In [14]:
```
import pandas as pd
import numpy as np
import pickle
from tensorflow.keras.models import load_model
import warnings
from sklearn.metrics import log_loss
from tqdm import tqdm
warnings.filterwarnings("ignore")
```

In [224…
```
def final_fun_1(X):

    #encoding categorical features
    X.iloc[:,1] = X.iloc[:,1].map({'trt_cp':0, 'ctl_vehicle':1})
    X.iloc[:,2] = X.iloc[:,2].map({24:0, 48:1, 72:2})
    X.iloc[:,3] = X.iloc[:,3].map({'D1':0, 'D2':1})

    #normalizing gene and cell columns
    transformer = pickle.load(open('transform.pkl','rb'))
    X.iloc[:,4:] = transformer.transform(X.iloc[:,4:])

    #getting more features using auto-encoder
    encoder = load_model('encoder.h5')
    encoded_features = encoder.predict(X.iloc[:,1:])

    #adding encoded features with original features
    total_X = pd.concat([X,pd.DataFrame(encoded_features)], axis=1)

    #loading pre-trained model
    model = pickle.load(open('final_model.pkl','rb'))

    #loading column names of target columns
    columns = pickle.load(open('target_columns.pkl','rb'))

    #predictions
    pred = model.predict(total_X.iloc[:,1:])
    pred_data = pd.DataFrame(pred, columns = columns)
    pred_data.insert(loc=0,column='sig_id',value = X['sig_id'])

    pred_prob = model.predict_proba(total_X.iloc[:,1:])
    pred_prob_data = pd.DataFrame(pred_prob,columns=columns)
    pred_prob_data.insert(loc=0,column='sig_id',value = X['sig_id'])

    return pred_data, pred_prob_data
```

In [37]:
```
def final_fun_2(X,y):

    #encoding categorical features
    X.iloc[:,1] = X.iloc[:,1].map({'trt_cp':0, 'ctl_vehicle':1})
    X.iloc[:,2] = X.iloc[:,2].map({24:0, 48:1, 72:2})
    X.iloc[:,3] = X.iloc[:,3].map({'D1':0, 'D2':1})

    #normalizing gene and cell columns
    transformer = pickle.load(open('transform.pkl','rb'))
    X.iloc[:,4:] = transformer.transform(X.iloc[:,4:])

    #getting more features using auto-encoder
    encoder = load_model('encoder.h5')
    encoded_features = encoder.predict(X.iloc[:,1:])

    #adding encoded features with original features
    total_X = pd.concat([X,pd.DataFrame(encoded_features)], axis=1)

    #loading pre-trained model
    model = pickle.load(open('final_model.pkl','rb'))

    #prediction
    pred_prob = model.predict_proba(total_X.iloc[:,1:])

    metric_value = []    #list to store log-loss of each target features

    for i in tqdm(range(y.shape[1]-1)):    #iterating over each target columns
        loss = log_loss(y.iloc[:,i],pred_prob[:,i], labels=[0, 1])    #computing log-loss
        metric_value.append(loss)


    return np.mean(metric_value)
```