valid_loss: 0.01689022798091173
FOLD: 3, EPOCH: 14, train_loss: 0.01665234383905218
FOLD: 3, EPOCH: 14, valid_loss: 0.016767617911100388
```

```
FOLD: 3, EPOCH: 15, train_loss: 0.016493407952390154
FOLD: 3, EPOCH: 15, valid_loss: 0.016756099052727222
FOLD: 3, EPOCH: 16, train_loss: 0.016360641260739085
FOLD: 3, EPOCH: 16, valid_loss: 0.01658147256821394
FOLD: 3, EPOCH: 17, train_loss: 0.016071100351812483
FOLD: 3, EPOCH: 17, valid_loss: 0.016557400301098823
FOLD: 3, EPOCH: 18, train_loss: 0.01579407758402581
FOLD: 3, EPOCH: 18, valid_loss: 0.016480549164116384
FOLD: 3, EPOCH: 19, train_loss: 0.015481043131831957
FOLD: 3, EPOCH: 19, valid_loss: 0.016160260029137133
FOLD: 3, EPOCH: 20, train_loss: 0.01508077405722571
FOLD: 3, EPOCH: 20, valid_loss: 0.016125589348375796
FOLD: 3, EPOCH: 21, train_loss: 0.01465328445746785
FOLD: 3, EPOCH: 21, valid_loss: 0.01603327341377735
FOLD: 3, EPOCH: 22, train_loss: 0.014198744846951394
FOLD: 3, EPOCH: 22, valid_loss: 0.015956933572888375
FOLD: 3, EPOCH: 23, train_loss: 0.013809717919195996
FOLD: 3, EPOCH: 23, valid_loss: 0.015958831161260605
FOLD: 3, EPOCH: 24, train_loss: 0.013649029650909155
FOLD: 3, EPOCH: 24, valid_loss: 0.015935721658170223
FOLD: 4, EPOCH: 0, train_loss: 0.4793169985772396
FOLD: 4, EPOCH: 0, valid_loss: 0.02354385010898113
FOLD: 4, EPOCH: 1, train_loss: 0.021169569127920534
FOLD: 4, EPOCH: 1, valid_loss: 0.01926589161157608
FOLD: 4, EPOCH: 2, train_loss: 0.018858174882119612
FOLD: 4, EPOCH: 2, valid_loss: 0.018729323893785475
FOLD: 4, EPOCH: 3, train_loss: 0.017648787529138076
FOLD: 4, EPOCH: 3, valid_loss: 0.01769738256931305
FOLD: 4, EPOCH: 4, train_loss: 0.017052796827692562
FOLD: 4, EPOCH: 4, valid_loss: 0.0176533280313015
FOLD: 4, EPOCH: 5, train_loss: 0.01691915518699252
FOLD: 4, EPOCH: 5, valid_loss: 0.017138376533985138
FOLD: 4, EPOCH: 6, train_loss: 0.016950514750415775
FOLD: 4, EPOCH: 6, valid_loss: 0.01711412876844406
FOLD: 4, EPOCH: 7, train_loss: 0.016980466040066717
FOLD: 4, EPOCH: 7, valid_loss: 0.01729469805955887
FOLD: 4, EPOCH: 8, train_loss: 0.017002722983356235
FOLD: 4, EPOCH: 8, valid_loss: 0.017412457875907422
FOLD: 4, EPOCH: 9, train_loss: 0.01700050390775309
FOLD: 4, EPOCH: 9, valid_loss: 0.017324964739382265
FOLD: 4, EPOCH: 10, train_loss: 0.017034049716074855
FOLD: 4, EPOCH: 10, valid_loss: 0.017245047464966773
FOLD: 4, EPOCH: 11, train_loss: 0.016954440902285026
FOLD: 4, EPOCH: 11, valid_loss: 0.017108303867280482
FOLD: 4, EPOCH: 12, train_loss: 0.01691856574216465
FOLD: 4, EPOCH: 12, valid_loss: 0.017044668532907963
FOLD: 4, EPOCH: 13, train_loss: 0.016791813671082057
FOLD: 4, EPOCH: 13, valid_loss: 0.017005215026438238
```

```
FOLD: 4, EPOCH: 14, train_loss: 0.016715758537151376
FOLD: 4, EPOCH: 14, valid_loss: 0.016962838508188725
FOLD: 4, EPOCH: 15, train_loss: 0.016533542171019277
FOLD: 4, EPOCH: 15, valid_loss: 0.016878145411610604
FOLD: 4, EPOCH: 16, train_loss: 0.01642287426887929
FOLD: 4, EPOCH: 16, valid_loss: 0.01669918976724148
FOLD: 4, EPOCH: 17, train_loss: 0.016158469110017733
FOLD: 4, EPOCH: 17, valid_loss: 0.016566284149885178
FOLD: 4, EPOCH: 18, train_loss: 0.01590813243394198
FOLD: 4, EPOCH: 18, valid_loss: 0.01645063243806362
FOLD: 4, EPOCH: 19, train_loss: 0.015547146133723714
FOLD: 4, EPOCH: 19, valid_loss: 0.01635195080190897
FOLD: 4, EPOCH: 20, train_loss: 0.015188460618409574
FOLD: 4, EPOCH: 20, valid_loss: 0.01612867407500744
FOLD: 4, EPOCH: 21, train_loss: 0.014767855494504883
FOLD: 4, EPOCH: 21, valid_loss: 0.016061410531401633
FOLD: 4, EPOCH: 22, train_loss: 0.014309588680360592
FOLD: 4, EPOCH: 22, valid_loss: 0.01602459490299225
FOLD: 4, EPOCH: 23, train_loss: 0.013992417755783821
FOLD: 4, EPOCH: 23, valid_loss: 0.016020320020616055
FOLD: 4, EPOCH: 24, train_loss: 0.013826004546262375
FOLD: 4, EPOCH: 24, valid_loss: 0.01600763350725174
FOLD: 5, EPOCH: 0, train_loss: 0.4817004766391248
FOLD: 5, EPOCH: 0, valid_loss: 0.02398435711860657
FOLD: 5, EPOCH: 1, train_loss: 0.02125679071815241
FOLD: 5, EPOCH: 1, valid_loss: 0.019251561388373375
FOLD: 5, EPOCH: 2, train_loss: 0.018961629285445425
FOLD: 5, EPOCH: 2, valid_loss: 0.01817365311086178
FOLD: 5, EPOCH: 3, train_loss: 0.01769971621989393
FOLD: 5, EPOCH: 3, valid_loss: 0.017467700727283954
FOLD: 5, EPOCH: 4, train_loss: 0.017010475040039642
FOLD: 5, EPOCH: 4, valid_loss: 0.017036359310150146
FOLD: 5, EPOCH: 5, train_loss: 0.016845518811827616
FOLD: 5, EPOCH: 5, valid_loss: 0.017098789624869822
FOLD: 5, EPOCH: 6, train_loss: 0.016879762149080126
FOLD: 5, EPOCH: 6, valid_loss: 0.01709855604916811
FOLD: 5, EPOCH: 7, train_loss: 0.017002106245074953
FOLD: 5, EPOCH: 7, valid_loss: 0.017300400510430337
FOLD: 5, EPOCH: 8, train_loss: 0.017020800087575606
FOLD: 5, EPOCH: 8, valid_loss: 0.0172310383990407
FOLD: 5, EPOCH: 9, train_loss: 0.01703525928869134
FOLD: 5, EPOCH: 9, valid_loss: 0.01736486092209816
FOLD: 5, EPOCH: 10, train_loss: 0.016998742878l3294
FOLD: 5, EPOCH: 10, valid_loss: 0.017137709781527520
FOLD: 5, EPOCH: 11, train_loss: 0.01696723349848572
FOLD: 5, EPOCH: 11, valid_loss: 0.017184167392551897
FOLD: 5, EPOCH: 12, train_loss: 0.016851482874884898
FOLD: 5, EPOCH: 12, valid_loss: 0.017023720778524876
```

```
FOLD: 5, EPOCH: 13, train_loss: 0.016821888186747118
FOLD: 5, EPOCH: 13, valid_loss: 0.016822215653955936
FOLD: 5, EPOCH: 14, train_loss: 0.016700780576690524
FOLD: 5, EPOCH: 14, valid_loss: 0.017011796571314335
FOLD: 5, EPOCH: 15, train_loss: 0.016646177755022536
FOLD: 5, EPOCH: 15, valid_loss: 0.016803045645356177
FOLD: 5, EPOCH: 16, train_loss: 0.01637126267792619
FOLD: 5, EPOCH: 16, valid_loss: 0.01667063307017088
FOLD: 5, EPOCH: 17, train_loss: 0.01618949746472292
FOLD: 5, EPOCH: 17, valid_loss: 0.016541803888976574
FOLD: 5, EPOCH: 18, train_loss: 0.015890630310540704
FOLD: 5, EPOCH: 18, valid_loss: 0.016334000527858734
FOLD: 5, EPOCH: 19, train_loss: 0.015577130551849092
FOLD: 5, EPOCH: 19, valid_loss: 0.016334536485373974
FOLD: 5, EPOCH: 20, train_loss: 0.015155713403356724
FOLD: 5, EPOCH: 20, valid_loss: 0.016105270460247994
FOLD: 5, EPOCH: 21, train_loss: 0.014757506371963592
FOLD: 5, EPOCH: 21, valid_loss: 0.016037894412875175
FOLD: 5, EPOCH: 22, train_loss: 0.01433591518018927
FOLD: 5, EPOCH: 22, valid_loss: 0.01594856910407543
FOLD: 5, EPOCH: 23, train_loss: 0.014011941267316844
FOLD: 5, EPOCH: 23, valid_loss: 0.015924684405326843
FOLD: 5, EPOCH: 24, train_loss: 0.013854472720552059
FOLD: 5, EPOCH: 24, valid_loss: 0.01590875320136547
FOLD: 6, EPOCH: 0, train_loss: 0.48087447163249764
FOLD: 6, EPOCH: 0, valid_loss: 0.02451790787279606
FOLD: 6, EPOCH: 1, train_loss: 0.02110087958981796
FOLD: 6, EPOCH: 1, valid_loss: 0.018993667103350164
FOLD: 6, EPOCH: 2, train_loss: 0.018932627501333652
FOLD: 6, EPOCH: 2, valid_loss: 0.017792280092835427
FOLD: 6, EPOCH: 3, train_loss: 0.01770821496920318
FOLD: 6, EPOCH: 3, valid_loss: 0.017682399600744247
FOLD: 6, EPOCH: 4, train_loss: 0.017116266524507887
FOLD: 6, EPOCH: 4, valid_loss: 0.017043814845383167
FOLD: 6, EPOCH: 5, train_loss: 0.01697496433115127
FOLD: 6, EPOCH: 5, valid_loss: 0.017222914844751358
FOLD: 6, EPOCH: 6, train_loss: 0.016955534661454812
FOLD: 6, EPOCH: 6, valid_loss: 0.01707905303686857
FOLD: 6, EPOCH: 7, train_loss: 0.017023652861667734
FOLD: 6, EPOCH: 7, valid_loss: 0.017122738286852835
FOLD: 6, EPOCH: 8, train_loss: 0.017074261690635664
FOLD: 6, EPOCH: 8, valid_loss: 0.017073553316295148
FOLD: 6, EPOCH: 9, train_loss: 0.017051597600992844
FOLD: 6, EPOCH: 9, valid_loss: 0.01701377496123314
FOLD: 6, EPOCH: 10, train_loss: 0.017065421912540384
FOLD: 6, EPOCH: 10, valid_loss: 0.01692074127495289
FOLD: 6, EPOCH: 11, train_loss: 0.01700886762796008
FOLD: 6, EPOCH: 11, valid_loss: 0.016885280907154084
```

```
FOLD: 6, EPOCH: 12, train_loss: 0.01693743449591455
FOLD: 6, EPOCH: 12, valid_loss: 0.016796109490096568
FOLD: 6, EPOCH: 13, train_loss: 0.016891283711709944
FOLD: 6, EPOCH: 13, valid_loss: 0.01683646850287914
FOLD: 6, EPOCH: 14, train_loss: 0.01666713138522745
FOLD: 6, EPOCH: 14, valid_loss: 0.01665860805660486
FOLD: 6, EPOCH: 15, train_loss: 0.01656966297855588
FOLD: 6, EPOCH: 15, valid_loss: 0.016589929424226282
FOLD: 6, EPOCH: 16, train_loss: 0.016391772504414426
FOLD: 6, EPOCH: 16, valid_loss: 0.016467161141335965
FOLD: 6, EPOCH: 17, train_loss: 0.01614115638088207
FOLD: 6, EPOCH: 17, valid_loss: 0.01629216264933348
FOLD: 6, EPOCH: 18, train_loss: 0.015884914308735707
FOLD: 6, EPOCH: 18, valid_loss: 0.016212784387171268
FOLD: 6, EPOCH: 19, train_loss: 0.015580061259267686
FOLD: 6, EPOCH: 19, valid_loss: 0.01610348969697952
FOLD: 6, EPOCH: 20, train_loss: 0.015166329828147985
FOLD: 6, EPOCH: 20, valid_loss: 0.015990643538534643
FOLD: 6, EPOCH: 21, train_loss: 0.014734883089454807
FOLD: 6, EPOCH: 21, valid_loss: 0.01593152154237032
FOLD: 6, EPOCH: 22, train_loss: 0.014351249894215948
FOLD: 6, EPOCH: 22, valid_loss: 0.01585207596421242
FOLD: 6, EPOCH: 23, train_loss: 0.01400099634541338
FOLD: 6, EPOCH: 23, valid_loss: 0.01584040280431509
FOLD: 6, EPOCH: 24, train_loss: 0.013842605865326058
FOLD: 6, EPOCH: 24, valid_loss: 0.01585599847137928
FOLD: 0, EPOCH: 0, train_loss: 0.4819419159725004
FOLD: 0, EPOCH: 0, valid_loss: 0.029000962823629378
FOLD: 0, EPOCH: 1, train_loss: 0.021385736243964053
FOLD: 0, EPOCH: 1, valid_loss: 0.019841325730085374
FOLD: 0, EPOCH: 2, train_loss: 0.018879663553975877
FOLD: 0, EPOCH: 2, valid_loss: 0.01818448305130005
FOLD: 0, EPOCH: 3, train_loss: 0.017701370204437752
FOLD: 0, EPOCH: 3, valid_loss: 0.017391722984611988
FOLD: 0, EPOCH: 4, train_loss: 0.017063813816223825
FOLD: 0, EPOCH: 4, valid_loss: 0.017066823914647103
FOLD: 0, EPOCH: 5, train_loss: 0.01689318541538756
FOLD: 0, EPOCH: 5, valid_loss: 0.017039356790482997
FOLD: 0, EPOCH: 6, train_loss: 0.016870030658129528
FOLD: 0, EPOCH: 6, valid_loss: 0.01706131622195244
FOLD: 0, EPOCH: 7, train_loss: 0.01700374434011526
FOLD: 0, EPOCH: 7, valid_loss: 0.017440717965364456
FOLD: 0, EPOCH: 8, train_loss: 0.017037532780142056
FOLD: 0, EPOCH: 8, valid_loss: 0.017364897578954697
FOLD: 0, EPOCH: 9, train_loss: 0.016992973404473997
FOLD: 0, EPOCH: 9, valid_loss: 0.01725954093039036
FOLD: 0, EPOCH: 10, train_loss: 0.01701850648082438
FOLD: 0, EPOCH: 10, valid_loss: 0.017373647056519986
```

```
FOLD: 0, EPOCH: 11, train_loss: 0.016997553268764294
FOLD: 0, EPOCH: 11, valid_loss: 0.017055438943207265
FOLD: 0, EPOCH: 12, train_loss: 0.016837483715443385
FOLD: 0, EPOCH: 12, valid_loss: 0.01716767031699419
FOLD: 0, EPOCH: 13, train_loss: 0.016833168973981523
FOLD: 0, EPOCH: 13, valid_loss: 0.017229015342891216
FOLD: 0, EPOCH: 14, train_loss: 0.016718744712115145
FOLD: 0, EPOCH: 14, valid_loss: 0.016829838044941425
FOLD: 0, EPOCH: 15, train_loss: 0.016576102611665824
FOLD: 0, EPOCH: 15, valid_loss: 0.016840714290738107
FOLD: 0, EPOCH: 16, train_loss: 0.016392288514140513
FOLD: 0, EPOCH: 16, valid_loss: 0.016550447754561902
FOLD: 0, EPOCH: 17, train_loss: 0.01614955798754481
FOLD: 0, EPOCH: 17, valid_loss: 0.016486600413918494
FOLD: 0, EPOCH: 18, train_loss: 0.015940942947252266
FOLD: 0, EPOCH: 18, valid_loss: 0.016269739381968976
FOLD: 0, EPOCH: 19, train_loss: 0.01557785538056878
FOLD: 0, EPOCH: 19, valid_loss: 0.01618572633713484
FOLD: 0, EPOCH: 20, train_loss: 0.015184191283674873
FOLD: 0, EPOCH: 20, valid_loss: 0.01607711985707283
FOLD: 0, EPOCH: 21, train_loss: 0.014837143125430661
FOLD: 0, EPOCH: 21, valid_loss: 0.016007880158722402
FOLD: 0, EPOCH: 22, train_loss: 0.014388534018681162
FOLD: 0, EPOCH: 22, valid_loss: 0.01593201119452715
FOLD: 0, EPOCH: 23, train_loss: 0.014069549017110649
FOLD: 0, EPOCH: 23, valid_loss: 0.01593382891267538
FOLD: 0, EPOCH: 24, train_loss: 0.013922515749728599
FOLD: 0, EPOCH: 24, valid_loss: 0.015925769172608854
FOLD: 1, EPOCH: 0, train_loss: 0.47959852935809666
FOLD: 1, EPOCH: 0, valid_loss: 0.023799810856580735
FOLD: 1, EPOCH: 1, train_loss: 0.021453201758111416
FOLD: 1, EPOCH: 1, valid_loss: 0.019690792337059976
FOLD: 1, EPOCH: 2, train_loss: 0.019037284330800684
FOLD: 1, EPOCH: 2, valid_loss: 0.01770801853388548
FOLD: 1, EPOCH: 3, train_loss: 0.01771230534428642
FOLD: 1, EPOCH: 3, valid_loss: 0.017455182895064356
FOLD: 1, EPOCH: 4, train_loss: 0.017094032629868205
FOLD: 1, EPOCH: 4, valid_loss: 0.017003441825509072
FOLD: 1, EPOCH: 5, train_loss: 0.016931396262834266
FOLD: 1, EPOCH: 5, valid_loss: 0.017131132483482362
FOLD: 1, EPOCH: 6, train_loss: 0.016966244269187757
FOLD: 1, EPOCH: 6, valid_loss: 0.01706491731107235
FOLD: 1, EPOCH: 7, train_loss: 0.01701424516686777
FOLD: 1, EPOCH: 7, valid_loss: 0.017207779474556446
FOLD: 1, EPOCH: 8, train_loss: 0.017049669579533088
FOLD: 1, EPOCH: 8, valid_loss: 0.01701597511768341
FOLD: 1, EPOCH: 9, train_loss: 0.017023839335255073
FOLD: 1, EPOCH: 9, valid_loss: 0.017088487073779107
```

```
FOLD: 1, EPOCH: 10, train_loss: 0.017090107339630727
FOLD: 1, EPOCH: 10, valid_loss: 0.016790531538426877
FOLD: 1, EPOCH: 11, train_loss: 0.017023369023690418
FOLD: 1, EPOCH: 11, valid_loss: 0.016850082613527776
FOLD: 1, EPOCH: 12, train_loss: 0.016925270003931864
FOLD: 1, EPOCH: 12, valid_loss: 0.016826210096478462
FOLD: 1, EPOCH: 13, train_loss: 0.016880371341747896
FOLD: 1, EPOCH: 13, valid_loss: 0.016634033545851707
FOLD: 1, EPOCH: 14, train_loss: 0.016705921439289236
FOLD: 1, EPOCH: 14, valid_loss: 0.01652762420475483
FOLD: 1, EPOCH: 15, train_loss: 0.016568427506302084
FOLD: 1, EPOCH: 15, valid_loss: 0.01653550911694765
FOLD: 1, EPOCH: 16, train_loss: 0.01642173574920617
FOLD: 1, EPOCH: 16, valid_loss: 0.016428660564124585
FOLD: 1, EPOCH: 17, train_loss: 0.016209890236001032
FOLD: 1, EPOCH: 17, valid_loss: 0.016250686496496202
FOLD: 1, EPOCH: 18, train_loss: 0.015970593121941802
FOLD: 1, EPOCH: 18, valid_loss: 0.016044013425707816
FOLD: 1, EPOCH: 19, train_loss: 0.01560809224413163
FOLD: 1, EPOCH: 19, valid_loss: 0.015912293195724486
FOLD: 1, EPOCH: 20, train_loss: 0.015226067924479237
FOLD: 1, EPOCH: 20, valid_loss: 0.015808549001812933
FOLD: 1, EPOCH: 21, train_loss: 0.014830686443415628
FOLD: 1, EPOCH: 21, valid_loss: 0.015722994096577167
FOLD: 1, EPOCH: 22, train_loss: 0.01438598065529348
FOLD: 1, EPOCH: 22, valid_loss: 0.015655721239745617
FOLD: 1, EPOCH: 23, train_loss: 0.014067835022448277
FOLD: 1, EPOCH: 23, valid_loss: 0.015638745799660684
FOLD: 1, EPOCH: 24, train_loss: 0.013878527205107974
FOLD: 1, EPOCH: 24, valid_loss: 0.01563992228358984
FOLD: 2, EPOCH: 0, train_loss: 0.48028866315678675
FOLD: 2, EPOCH: 0, valid_loss: 0.02396588198840618
FOLD: 2, EPOCH: 1, train_loss: 0.021166407813628513
FOLD: 2, EPOCH: 1, valid_loss: 0.019102503024041654
FOLD: 2, EPOCH: 2, train_loss: 0.018869745057253612
FOLD: 2, EPOCH: 2, valid_loss: 0.017736199051141738
FOLD: 2, EPOCH: 3, train_loss: 0.01764768829607234
FOLD: 2, EPOCH: 3, valid_loss: 0.017304112426936625
FOLD: 2, EPOCH: 4, train_loss: 0.017051171853530164
FOLD: 2, EPOCH: 4, valid_loss: 0.017034983821213247
FOLD: 2, EPOCH: 5, train_loss: 0.016842509562871893
FOLD: 2, EPOCH: 5, valid_loss: 0.01716715272516012
FOLD: 2, EPOCH: 6, train_loss: 0.016891199550559732
FOLD: 2, EPOCH: 6, valid_loss: 0.017157419547438622
FOLD: 2, EPOCH: 7, train_loss: 0.016971624209260454
FOLD: 2, EPOCH: 7, valid_loss: 0.0174099225923419
FOLD: 2, EPOCH: 8, train_loss: 0.017076523537386437
FOLD: 2, EPOCH: 8, valid_loss: 0.017136855609714986
```

```
FOLD: 2, EPOCH: 9, train_loss: 0.01701264790430361
FOLD: 2, EPOCH: 9, valid_loss: 0.017091013602912427
FOLD: 2, EPOCH: 10, train_loss: 0.01697970200076598
FOLD: 2, EPOCH: 10, valid_loss: 0.01714639026671648
FOLD: 2, EPOCH: 11, train_loss: 0.01692813931375134
FOLD: 2, EPOCH: 11, valid_loss: 0.016945201344788073
FOLD: 2, EPOCH: 12, train_loss: 0.01691362883287425
FOLD: 2, EPOCH: 12, valid_loss: 0.016982906498014927
FOLD: 2, EPOCH: 13, train_loss: 0.016854414830402453
FOLD: 2, EPOCH: 13, valid_loss: 0.017027838826179503
FOLD: 2, EPOCH: 14, train_loss: 0.016720483142907926
FOLD: 2, EPOCH: 14, valid_loss: 0.01674292128533125
FOLD: 2, EPOCH: 15, train_loss: 0.016571240791049945
FOLD: 2, EPOCH: 15, valid_loss: 0.016858424879610538
FOLD: 2, EPOCH: 16, train_loss: 0.016394051057951792
FOLD: 2, EPOCH: 16, valid_loss: 0.016361519545316696
FOLD: 2, EPOCH: 17, train_loss: 0.01612644449991434
FOLD: 2, EPOCH: 17, valid_loss: 0.01651067692786455
FOLD: 2, EPOCH: 18, train_loss: 0.015905844769915755
FOLD: 2, EPOCH: 18, valid_loss: 0.016277469396591186
FOLD: 2, EPOCH: 19, train_loss: 0.015612077810579822
FOLD: 2, EPOCH: 19, valid_loss: 0.016163806468248366
FOLD: 2, EPOCH: 20, train_loss: 0.01520488096312398
FOLD: 2, EPOCH: 20, valid_loss: 0.016008062288165092
FOLD: 2, EPOCH: 21, train_loss: 0.014767379690037698
FOLD: 2, EPOCH: 21, valid_loss: 0.01589284084737301
FOLD: 2, EPOCH: 22, train_loss: 0.014368713624319253
FOLD: 2, EPOCH: 22, valid_loss: 0.015871377401053904
FOLD: 2, EPOCH: 23, train_loss: 0.014046999651203755
FOLD: 2, EPOCH: 23, valid_loss: 0.015850137658417224
FOLD: 2, EPOCH: 24, train_loss: 0.013908739265079807
FOLD: 2, EPOCH: 24, valid_loss: 0.015869730412960054
FOLD: 3, EPOCH: 0, train_loss: 0.48089370620595356
FOLD: 3, EPOCH: 0, valid_loss: 0.02451129544878006
FOLD: 3, EPOCH: 1, train_loss: 0.02132698240987703
FOLD: 3, EPOCH: 1, valid_loss: 0.019514541029930114
FOLD: 3, EPOCH: 2, train_loss: 0.01892708204559931
FOLD: 3, EPOCH: 2, valid_loss: 0.01798509865999222
FOLD: 3, EPOCH: 3, train_loss: 0.01769003870763949
FOLD: 3, EPOCH: 3, valid_loss: 0.017789019122719765
FOLD: 3, EPOCH: 4, train_loss: 0.017002633120231078
FOLD: 3, EPOCH: 4, valid_loss: 0.01731603778898716
FOLD: 3, EPOCH: 5, train_loss: 0.016890021642910785
FOLD: 3, EPOCH: 5, valid_loss: 0.01730370569974184
FOLD: 3, EPOCH: 6, train_loss: 0.016908648797646673
FOLD: 3, EPOCH: 6, valid_loss: 0.017231047414243222
FOLD: 3, EPOCH: 7, train_loss: 0.016987193147746885
FOLD: 3, EPOCH: 7, valid_loss: 0.017173033282160757
```

```
FOLD: 3, EPOCH: 8, train_loss: 0.016991135334836788
FOLD: 3, EPOCH: 8, valid_loss: 0.017132566347718238
FOLD: 3, EPOCH: 9, train_loss: 0.01698526016556892
FOLD: 3, EPOCH: 9, valid_loss: 0.017188773714005946
FOLD: 3, EPOCH: 10, train_loss: 0.016971099120723147
FOLD: 3, EPOCH: 10, valid_loss: 0.017182458266615866
FOLD: 3, EPOCH: 11, train_loss: 0.016966379070211025
FOLD: 3, EPOCH: 11, valid_loss: 0.017092336788773536
FOLD: 3, EPOCH: 12, train_loss: 0.016852155714264128
FOLD: 3, EPOCH: 12, valid_loss: 0.016948508135974406
FOLD: 3, EPOCH: 13, train_loss: 0.01677230680298035
FOLD: 3, EPOCH: 13, valid_loss: 0.016914762929081915
FOLD: 3, EPOCH: 14, train_loss: 0.016702882153596603
FOLD: 3, EPOCH: 14, valid_loss: 0.016855181604623796
FOLD: 3, EPOCH: 15, train_loss: 0.016546128438088763
FOLD: 3, EPOCH: 15, valid_loss: 0.016807144656777382
FOLD: 3, EPOCH: 16, train_loss: 0.016328136861121573
FOLD: 3, EPOCH: 16, valid_loss: 0.0166126412153244
FOLD: 3, EPOCH: 17, train_loss: 0.016119792037421747
FOLD: 3, EPOCH: 17, valid_loss: 0.016462809592485427
FOLD: 3, EPOCH: 18, train_loss: 0.015870812773501792
FOLD: 3, EPOCH: 18, valid_loss: 0.016505549289286138
FOLD: 3, EPOCH: 19, train_loss: 0.015563015756355662
FOLD: 3, EPOCH: 19, valid_loss: 0.0162929405272007
FOLD: 3, EPOCH: 20, train_loss: 0.0151665693237668
FOLD: 3, EPOCH: 20, valid_loss: 0.016122642904520035
FOLD: 3, EPOCH: 21, train_loss: 0.014708973416665784
FOLD: 3, EPOCH: 21, valid_loss: 0.016116833351552486
FOLD: 3, EPOCH: 22, train_loss: 0.014291971206006145
FOLD: 3, EPOCH: 22, valid_loss: 0.01604284904897213
FOLD: 3, EPOCH: 23, train_loss: 0.01397979850595703
FOLD: 3, EPOCH: 23, valid_loss: 0.016011085473001004
FOLD: 3, EPOCH: 24, train_loss: 0.013815053592936522
FOLD: 3, EPOCH: 24, valid_loss: 0.01601501151919365
FOLD: 4, EPOCH: 0, train_loss: 0.4814382762249027
FOLD: 4, EPOCH: 0, valid_loss: 0.024914344623684883
FOLD: 4, EPOCH: 1, train_loss: 0.021361648973052195
FOLD: 4, EPOCH: 1, valid_loss: 0.0193980748206377
FOLD: 4, EPOCH: 2, train_loss: 0.01892429678289055
FOLD: 4, EPOCH: 2, valid_loss: 0.01807621240615845
FOLD: 4, EPOCH: 3, train_loss: 0.017710748255303523
FOLD: 4, EPOCH: 3, valid_loss: 0.017391298189759253
FOLD: 4, EPOCH: 4, train_loss: 0.017085795987676195
FOLD: 4, EPOCH: 4, valid_loss: 0.017147911451756953
FOLD: 4, EPOCH: 5, train_loss: 0.01685315921117051
FOLD: 4, EPOCH: 5, valid_loss: 0.017055432088673116
FOLD: 4, EPOCH: 6, train_loss: 0.016976427312205437
FOLD: 4, EPOCH: 6, valid_loss: 0.017009622342884542
```

```
FOLD: 4, EPOCH: 7, train_loss: 0.016933950215426027
FOLD: 4, EPOCH: 7, valid_loss: 0.017158462963998318
FOLD: 4, EPOCH: 8, train_loss: 0.017022862352532187
FOLD: 4, EPOCH: 8, valid_loss: 0.017318418174982073
FOLD: 4, EPOCH: 9, train_loss: 0.01705175664174516
FOLD: 4, EPOCH: 9, valid_loss: 0.017287997528910637
FOLD: 4, EPOCH: 10, train_loss: 0.016970691486534212
FOLD: 4, EPOCH: 10, valid_loss: 0.016964208111166956
FOLD: 4, EPOCH: 11, train_loss: 0.016943717198002906
FOLD: 4, EPOCH: 11, valid_loss: 0.017264471501111985
FOLD: 4, EPOCH: 12, train_loss: 0.016905814263538845
FOLD: 4, EPOCH: 12, valid_loss: 0.01703455775976181
FOLD: 4, EPOCH: 13, train_loss: 0.016825945325651948
FOLD: 4, EPOCH: 13, valid_loss: 0.01702758491039276
FOLD: 4, EPOCH: 14, train_loss: 0.016757222063535332
FOLD: 4, EPOCH: 14, valid_loss: 0.01686743963509798
FOLD: 4, EPOCH: 15, train_loss: 0.01653239261485687
FOLD: 4, EPOCH: 15, valid_loss: 0.016775177568197252
FOLD: 4, EPOCH: 16, train_loss: 0.016369672016981914
FOLD: 4, EPOCH: 16, valid_loss: 0.016644030027091503
FOLD: 4, EPOCH: 17, train_loss: 0.016129688708251027
FOLD: 4, EPOCH: 17, valid_loss: 0.016494632251560688
FOLD: 4, EPOCH: 18, train_loss: 0.01589710402878977
FOLD: 4, EPOCH: 18, valid_loss: 0.016339619494974613
FOLD: 4, EPOCH: 19, train_loss: 0.01558316388756645
FOLD: 4, EPOCH: 19, valid_loss: 0.016252162493765356
FOLD: 4, EPOCH: 20, train_loss: 0.015173551879924576
FOLD: 4, EPOCH: 20, valid_loss: 0.016077344305813313
FOLD: 4, EPOCH: 21, train_loss: 0.014750995673239231
FOLD: 4, EPOCH: 21, valid_loss: 0.016080063804984092
FOLD: 4, EPOCH: 22, train_loss: 0.014366832479429083
FOLD: 4, EPOCH: 22, valid_loss: 0.016005329862236975
FOLD: 4, EPOCH: 23, train_loss: 0.014000736170632093
FOLD: 4, EPOCH: 23, valid_loss: 0.015978044904768465
FOLD: 4, EPOCH: 24, train_loss: 0.013829051802048877
FOLD: 4, EPOCH: 24, valid_loss: 0.015968659184873103
FOLD: 5, EPOCH: 0, train_loss: 0.4802205621638671
FOLD: 5, EPOCH: 0, valid_loss: 0.02413658283650875
FOLD: 5, EPOCH: 1, train_loss: 0.02126514426350188
FOLD: 5, EPOCH: 1, valid_loss: 0.019291314631700515
FOLD: 5, EPOCH: 2, train_loss: 0.01904139505559895
FOLD: 5, EPOCH: 2, valid_loss: 0.018079844638705253
FOLD: 5, EPOCH: 3, train_loss: 0.017673574711139103
FOLD: 5, EPOCH: 3, valid_loss: 0.01754952758550644
FOLD: 5, EPOCH: 4, train_loss: 0.01707403847098756
FOLD: 5, EPOCH: 4, valid_loss: 0.017020833753049372
FOLD: 5, EPOCH: 5, train_loss: 0.016875136092457235
FOLD: 5, EPOCH: 5, valid_loss: 0.017544100135564802
```

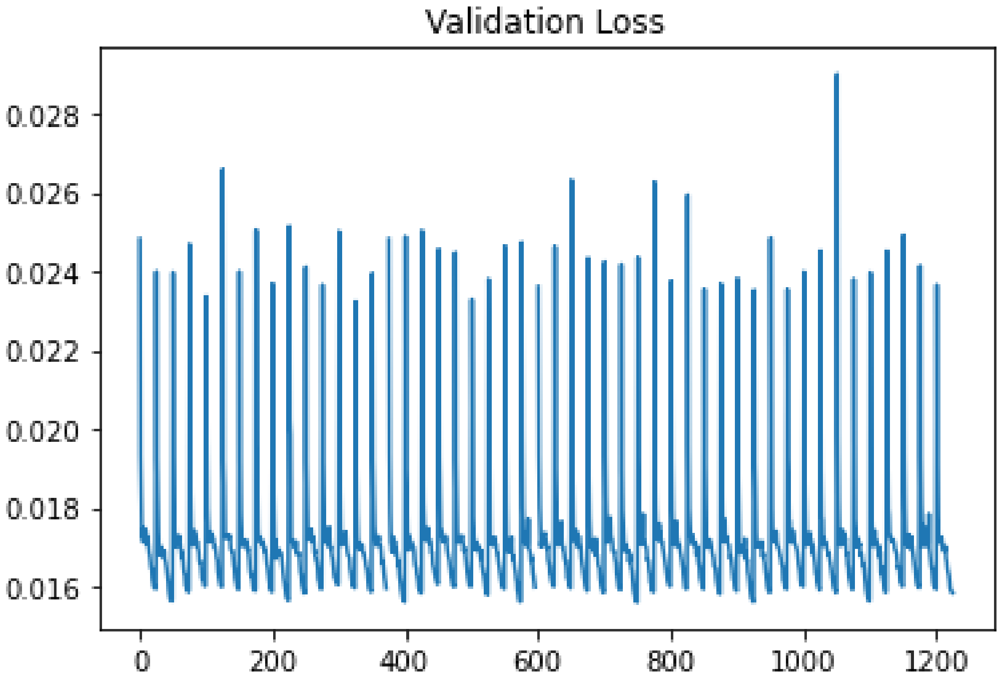
```
FOLD: 5, EPOCH: 6, train_loss: 0.016861857966754306
FOLD: 5, EPOCH: 6, valid_loss: 0.017278988882899284
FOLD: 5, EPOCH: 7, train_loss: 0.01700489751386399
FOLD: 5, EPOCH: 7, valid_loss: 0.017168056443333624
FOLD: 5, EPOCH: 8, train_loss: 0.01706573140940496
FOLD: 5, EPOCH: 8, valid_loss: 0.017130962535738946
FOLD: 5, EPOCH: 9, train_loss: 0.017041101608247983
FOLD: 5, EPOCH: 9, valid_loss: 0.017088485285639762
FOLD: 5, EPOCH: 10, train_loss: 0.017026309031663703
FOLD: 5, EPOCH: 10, valid_loss: 0.01712613705545664
FOLD: 5, EPOCH: 11, train_loss: 0.017002666229698933
FOLD: 5, EPOCH: 11, valid_loss: 0.017003004290163517
FOLD: 5, EPOCH: 12, train_loss: 0.016979180574163692
FOLD: 5, EPOCH: 12, valid_loss: 0.01709418583661318
FOLD: 5, EPOCH: 13, train_loss: 0.016843714079737258
FOLD: 5, EPOCH: 13, valid_loss: 0.01783735156059265
FOLD: 5, EPOCH: 14, train_loss: 0.016679469413649874
FOLD: 5, EPOCH: 14, valid_loss: 0.016959945634007455
FOLD: 5, EPOCH: 15, train_loss: 0.016565393016603935
FOLD: 5, EPOCH: 15, valid_loss: 0.01680417448282242
FOLD: 5, EPOCH: 16, train_loss: 0.016330036436061877
FOLD: 5, EPOCH: 16, valid_loss: 0.016605207063257695
FOLD: 5, EPOCH: 17, train_loss: 0.016166272794916517
FOLD: 5, EPOCH: 17, valid_loss: 0.016463107392191888
FOLD: 5, EPOCH: 18, train_loss: 0.01583819335237855
FOLD: 5, EPOCH: 18, valid_loss: 0.016459889747202396
FOLD: 5, EPOCH: 19, train_loss: 0.015510343008858412
FOLD: 5, EPOCH: 19, valid_loss: 0.016218247637152673
FOLD: 5, EPOCH: 20, train_loss: 0.01512312514669433
FOLD: 5, EPOCH: 20, valid_loss: 0.016065676957368852
FOLD: 5, EPOCH: 21, train_loss: 0.014708964477236174
FOLD: 5, EPOCH: 21, valid_loss: 0.01603235591202974
FOLD: 5, EPOCH: 22, train_loss: 0.014291144869461352
FOLD: 5, EPOCH: 22, valid_loss: 0.015965398848056794
FOLD: 5, EPOCH: 23, train_loss: 0.013965516325821277
FOLD: 5, EPOCH: 23, valid_loss: 0.015956440903246404
FOLD: 5, EPOCH: 24, train_loss: 0.013813428726478094
FOLD: 5, EPOCH: 24, valid_loss: 0.015930245742201806
FOLD: 6, EPOCH: 0, train_loss: 0.480280967365925
FOLD: 6, EPOCH: 0, valid_loss: 0.023650210872292517
FOLD: 6, EPOCH: 1, train_loss: 0.021199466711303003
FOLD: 6, EPOCH: 1, valid_loss: 0.019135429859161376
FOLD: 6, EPOCH: 2, train_loss: 0.018989772630893454
FOLD: 6, EPOCH: 2, valid_loss: 0.017774301692843438
FOLD: 6, EPOCH: 3, train_loss: 0.017688723483762773
FOLD: 6, EPOCH: 3, valid_loss: 0.01734578989446163
FOLD: 6, EPOCH: 4, train_loss: 0.01713540465856085
FOLD: 6, EPOCH: 4, valid_loss: 0.01712867643684149
```

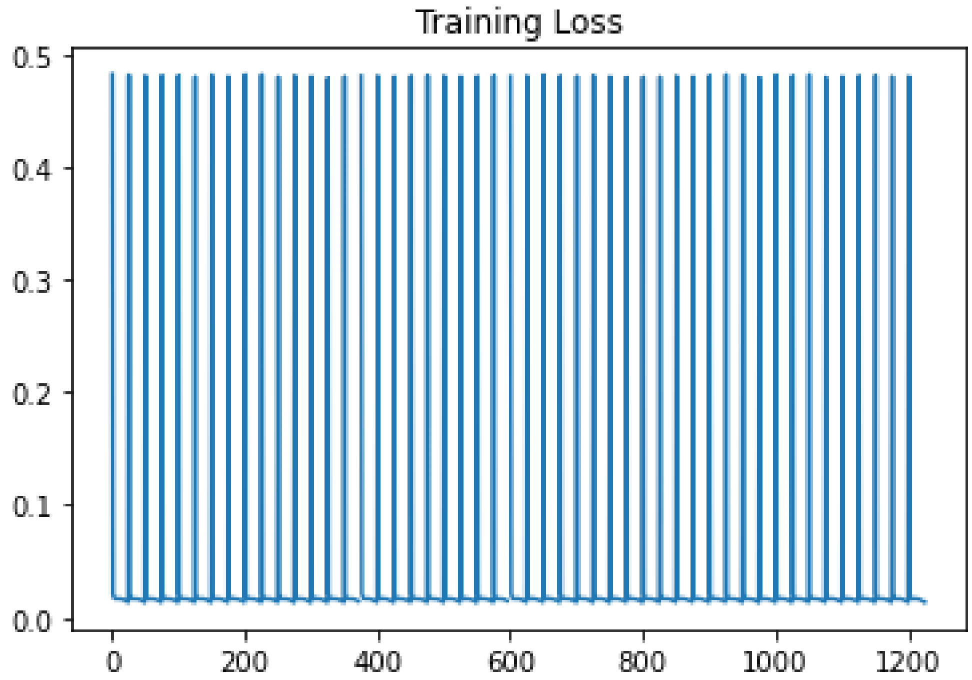
```
FOLD: 6, EPOCH: 5, train_loss: 0.016926547778504237
FOLD: 6, EPOCH: 5, valid_loss: 0.01710812121629715
FOLD: 6, EPOCH: 6, train_loss: 0.0170036763282252
FOLD: 6, EPOCH: 6, valid_loss: 0.017142675705254077
FOLD: 6, EPOCH: 7, train_loss: 0.017026163301855125
FOLD: 6, EPOCH: 7, valid_loss: 0.01723992347717285
FOLD: 6, EPOCH: 8, train_loss: 0.017082421558902782
FOLD: 6, EPOCH: 8, valid_loss: 0.01707722622901201
FOLD: 6, EPOCH: 9, train_loss: 0.017043254129132445
FOLD: 6, EPOCH: 9, valid_loss: 0.01698746148496866
FOLD: 6, EPOCH: 10, train_loss: 0.017063623471629052
FOLD: 6, EPOCH: 10, valid_loss: 0.01691389985382557
FOLD: 6, EPOCH: 11, train_loss: 0.01701099362832551
FOLD: 6, EPOCH: 11, valid_loss: 0.016918593309819697
FOLD: 6, EPOCH: 12, train_loss: 0.016962517496375812
FOLD: 6, EPOCH: 12, valid_loss: 0.016884964928030967
FOLD: 6, EPOCH: 13, train_loss: 0.016819300206035982
FOLD: 6, EPOCH: 13, valid_loss: 0.017013469077646733
FOLD: 6, EPOCH: 14, train_loss: 0.016720949387063786
FOLD: 6, EPOCH: 14, valid_loss: 0.01670393142849207
FOLD: 6, EPOCH: 15, train_loss: 0.01657447481185806
FOLD: 6, EPOCH: 15, valid_loss: 0.01661783494055271
FOLD: 6, EPOCH: 16, train_loss: 0.016433359244141448
FOLD: 6, EPOCH: 16, valid_loss: 0.016441736556589603
FOLD: 6, EPOCH: 17, train_loss: 0.016177555254730237
FOLD: 6, EPOCH: 17, valid_loss: 0.016382538452744483
FOLD: 6, EPOCH: 18, train_loss: 0.01592019636842872
FOLD: 6, EPOCH: 18, valid_loss: 0.01630394671112299
FOLD: 6, EPOCH: 19, train_loss: 0.015576305495080899
FOLD: 6, EPOCH: 19, valid_loss: 0.016181243360042574
FOLD: 6, EPOCH: 20, train_loss: 0.01525749311763413
FOLD: 6, EPOCH: 20, valid_loss: 0.015973382778465748
FOLD: 6, EPOCH: 21, train_loss: 0.014793983508585667
FOLD: 6, EPOCH: 21, valid_loss: 0.01592020835727453
FOLD: 6, EPOCH: 22, train_loss: 0.01439013465174607
FOLD: 6, EPOCH: 22, valid_loss: 0.01586281456053257
FOLD: 6, EPOCH: 23, train_loss: 0.014098920555291126
FOLD: 6, EPOCH: 23, valid_loss: 0.015853472501039505
FOLD: 6, EPOCH: 24, train_loss: 0.013911979653093279
FOLD: 6, EPOCH: 24, valid_loss: 0.015843976363539694
```

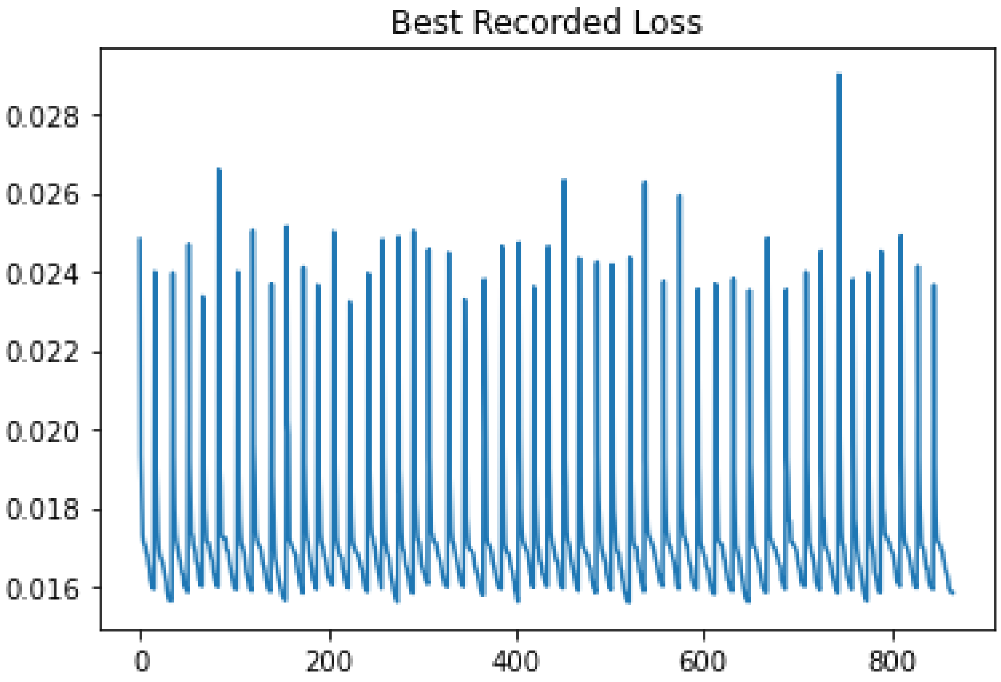
[99]:

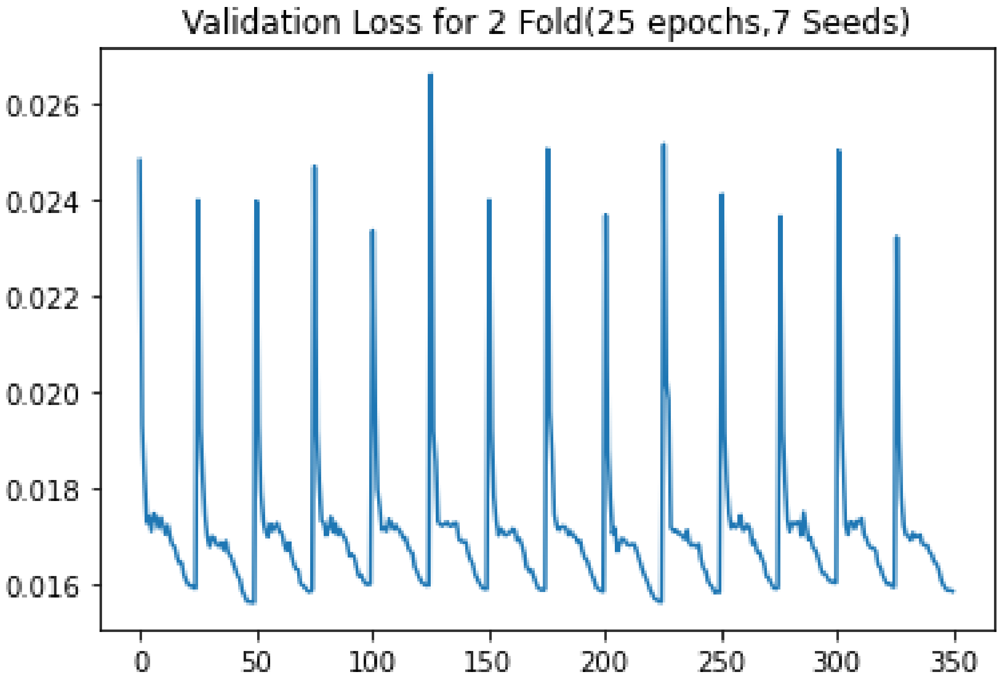
```
import matplotlib.pyplot as plt
import numpy as np
x = np.linspace(0,1225,1225);
plt.title("Validation Loss")
plt.plot(x,valid_loss_)
plt.show()
```

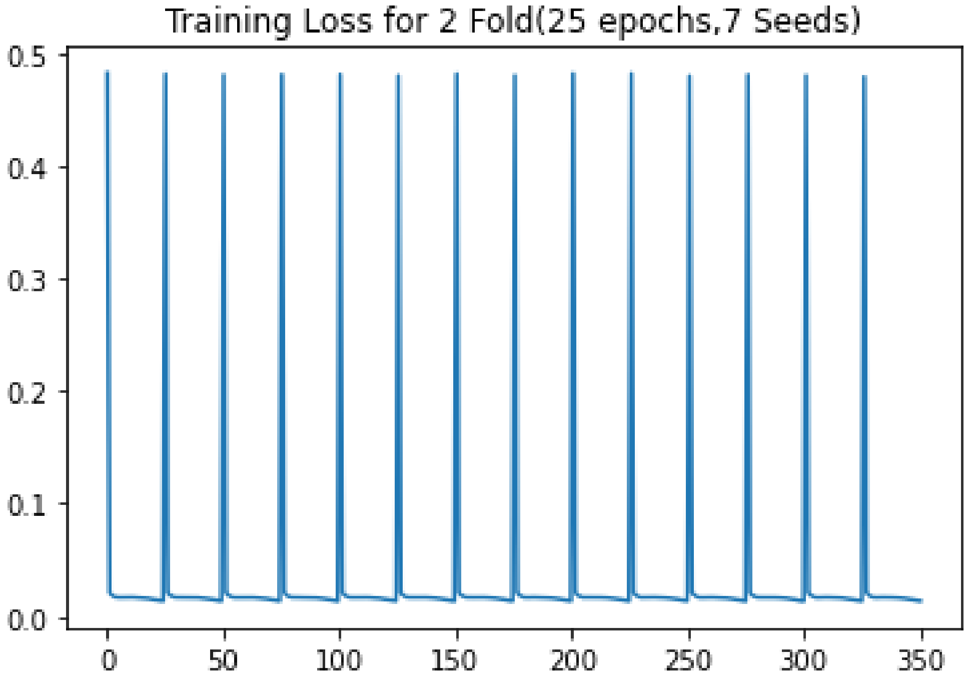
```
x = np.linspace(0,1225,1225);
plt.title("Training Loss")
plt.plot(x,train_loss_)
plt.show()

x = np.linspace(0,865,865);
plt.title("Best Recorded Loss")
plt.plot(x,best_loss_)
plt.show()

x = np.linspace(0,350,350);
plt.title("Validation Loss for 2 Fold(25 epochs,7 Seeds)")
plt.plot(x,valid_loss_[:350])
plt.show()

x = np.linspace(0,350,350);
plt.title("Training Loss for 2 Fold(25 epochs,7 Seeds)")
plt.plot(x,train_loss_[:350])
plt.show()

x = np.linspace(0,125,125);
plt.title("Approx. Best Recorded Loss for 2 Fold(25 epochs,7 Seeds)")
plt.plot(x,best_loss_[:125])
plt.show()
```

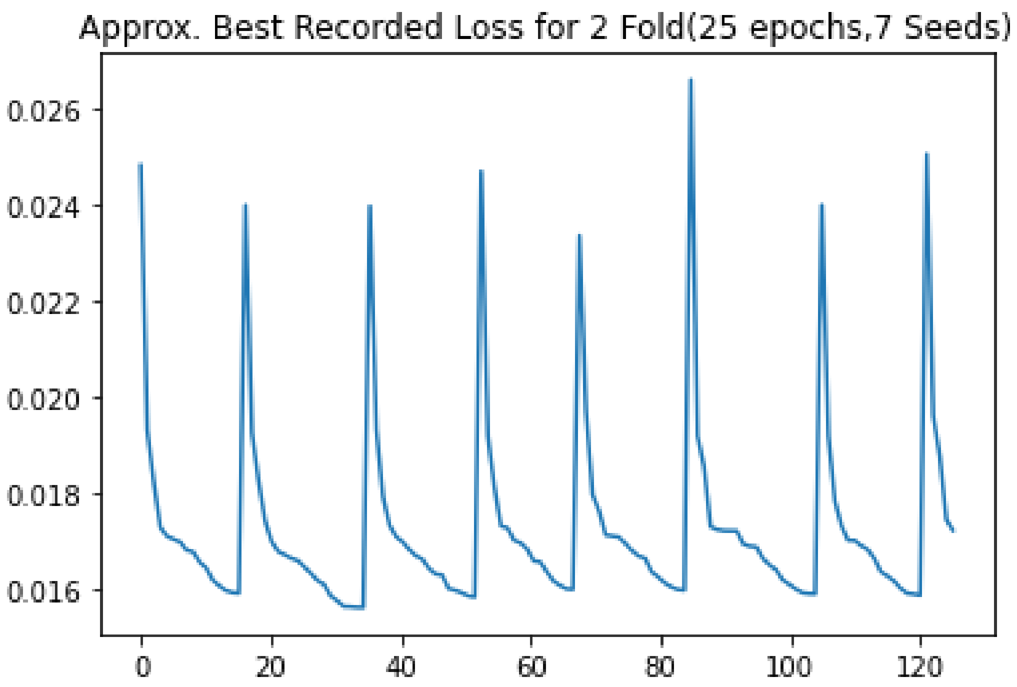



Training Loss
0.5
0.4
0.3
0.2
0.1
0.0
0
200
400
600
800
1000
1200


Best Recorded Loss
0.028
0.026
0.024
0.022
0.020
0.018
0.016
0
200
400
600
800


Validation Loss for 2 Fold(25 epochs,7 Seeds)
0.026
0.024
0.022
0.020
0.018
0.016
0
50
100
150
200
250
300
350


Training Loss for 2 Fold(25 epochs,7 Seeds)
0.5
0.4
0.3
0.2
0.1
0.0
0
50
100
150
200
250
300
350



12 CALCULATING LOG LOSS

[92]:

```
valid_results = train_targets_scored.drop(columns=target_cols).
 ↪merge(train[['sig_id']+target_cols], on='sig_id', how='left').fillna(0)



y_true = train_targets_scored[target_cols].values
y_pred = valid_results[target_cols].values

score = 0
for i in range(len(target_cols)):
    score_ = log_loss(y_true[:, i], y_pred[:, i])
    score += score_ / target.shape[1]

print("CV log_loss: ", score)
```

```
CV log_loss:  0.014467723033458807
```

13 GENERATING SUBMISSION FILE

[93]:

```
sub = sample_submission.drop(columns=target_cols).
 ↪merge(test[['sig_id']+target_cols], on='sig_id', how='left').fillna(0)
sub.to_csv('submission.csv', index=False)
sub.shape
```

```
[93]: (3982, 207)
```