BIOInformatics with Python practice proj 4/28/24

* Regression Model (USE jupyter notebook)
  * chembl database → can be used for original bstm2 projects

Pip install chembl resource client
(is in python library)
→ do our due diligence on chembl to pick what protein etc you want to work on.

Import pandas, Import chembl client
→ Import chembl (exact code →
From chembl_webresource_client.new client
Import new_client

Search for target proteins
(like you search in search bar)
In this example we are searching for 'coronavirus' → (so like we're searching in search bar coronavirus)

DATA COLLECTION AND PREPARATION

TARGET SEARCH

New client → target

target search (protein/enzyme)

assigned as variable

then

targets is variable for search in dictionary

then put in data frame

print (refer to code in bioinfo folder)

(Data frame is with Pandas)

* You won't be able to view the data with a .py regular output so make sure to do proj on the jupyter notebook.

*IMPORTANT

so you can load the data frame properly, you just need to write it as targets and NOT print

You are presented with a dataset
now you choose your protein
→ select entry, with ID
→ returns ID

Now we retrieve bioactivity data from the thing which is classified by ID)

So selected target → our targets ID
Activity → thing which is being filtered through

res → data (targeting things with our ID, searching for things considered ICSO?)

filter put on the new_client.activity

*It will take a while to load so don't freak out

you can limit what appears of dataframe
df.head(x)

We can check for unique types in dataframe, .unique()

"We want to define a particular standard type to make our dataset more uniform"

Standard value is potency of the drug So we want standard value to be as low as possible

IC50, the standard value indicates how much of the drug you need to reach 50% inhibition effect

now we write out data frame into CSV file (commonly used to store tabular data) Text file

we set the index to false in order to keep it from being displayed as data in CSV file. ⇒ leads to cluttering to less readable

There's just no need for it to be included | it saves it to the folder

Now we need to be able to access data from google drive

downloaded google API Python client

*Important ⇒ we need to import file (CSV file) from system onto code)

I couldn't find a way to import from google drive. But you use a pd.read(file path) method and you define it as variable

so now we bought the data file which we downloaded onto the code, now we filter all standard values which have no standard value

Data preprocess

SORTING DATA

We Sort Data into 3 types
Active
Inactive
Intermediate

These are all IC50's, so we sort based on value (based on nM amount)

ex: 2 100nM is active

We iterate through all instances of the standard values in our second dataframes

if ---.
elif---
else---

Then we append the label to it's class

Data set is comprised of many compounds/molecules which are drugs

These compounds exert some effect on the target proteins

(ex: you take medication it exists effects on you)

Medicine works on that protein

So each compound has a unique chembl ID, so some compounds

might have similar IDS, in that case we want to remove duplicates to prevent redundancy.

df2.molecule_chembl_ID →
displays chembl IDS
so we iterate through and add to a list

do same for canonical smiles
SMILES (simplified molecular input Line Entry System)
way of representing chemical molecules
ex: the SMILES string for H_2O is "O" and "CC"

same for standard value

we are making a new data frame with only the
SMILES
STANDARD IDS
and Chembl ID

pd.concat → combines multiple data frames into a single one

pd.Series → creates one dimensional arrays that are labled

So we essentially made a new data frame which sorts the compounds to their appropriate class based on nM, also has Chembl ID, canonical_smiles, and standard_value.

Now we create CSV file from this