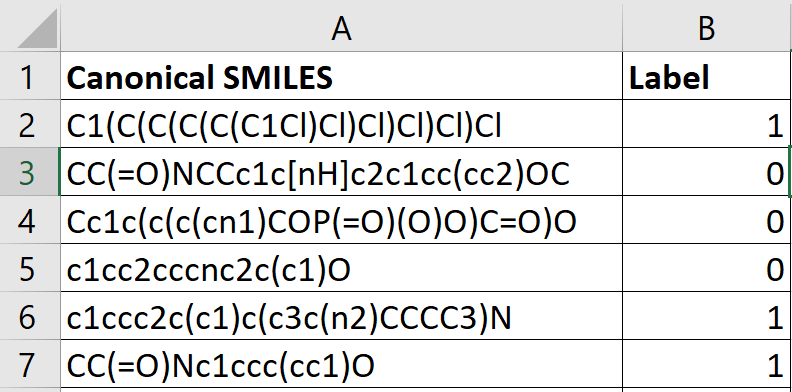
**Steps to run the Pipeline Code –**

**Steps A), B), C) and D) are necessary to execute to prepare final Input Data for main Pipeline Code -**

**Note-** Please make sure that following Folders are empty before running –

* 1. SMI Files
  2. UGRNN\ugrnn\data\DILI\Model\_Train\_DILI\_Data

1. Place External Test Data inside Folder **“External Test Set”** with File Name “**External\_Test\_Set”** in “**csv”** format.



Example Image of **“External\_Test\_Set.csv”**(Needed in Exact Above Format)

1. In **“Scripts”** Folder, Run **“Create\_SMI\_Files.py”**

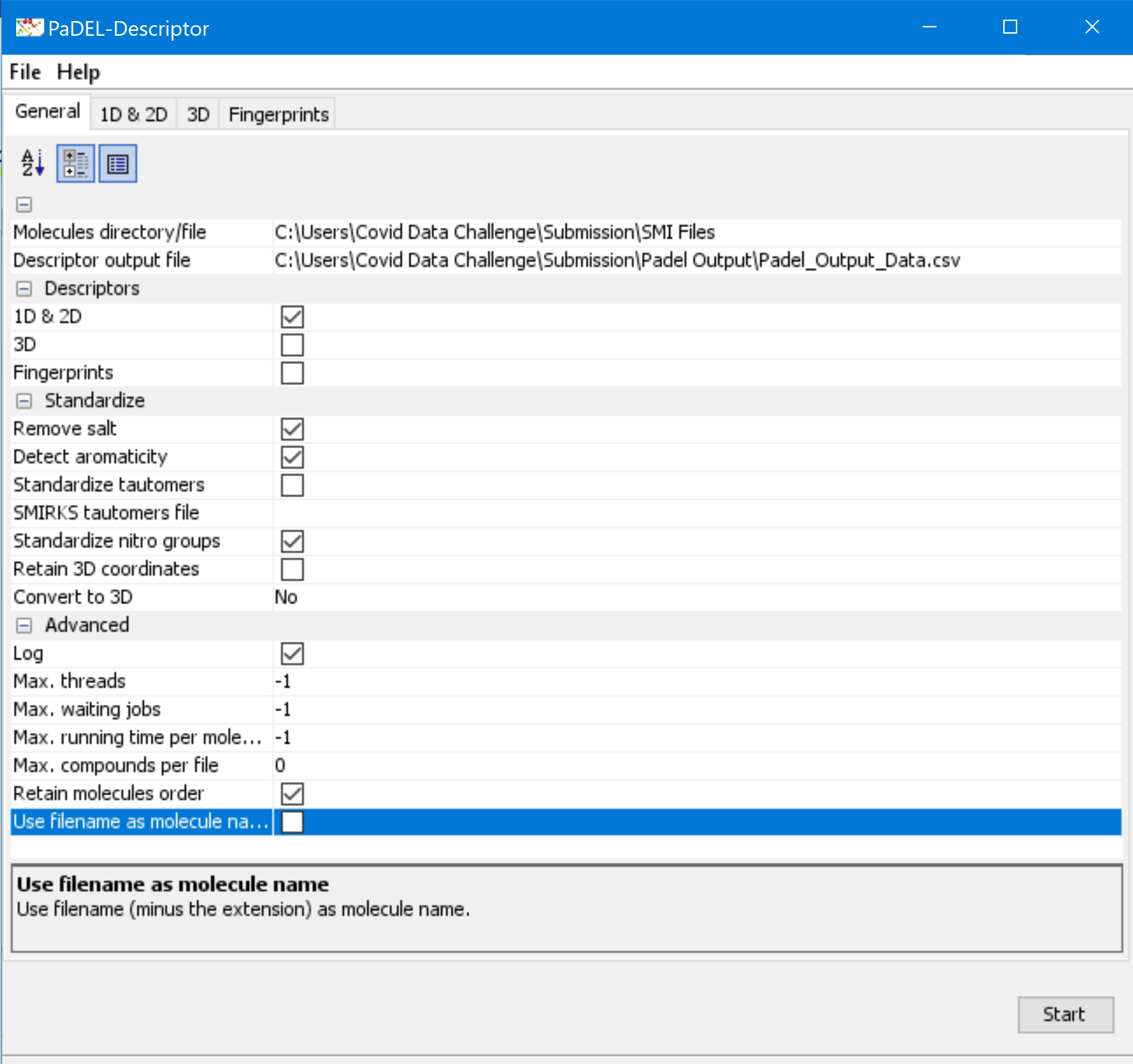
(This will create .smi files for each Molecule present in file **“External\_Test\_Set.csv”** in Folder **“SMI Files”**)

1. In Folder **“PaDEL-Descriptor”,** Run **“PaDEL-Descriptor Tool”** as follows–
   1. In Command line,

Change Directory to Folder **“PaDEL-Descriptor” –>** **cd <path>/Submission/PaDEL-Descriptor**

Next, Run command -> **java -jar PaDEL-Descriptor.jar**

**(This will Open the PaDel Descriptor Tool as shown in below image)**



PaDel-Descriptor Tool Window

* 1. In Option “Molecules directory/file” – Give location of Folder **“SMI Files/”**
  2. In Option “Descriptor Output File” – Give Location of Folder **“Padel Output”** with Output File Name **“Padel\_Output\_Data.csv”**
  3. Keep the Default Features on as shown in above Image and click on “**Start”**

**(This will create file “Padel\_Output\_Data.csv” in Folder “Padel Output” containing 1444 features for each molecule present in “External\_Test\_Set.csv”)**

1. In Folder **“Script”**, Run **“Extract\_PaDel\_features.py”**

(This will generate **“Input\_Data.csv”** in Folder **“Input Data”** to be given as input to Final Pipeline Code)

1. In Folder **“Script”**, Run **“Main\_Pipeline\_Script.py”**

**(This is the main Script Containing the Pipeline as follows)**

**def load\_model(file):**

#function to load Catboost Classification Model

**def create\_descriptor(smiles,choice):**

#This function combines the PaDEL and Graph Kernel UGRNN Features to make the final data for doing Classification Predictions

**def Save\_Results(pred,probs):**

#This function generates and saves the External Test Set Results in file **“External\_Test\_Set\_Result\_Predictions.csv”** in Folder **“External Test Set Results”**

**def calculate\_metrics(label,pred,probs):**

#This function is called inside above **“Save\_Results()”** Function and is used to generate Evaluation Metrics and save them in file **“External\_Test\_Set\_Results\_Metrics.csv”** in Folder **“External Test Set Results”**

**if \_\_name\_\_ == '\_\_main\_\_':**

model = load\_model(file\_path**)#loads model**

**#NECESSARY - RUN BELOW CODE FOR GENERATING GRAPH KERNEL UGRNN FEATURES**

**#Trains UGRNN Model on our previous Train and Validation Set**

os.chdir("../UGRNN/ugrnn/")

if (os.system('python train.py') == 0):

print("UGRNN Embeddings Generated Sucessfully")

else:

print("UGRNN Embeddings Generation Failed")

sys.exit("UGRNN Embeddings Generation Failed")

os.chdir("../../Scripts/")

fp = create\_descriptor(smiles,fp\_type**)#Prepares final data for making predictions**

pred = model.predict(fp)**#For Classifiation Predictions**

probs = model.predict\_proba(fp)**#For Classification Confidence**

Save\_Results(pred,probs)**#This generates and saves the External Set Results and Evaluation Metrics**

**For Reproducing our Train and Test Set Results –**

1. In Folder **“Scripts”,** Run script **“Produce\_Our\_Train\_Test\_Results.csv”**

(This will generate the Predictions Results and Evaluation Metrics for Our Train and Test set used and save them in Folder **“Our Train and Test Results”**)

(The Train and Test Set Used is Present in Folder **“Our Train and Test Set”)**