在code資料夾中總共有五份程式，以下以程式1-5做說明:

程式1

按照順序執行，先驗證版本並且安裝套件

step 1.1: 會進行rnadom seed的挑選，這個步驟會切割出訓練集與測試集，會影響後續模型的預測效果，請謹慎執行，固定下來後也能確保再現性，詳細流程參考註解

setp 1.2: 套用你找到的random seed然後切割資料集，使用deepchem package中的csv\_loader，input :csv file / output: deepchem dataset object，此步驟中會再由訓練集中切割出5 fold cross-validation dataset

step 1.3.1 &1.3.2: hyperparameter searching of GCN/ D\_MPNN 因為兩個模型各自的參數不同分成兩個，參數細節請看註解與deepchem說明文件

step 1.3.3: Optuna套件提供的視覺化功能，能視覺化哪個參數在尋找超參數過程中影響力大

step 1.4: 套用你找到的超參數，初始化你的模型，進行後續正式的訓練，當你想要重新開始訓練模型時需執行gc.collect() ＆ torch.cuda.empty\_cache()清除記憶體，並重新執行step 1.4 初始化模型才不會被之前的訓練結果影響

step 1.5 為主要的模型訓練與紀錄模型效能的程式，細節參考註解，記得return值的順序有指定，另外此步驟會儲存best performance checkpoint，也記得儲存下來可以利用model.restore() 這個功能load trained models，防止訓練意外中斷，並用於預測沒看過的compound

step 1.6: 紀錄best performance model預測test set的數值，還有一份統計觀察值與預測值大於60分鐘的compound的文件，以及紀錄training/validation/testing curve數值的文件

step 1.7: 利用step 1.5儲存的best performance checkpoint 或是任意 checkpoint對沒看過的compound進行預測，輸出的值可以是 ln(half-lfe) 與training時用的單位一致，或是minutes

step 1.8: 畫出train/ valid/ test-set各自在不同epoch訓練下的performance，目前有R2/ RMSE

step 1.9: 把model對於testset的預測值轉變成classification，分成 unstable(<30 min)/ moderate (30-60 min)/ stable (> 60 min)三個classes，並print 出Confusion Matrix/ classification\_report/ matthews\_corrcoef

step 1.10: 畫出分子的功能，能決定一列畫幾個分子並儲存成圖

—------------------------------------------------------------------------------------------------------------------------

程式2

2.1 - 2.4:就是根據dataset這個資料夾中的CSV檔進行各種data selection，細節請參考註解。

2.5: 是透過oversampling minor class實現 even class training，此處的class labels來自於不同clustering methods對於datasets的分類結果，需要因為訓練model時需要輸入CSV所以2.5的output是CSV file，分成train/ test set CSV

2.6: 找Maximal Common Structure(MCS)，有分成直接尋找，或是先抓出scaffold在找MCS

2.7 確定 2.6中找到的MCS於dataset中的分布頻率，詳細見註解。注意2.7一定需要執行過2,6後再接著執行， 因為會需要2.6中的MCS

------------------------------------------------------------------------------------------------------------------------

程式3

按照順序執行，先驗證版本並且安裝套件

step 3.1: 定義將SMILES轉換成igraph objects作為後續WWL graph kernels分析的input，此處將DMPNN node / edge feature 都存入graph中，如果要換成其他的GNN 架構則可以參考註解一步步完成

step 3.2: 將dataset用step 3.1的function進行轉換並儲存igraph objects

step 3.3.1 & 3.3.2: WWL graph kernels分析的主要流程，並在step 3.3.2執行完後會獲得WWL distance matrix (in shape: N\*N, N= number of samples of dataset)

step 3.4: 評估WWL graph kernels結果，原始paper作者使用resulting WWL distance matrix進行SVM classifier prediction，然後藉由SVM classifier的performance來評估選擇哪個iteration的WWL distance matrix，此處我們換成SVR regressorl來進行相同的評估

—------------------------------------------------------------------------------------------------------------------------

程式4

先執行step 4.1(load chosen WWL distance matrix as input) & 4.2 (select random seed for tsne projection using in visualizing different clustering methods)

step 4.3 - 4.5 沒有順序， 逐個執行就好，如何挑選最好的cluster result參考註解

step 4.3 (Kmeans): output包含silhouette plot & tsne plot; elbow method plot; CSV file including SMILES, half-life, cluster labels, this csv file can used in 程式2

step 4.4 (HDBSCAN): output包含tsne plot; CSV file including SMILES, half-life, cluster labels, this csv file can used in 程式2

step 4.3 (Kmedoids, 包含FastPAM1, FasterPAM, FasterMSC三種): output包含medoid\_silhouette\_score &minMaxscale\_kmedoids\_inertia的score curves, tsne plots; CSV file including SMILES, half-life, cluster labels, this csv file can used in 程式2

step 4.6: 根據cluster labels(在同cluster內)尋找MCS，並畫出MCS 與疊加MCS於source compounds上的圖

step 4.7: 利用unstable(<30 min)/ moderate (30-60 min)/ stable (> 60 min) 的class資訊，與cluster labels，兩種labeling畫出MDS plot，以視覺化不同clustering methods對於WWL distance matrix的影響，與cluster labels如何在資料點間分布

—------------------------------------------------------------------------------------------------------------------------程式5

5.1.1& 5.1.2 產生不同descriptors，根據需求使用，請參考註解

5.2: input: 承 5.1.1& 5.1.2或是任意fingerprint files，畫出tsne plots，並利用KL divergence來決定最終進行ML model時使用哪個fingerprint較為合適

5.3 - 5.6 參考註解就可以完成ML methods(RF, SVM, XGBoost) 的random seed(for splitting dataset and reproduciable) ; hyperparameter optiminzation; ML training and prediction