عنوان پروژه: ساخت مدل یادگیری ماشین برای پیشبینی PIC50 روی بیماری الزایمر و تارگت cholinesterase

درس: طراحی محاسباتی دارو

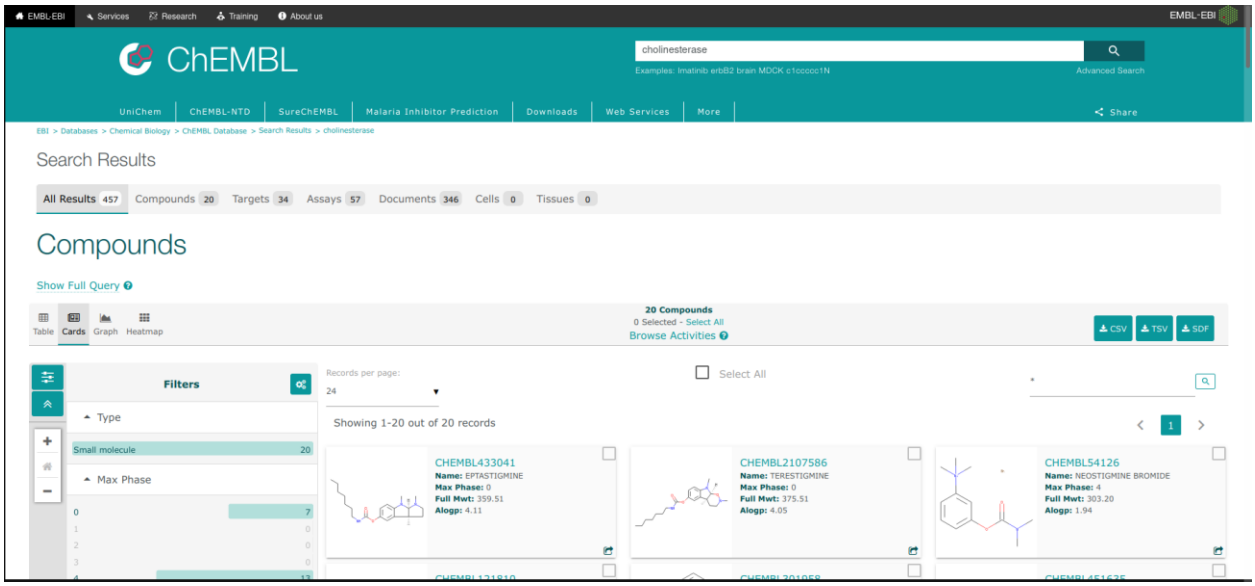
استاد: آقای دکتر قرقانی

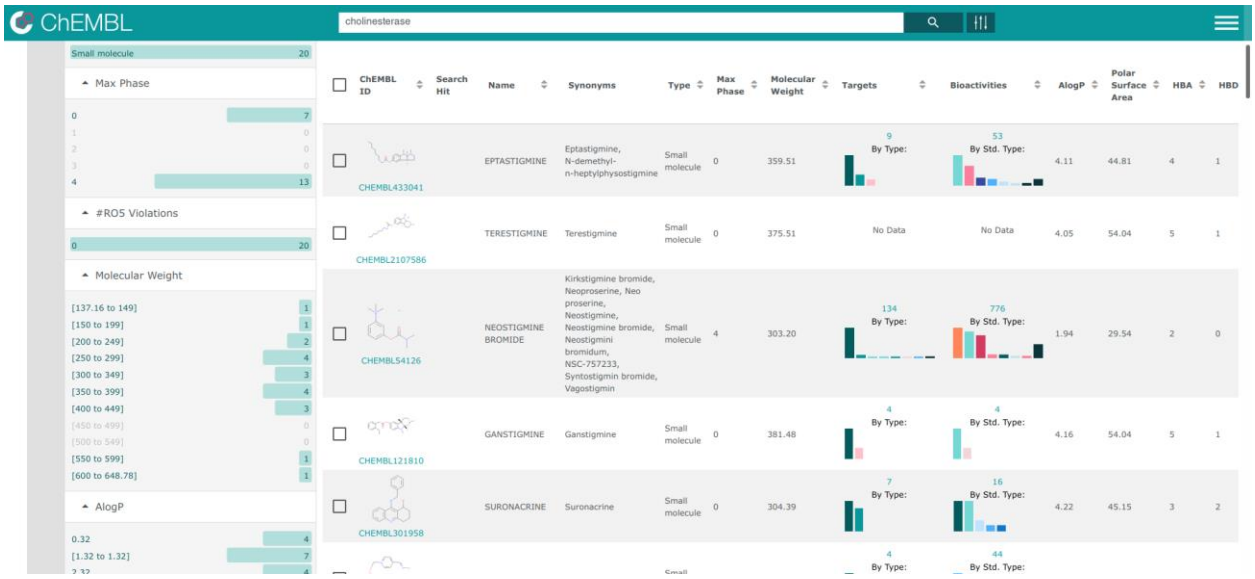
دانشجو: محبوبه گلچین پور لیلی

بهمن ۱۴۰۱

- هدف: ساخت یک مدل یادگیری ماشین با استفاده از دیتای bioactivity از سایت ChEMBL برای پیشبینی PIC50 روی بیماری الزایمر و تارگت cholinesterase
- دیتابیس مورد استفاده در پروژه: وبسایتChEMBL
- زبان برنامه نویسی: پایتون و محیط Jupyter notebook
- بیماری مورد نظر: آلزایمر
- تارگت مورد نظر: cholinesterase

ابتدا تارگت پروتینی مورد نظر مان که cholinesterase است را در سایت ChEMBL از بین تارگت های موجود جست و جو می کنیم. همان طور که مشخص است ۲۰ compounde برای cholinesterase پیدا شد.





در ادامه با استفاده از محیط Jupyter notebook و زبان برنامه نویسی پایتون، جمع آوری داده ها و پیش پردازش داده ها از سایت ChEMBL انجام میشود.

ابتدا پکیج وب سرویس ChEMBL را نصب می کنیم تا دیتای bioactivity را از دیتابیس ChEMBL استخراج کنیم.

Installing libraries

Install the ChEMBL web service package so that we can retrieve bioactivity data from the ChEMBL Database.

```
! pip install chembl_webresource_client
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting chembl_webresource_client
  Downloading chembl_webresource_client-0.10.8-py3-none-any.whl (55 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 55.2/55.2 KB 3.1 MB/s eta 0:00:00
Requirement already satisfied: urllib3 in /usr/local/lib/python3.8/dist-packages (from chembl_webresource_client) (1.24.3)
Requirement already satisfied: requests>=2.18.4 in /usr/local/lib/python3.8/dist-packages (from chembl_webresource_client) (2.25.1)
Collecting requests-cache~=0.7.0
  Downloading requests_cache-0.7.5-py3-none-any.whl (39 kB)
Requirement already satisfied: easydict in /usr/local/lib/python3.8/dist-packages (from chembl_webresource_client) (1.10)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.8/dist-packages (from requests>=2.18.4->chembl_webresource_client) (2022.12.7)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.8/dist-packages (from requests>=2.18.4->chembl_webresource_client) (2.10)
Requirement already satisfied: chardet<5,>=3.0.2 in /usr/local/lib/python3.8/dist-packages (from requests>=2.18.4->chembl_webresource_client) (4.0.0)
Collecting attrs<22.0,>=21.2
  Downloading attrs-21.4.0-py2.py3-none-any.whl (60 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 60.6/60.6 KB 6.4 MB/s eta 0:00:00
Collecting url-normalize<2.0,>=1.4
  Downloading url_normalize-1.4.3-py2.py3-none-any.whl (6.8 kB)
Collecting itsdangerous>=2.0.1
  Downloading itsdangerous-2.1.2-py3-none-any.whl (15 kB)
Requirement already satisfied: pyyaml>=5.4 in /usr/local/lib/python3.8/dist-packages (from requests-cache~=0.7.0->chembl_webresource_client) (6.0)
Requirement already satisfied: six in /usr/local/lib/python3.8/dist-packages (from url-normalize<2.0,>=1.4->requests-cache~=0.7.0->chembl_webresource_client
Installing collected packages: url-normalize, itsdangerous, attrs, requests-cache, chembl_webresource_client
  Attempting uninstall: itsdangerous
    Found existing installation: itsdangerous 1.1.0
    Uninstalling itsdangerous-1.1.0:
      Successfully uninstalled itsdangerous-1.1.0
  Attempting uninstall: attrs
    Found existing installation: attrs 22.2.0
    Uninstalling attrs-22.2.0:
      Successfully uninstalled attrs-22.2.0
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source of the following dep
flask 1.1.4 requires itsdangerous<2.0,>=0.24, but you have itsdangerous 2.1.2 which is incompatible.
Successfully installed attrs-21.4.0 chembl_webresource_client-0.10.8 itsdangerous-2.1.2 requests-cache-0.7.5 url-normalize-1.4.3
```

در ادامه تارگت مورد نظر ما که در بیماری الزایمر **cholinesterase** انتخاب کردیم را از دیتابیس جست و جو و داده های آن را استخراج میکنیم. همانطور که مشاهد میکنید ده سطر اول داده ها قابل مشاهده می باشد.

Target search for Acetylcholinesterase

What is the target in Alzheimer's disease? The complex nature of neurodegenerative diseases has developed a pressing need to design multitarget-directed ligands to address the complementary pathways involved in these diseases. The major enzyme targets for development of therapeutics for Alzheimer's disease are **cholinesterase** and β-secretase enzymes.

```
[5]
target = new_client.target
target_query = target.search('cholinesterase')
targets = pd.DataFrame.from_dict(target_query)
```

```
targets.head(10)
```

	cross_references	organism	pref_name	score	species_group_flag	target_chembl_id	target_components	target_type	tax_id
0	[{'xref_id': 'P81908', 'xref_name': None, 'xre...	Equus caballus	Cholinesterase	19.0	False	CHEMBL5763	[{'accession': 'P81908', 'component_descriptio...	SINGLE PROTEIN	9796
1	[]	Canis lupus familiaris	Cholinesterase	19.0	False	CHEMBL4630814	[{'accession': 'P32750', 'component_descriptio...	SINGLE PROTEIN	9615
2	[{'xref_id': 'P06276', 'xref_name': None, 'xre...	Homo sapiens	Butyrylcholinesterase	18.0	False	CHEMBL1914	[{'accession': 'P06276', 'component_descriptio...	SINGLE PROTEIN	9606
3	[{'xref_id': 'NBK23441', 'xref_name': 'Butyryl...	Mus musculus	Butyrylcholinesterase	18.0	False	CHEMBL2528	[{'accession': 'Q03311', 'component_descriptio...	SINGLE PROTEIN	10090
4	[]	Homo sapiens	Cholinesterases; ACHE & BCHE	16.0	False	CHEMBL2095233	[{'accession': 'P06276', 'component_descriptio...	SELECTIVITY GROUP	9606
5	[{'xref_id': 'Q9N1N9', 'xref_name': None, 'xre...	Equus caballus	Butyrylcholinesterase	11.0	False	CHEMBL5077	[{'accession': 'Q9N1N9', 'component_descriptio...	SINGLE PROTEIN	9796
6	[{'xref_id': 'Q95P20', 'xref_name': None, 'xre...	Musca domestica	Acetylcholinesterase	11.0	False	CHEMBL5752	[{'accession': 'Q95P20', 'component_descriptio...	SINGLE PROTEIN	7370
7	[]	Plutella xylostella	Acetylcholinesterase	11.0	False	CHEMBL2242729	[{'accession': 'Q8MZM0', 'component_descriptio...	SINGLE PROTEIN	51655
8	[]	Musca domestica	Acetylcholinesterase	11.0	False	CHEMBL2242743	[{'accession': 'Q7YWJ9', 'component_descriptio...	SINGLE PROTEIN	7370
9	[]	Bemisia tabaci	AChE2	11.0	False	CHEMBL2366409	[{'accession': 'B3SST5', 'component_descriptio...	SINGLE PROTEIN	7038

سپس دیتای **bioactivity** مربوط به **cholinesterase** انسانی را استخراج میکنیم برای اینکار از بین داده های موجود داده سطر دوم که ارگانیسم آن با **Homo sapiens** مشخص شده است را انتخاب میکنیم و دیتای **bioactivity** آن را بازیابی میکنیم.

Select and retrieve bioactivity data for *Human Acetylcholinesterase* (second entry)

We will assign the fifth entry (which corresponds to the target protein, *Human Acetylcholinesterase*) to the ***selected_target*** variable

```
[8] selected_target = targets.target_chembl_id[2]
    selected_target
```

```
'CHEMBL1914'
```

```
[ ]
```

Here, we will retrieve only bioactivity data for *Human Acetylcholinesterase* (CHEMBL220) that are reported as pChEMBL values.

```
[9] activity = new_client.activity
    res = activity.filter(target_chembl_id=selected_target).filter(standard_type="IC50")
```

```
res
```


```
[{'activity_comment': None, 'activity_id': 33968, 'activity_properties': [], 'assay_chembl_id': 'CHEMBL654878', 'assay_description': 'Inhibitory concentration against butyrylcholinesterase.', 'assay_type':
'B', 'assay_variant_accession': None, 'assay_variant_mutation': None, 'bao_endpoint': 'BAO_0000190', 'bao_format': 'BAO_0000357', 'bao_label': 'single protein format', 'canonical_smiles':
'CCOc1nn(-c2cccc(OCc3ccccc3)c2)c(=O)o1', 'data_validity_comment': None, 'data_validity_description': None, 'document_chembl_id': 'CHEMBL1148382', 'document_journal': 'J. Med. Chem.', 'document_year': 2004,
'ligand_efficiency': {'bei': '19.33', 'le': '0.36', 'lle': '3.24', 'sei': '9.08'}, 'molecule_chembl_id': 'CHEMBL133897', 'molecule_pref_name': None, 'parent_molecule_chembl_id': 'CHEMBL133897',
'pchembl_value': '6.04', 'potential_duplicate': 0, 'qudt_units': 'http://www.openphacts.org/units/Nanomolar', 'record_id': 252547, 'relation': '=', 'src_id': 1, 'standard_flag': 1, 'standard_relation':
'=', 'standard_text_value': None, 'standard_type': 'IC50', 'standard_units': 'nM', 'standard_upper_value': None, 'standard_value': '920.0', 'target_chembl_id': 'CHEMBL1914', 'target_organism': 'Homo
sapiens', 'target_pref_name': 'Butyrylcholinesterase', 'target_tax_id': '9606', 'text_value': None, 'toid': None, 'type': 'IC50', 'units': 'uM', 'uo_units': 'UO_0000065', 'upper_value': None, 'value':
'0.92'}, {'activity_comment': None, 'activity_id': 37562, 'activity_properties': [], 'assay_chembl_id': 'CHEMBL654878', 'assay_description': 'Inhibitory concentration against butyrylcholinesterase.',
'assay_type': 'B', 'assay_variant_accession': None, 'assay_variant_mutation': None, 'bao_endpoint': 'BAO_0000190', 'bao_format': 'BAO_0000357', 'bao_label': 'single protein format', 'canonical_smiles':
'O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC1CC1', 'data_validity_comment': None, 'data_validity_description': None, 'document_chembl_id': 'CHEMBL1148382', 'document_journal': 'J. Med. Chem.', 'document_year':
2004, 'ligand_efficiency': {'bei': '16.04', 'le': '0.33', 'lle': '1.50', 'sei': '11.85'}, 'molecule_chembl_id': 'CHEMBL336398', 'molecule_pref_name': None, 'parent_molecule_chembl_id': 'CHEMBL336398',
'pchembl_value': '6.05', 'potential_duplicate': 0, 'qudt_units': 'http://www.openphacts.org/units/Nanomolar', 'record_id': 252533, 'relation': '=', 'src_id': 1, 'standard_flag': 1, 'standard_relation':
'=', 'standard_text_value': None, 'standard_type': 'IC50', 'standard_units': 'nM', 'standard_upper_value': None, 'standard_value': '900.0', 'target_chembl_id': 'CHEMBL1914', 'target_organism': 'Homo
sapiens', 'target_pref_name': 'Butyrylcholinesterase', 'target_tax_id': '9606', 'text_value': None, 'toid': None, 'type': 'IC50', 'units': 'uM', 'uo_units': 'UO_0000065', 'upper_value': None, 'value':
'0.9'}, {'activity_comment': None, 'activity_id': 37566, 'activity_properties': [], 'assay_chembl_id': 'CHEMBL654878', 'assay_description': 'Inhibitory concentration against butyrylcholinesterase.',
'assay_type': 'B', 'assay_variant_accession': None, 'assay_variant_mutation': None, 'bao_endpoint': 'BAO_0000190', 'bao_format': 'BAO_0000357', 'bao_label': 'single protein format', 'canonical_smiles':
```


Handling missing data

در ادامه اگر هر کامپوندی مقدارهای ستون **standard_value** و یا ستون **canonical_smiles** نداشته باشد آن کامپوند را حذف می کنیم.

Handling missing data

If any compounds has missing value for the **standard_value** and **canonical_smiles** column then drop it.

```
df2 = df[df.standard_value.notna()]
df2 = df2[df.canonical_smiles.notna()]
df2.head(4)
```

```
<ipython-input-15-989003b2ae9e>:2: UserWarning: Boolean Series key will be reindexed to match DataFrame index.
  df2 = df2[df.canonical_smiles.notna()]
```

	activity_comment	activity_id	activity_properties	assay_chembl_id	assay_description	assay_type	assay_variant_accession	assay_variant_mutation	bao_endpoint	bao_format	...	target_organism	targe
0	None	33968	[]	CHEMBL654878	Inhibitory concentration against butyrylcholin...	B	None	None	BAO_0000190	BAO_0000357	...	Homo sapiens	Butyrylcl
1	None	37562	[]	CHEMBL654878	Inhibitory concentration against butyrylcholin...	B	None	None	BAO_0000190	BAO_0000357	...	Homo sapiens	Butyrylcl
2	None	37566	[]	CHEMBL654878	Inhibitory concentration against butyrylcholin...	B	None	None	BAO_0000190	BAO_0000357	...	Homo sapiens	Butyrylcl
3	None	38901	[]	CHEMBL654878	Inhibitory concentration against butyrylcholin...	B	None	None	BAO_0000190	BAO_0000357	...	Homo sapiens	Butyrylcl

4 rows × 45 columns

سپس سه کلاس مختلف **active, inactive ,intermediate** برای مقادیرستون **standard_value** در نظر می گیریم و در ستون **bioactivity_class** ذخیره می کنیم. مقادیر **canonical_smiles** و **molecule_chembl_id** را نیز از دیتا استخراج و در دو ستون **mol_cid** و **canonical_smiles** قرار میدهیم.

```
bioactivity_class = []
for i in df2.standard_value:
  #print(i)
  if float(i)>= 10000:
    bioactivity_class.append("inactive")
  elif float(i) <= 1000:
    bioactivity_class.append("active")
  else:
    bioactivity_class.append("intermediate")
```

```
mol_cid =[]
for i in df2.molecule_chembl_id:
    mol_cid.append(i)
```

```
#mol_cid
```

```
canonical_smiles =[]
for i in df2.canonical_smiles:
    canonical_smiles.append(i)
```

```
standard_value =[]
for i in df2.standard_value:
    standard_value.append(i)
```

+ Code

Data pre-processing of the bioactivity data

Combine the 3 columns (molecule_chembl_id,canonical_smiles,standard_value) and bioactivity_class into a DataFrame

```
selection = ['molecule_chembl_id','canonical_smiles','standard_value']
df3 = df2[selection]
df3
```

	molecule_chembl_id	canonical_smiles	standard_value
0	CHEMBL460447	CC(=O)O[C@H]1[C@@H](NC(=O)C=C(C)C)CC[C@]2(C)[C...	10000.0
1	CHEMBL94	CNC(=O)Oc1ccc2c(c1)[C@]1(C)CCN(C)[C@@H]1N2C	857.0
2	CHEMBL500603	C/C=C(\C)C(=O)N[C@H]1[C@@H](OC(C)=O)C[C@]2(C)[...	25000.0
3	CHEMBL95	Nc1c2c(nc3ccccc13)CCCC2	4.0
4	CHEMBL502	COc1cc2c(cc1OC)C(=O)C(CC1CCN(Cc3ccccc3)CC1)C2	2700.0
...	...	...	...
5214	CHEMBL4854972	O=S(=O)(c1ccccc1)n1ccc2c(OCCNCCNCc3cccc(C(F)(F...	2118.0
5215	CHEMBL4845823	CCN(C)C(=O)Oc1cccc(CNCCNCCOc2cccc3c2ccn3S(=O)(...	454.5
5216	CHEMBL502	COc1cc2c(cc1OC)C(=O)C(CC1CCN(Cc3ccccc3)CC1)C2	1830.0
5217	CHEMBL95	Nc1c2c(nc3ccccc13)CCCC2	15.0
5218	CHEMBL636	CCN(C)C(=O)Oc1cccc([C@H](C)N(C)C)c1	2195.0

4929 rows × 3 columns

+ Code + Text

در ادامه ستون های **molecule_chembl_id, canonical_smiles, standard_value** را در یک فایل ذخیره میکنیم.

```
[29] pd.concat([df3,pd.Series(bioactivity_class)],axis=1)
```

	molecule_chembl_id	canonical_smiles	bioactivity_class	standard_value	0
0	CHEMBL133897	CCOc1nn(-c2cccc(OCc3ccccc3)c2)c(=O)o1	active	920.0	active
1	CHEMBL336398	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC1CC1	active	900.0	active
2	CHEMBL131588	CN(C(=O)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F)c1ccccc1	inactive	50000.0	inactive
3	CHEMBL130628	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F	active	1000.0	active
4	CHEMBL130478	CSc1nc(-c2ccc(OC(F)(F)F)cc2)nn1C(=O)N(C)C	active	200.0	active
...	...	...	...	...	...
4005	CHEMBL4848527	CC1=CC2Cc3nc4cc(Cl)ccc4c(NCCCCCCCCC(=O)NCc4cc[...	active	75.8	active
4006	CHEMBL4872514	COc1cc(CNC(=O)CCCCCCCCNc2c3c(nc4cc(Cl)ccc24)CC...	active	88.5	active
4007	CHEMBL3304306	CC1=CC2Cc3nc4cc(Cl)ccc4c(NCCCCCCCCCNC(=O)c4cc(...	active	620.0	active
4008	CHEMBL140476	CC1=CC2Cc3nc4cc(Cl)ccc4c(N)c3C(C1)C2	active	181.0	active
4009	CHEMBL502	COc1cc2c(cc1OC)C(=O)C(CC1CCN(Cc3ccccc3)CC1)C2	intermediate	6980.0	intermediate

4010 rows × 5 columns

Saves dataframe to CSV file

```
[ ] df3.to_csv('/content/drive/MyDrive/DrugDiscovery/Data/Alzheimer/cholinesterase_0/cholinesterase__data_preprocessed.csv', index=False)
```

برای مرحله دوم **conda** و **rfkit** را نصب میکنیم.

```
[1] from google.colab import drive
    drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

Install conda and rdkit

```
! wget https://repo.anaconda.com/miniconda/Miniconda3-py37_4.8.2-Linux-x86_64.sh
! chmod +x Miniconda3-py37_4.8.2-Linux-x86_64.sh
! bash ./Miniconda3-py37_4.8.2-Linux-x86_64.sh -b -f -p /usr/local
! conda install -c rdkit rdkit -y
import sys
sys.path.append('/usr/local/lib/python3.7/site-packages/')
```

```
    - wheel==0.34.2=py37_0
    - xz==5.2.4=h14c3975_4
    - yaml==0.1.7=had09818_2
    - zlib==1.2.11=h7b6447c_3


The following NEW packages will be INSTALLED:

  _libgcc_mutex      pkgs/main/linux-64::_libgcc_mutex-0.1-main
  asn1crypto         pkgs/main/linux-64::asn1crypto-1.3.0-py37_0
  ca-certificates    pkgs/main/linux-64::ca-certificates-2020.1.1-0
  certifi            pkgs/main/linux-64::certifi-2019.11.28-py37_0
  cffi               pkgs/main/linux-64::cffi-1.14.0-py37h2e261b9_0
  chardet            pkgs/main/linux-64::chardet-3.0.4-py37_1003
  conda              pkgs/main/linux-64::conda-4.8.2-py37_0
  conda-package-han~ pkgs/main/linux-64::conda-package-handling-1.6.0-py37h7b6447c_0
  cryptography       pkgs/main/linux-64::cryptography-2.8-py37h1ba5d50_0
  idna               pkgs/main/linux-64::idna-2.8-py37_0
  ld_impl_linux-64   pkgs/main/linux-64::ld_impl_linux-64-2.33.1-h53a641e_7
  libedit            pkgs/main/linux-64::libedit-3.1.20181209-hc058e9b_0
  libffi             pkgs/main/linux-64::libffi-3.2.1-hd88cf55_4
  libgcc-ng          pkgs/main/linux-64::libgcc-ng-9.1.0-hdf63c60_0
  libstdcxx-ng       pkgs/main/linux-64::libstdcxx-ng-9.1.0-hdf63c60_0
```

سپس دیتایی را که در مراحل قبل پیش پردازش کردیم فراخوانی میکنیم.

Load bioactivity data

```
[3] import pandas as pd
```

```
df = pd.read_csv('/content/drive/MyDrive/DrugDiscovery/Data/Alzheimer/cholinesterase_0/cholinesterase__data_preprocessed.csv')
```

در این مرحله پکیج **Lipinski, descriptors** را نصب میکنیم. کتابخانه های آن را نصب میکنیم.

```
[5] import numpy as np
    from rdkit import Chem
    from rdkit.Chem import Descriptors, Lipinski
```

Calculate descriptors

```
# Inspired by: https://codeocean.com/explore/capsules?query=tag:data-curation

def lipinski(smiles, verbose=False):

    moldata= []
    for elem in smiles:
        mol=Chem.MolFromSmiles(elem)
        moldata.append(mol)

    baseData= np.arange(1,1)
    i=0
    for mol in moldata:

        desc_MolWt = Descriptors.MolWt(mol)
        desc_MolLogP = Descriptors.MolLogP(mol)
        desc_NumHDonors = Lipinski.NumHDonors(mol)
        desc_NumHAcceptors = Lipinski.NumHAcceptors(mol)

        row = np.array([desc_MolWt,
                        desc_MolLogP,
                        desc_NumHDonors,
                        desc_NumHAcceptors])

        if(i==0):
            baseData=row
        else:
            baseData=np.vstack([baseData, row])
        i=i+1

    columnNames=["MW","LogP","NumHDonors","NumHAcceptors"]
    descriptors = pd.DataFrame(data=baseData,columns=columnNames)

    return descriptors
```

تابع **Lipinski** را فراخوانی و دیتا بیواکتیویتی را به عنوان ورودی به تابع می دهیم. سپس تابع مقادیر "MW","LogP","NumHDonors","NumHAcceptors" را برای ما محاسبه می کند.

Combine DataFrames

Let's take a look at the 2 DataFrames that will be combined.

```
df_lipinski
```

	MW	LogP	NumHDonors	NumHAcceptors
0	312.325	2.8032	0.0	6.0
1	376.913	4.5546	0.0	5.0
2	426.851	5.3574	0.0	5.0
3	404.845	4.7069	0.0	5.0
4	346.334	3.0953	0.0	6.0
...	...	...	...	...
4005	547.143	7.0315	3.0	4.0
4006	561.170	7.7468	2.0	5.0
4007	692.256	8.6434	4.0	7.0
4008	284.790	4.4664	1.0	2.0
4009	379.500	4.3611	0.0	4.0

4010 rows × 4 columns

در ادامه مقادیر محاسبه شده را به مقادیر قبلی دیتافریم الحاق می کنیم.

Now, let's combine the 2 DataFrame

```
[9] df_combined = pd.concat([df,df_lipinski], axis=1)
```

```
df_combined
```

	molecule_chembl_id	canonical_smiles	bioactivity_class	standard_value	MW	LogP	NumHDonors	NumHAcceptors
0	CHEMBL133897	CCOc1nn(-c2cccc(OCc3ccccc3)c2)c(=O)o1	active	920.0	312.325	2.8032	0.0	6.0
1	CHEMBL336398	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC1CC1	active	900.0	376.913	4.5546	0.0	5.0
2	CHEMBL131588	CN(C(=O)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F)c1ccccc1	inactive	50000.0	426.851	5.3574	0.0	5.0
3	CHEMBL130628	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F	active	1000.0	404.845	4.7069	0.0	5.0
4	CHEMBL130478	CSc1nc(-c2ccc(OC(F)(F)F)cc2)nn1C(=O)N(C)C	active	200.0	346.334	3.0953	0.0	6.0
...	...	...	...	...	...	...	...	...
4005	CHEMBL4848527	CC1=CC2Cc3nc4cc(Cl)ccc4c(NCCCCCCCCC(=O)NCc4cc[...	active	75.8	547.143	7.0315	3.0	4.0
4006	CHEMBL4872514	COc1cc(CNC(=O)CCCCCCCCNc2c3c(nc4cc(Cl)ccc24)CC...	active	88.5	561.170	7.7468	2.0	5.0
4007	CHEMBL3304306	CC1=CC2Cc3nc4cc(Cl)ccc4c(NCCCCCCCCCNC(=O)c4cc(...	active	620.0	692.256	8.6434	4.0	7.0
4008	CHEMBL140476	CC1=CC2Cc3nc4cc(Cl)ccc4c(N)c3C(C1)C2	active	181.0	284.790	4.4664	1.0	2.0
4009	CHEMBL502	COc1cc2c(cc1OC)C(=O)C(CC1CCN(Cc3ccccc3)CC1)C2	intermediate	6980.0	379.500	4.3611	0.0	4.0

4010 rows × 8 columns

در این مرحله **IC50** را محاسبه و **pIC50** را از روی آن بدست می آوریم.

Convert IC50 to pIC50

To allow **IC50** data to be more uniformly distributed, we will convert **IC50** to the negative logarithmic scale which is essentially **-log10(IC50)**.

This custom function pIC50() will accept a DataFrame as input and will:

- Take the IC50 values from the `standard_value` column and converts it from nM to M by multiplying the value by 10^{-9}
- Take the molar value and apply -log10
- Delete the `standard_value` column and create a new `pIC50` column

```
[ ] # https://github.com/chaninlab/estrogen-receptor-alpha-qsar/blob/master/02_ER_alpha_RO5.ipynb

    import numpy as np

    def pIC50(input):
        pIC50 = []

        for i in input['standard_value_norm']:
            molar = i*(10**-9) # Converts nM to M
            pIC50.append(-np.log10(molar))

        input['pIC50'] = pIC50
        x = input.drop('standard_value_norm', 1)

        return x
```

Point to note: Values greater than 100,000,000 will be fixed at 100,000,000 otherwise the negative logarithmic value will become negative.

+ Code + Text

بعد از محاسبه **pIC50** آن را نیز به دیتا الحاق می کنیم و در پایان دیتا را ذخیره می کنیم.

```
[19] df_final = pIC50(df_norm)
     df_final
```

```
<ipython-input-11-bf09df1ccbf9>:10: RuntimeWarning: divide by zero encountered in log10
  pIC50.append(-np.log10(molar))
<ipython-input-11-bf09df1ccbf9>:13: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only
  x = input.drop('standard_value_norm', 1)
```

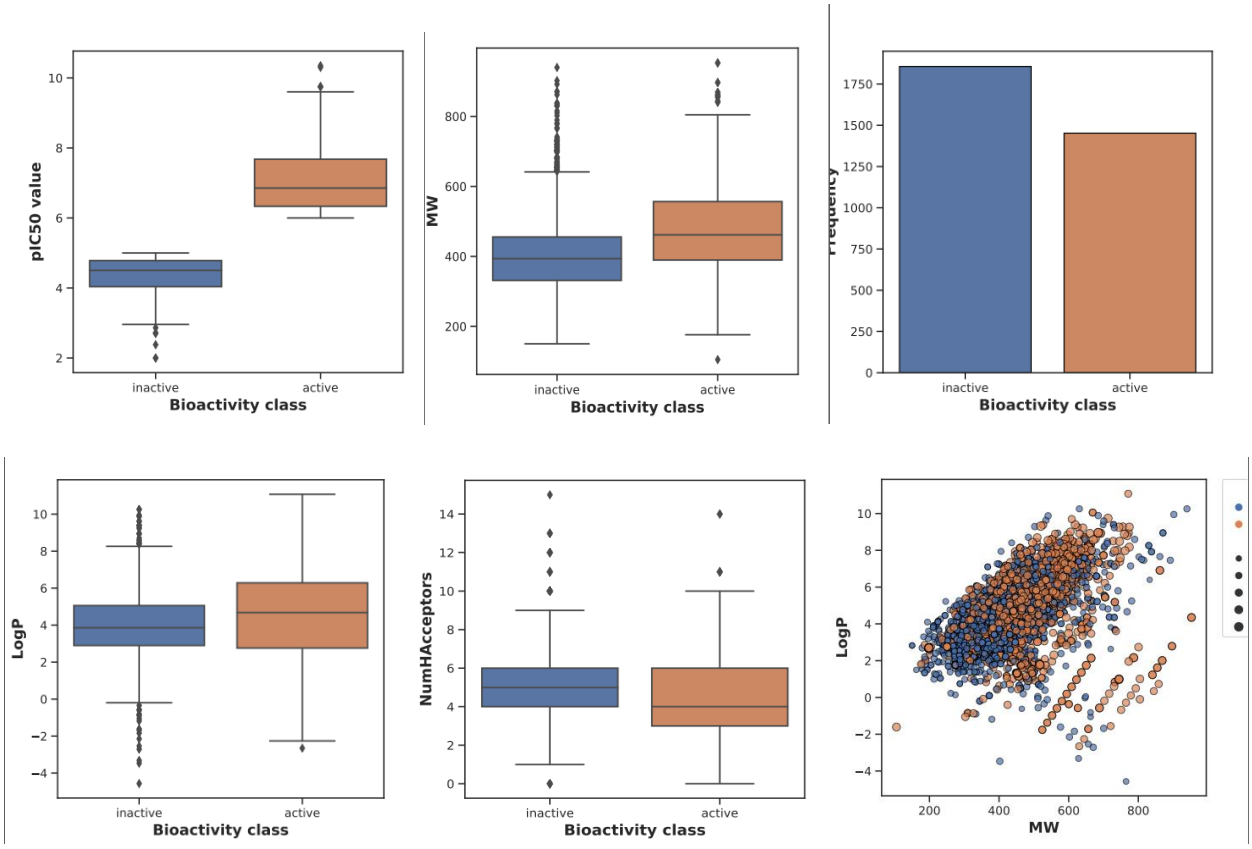
	molecule_chembl_id	canonical_smiles	bioactivity_class	MW	LogP	NumHDonors	NumHAcceptors	pIC50
0	CHEMBL133897	CCOc1nn(-c2cccc(OCc3ccccc3)c2)c(=O)o1	active	312.325	2.8032	0.0	6.0	6.036212
1	CHEMBL336398	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC1CC1	active	376.913	4.5546	0.0	5.0	6.045757
2	CHEMBL131588	CN(C(=O)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F)c1ccccc1	inactive	426.851	5.3574	0.0	5.0	4.301030
3	CHEMBL130628	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F	active	404.845	4.7069	0.0	5.0	6.000000
4	CHEMBL130478	CSc1nc(-c2ccc(OC(F)(F)F)cc2)nn1C(=O)N(C)C	active	346.334	3.0953	0.0	6.0	6.698970
...	...	...	...	...	...	...	...	...
4005	CHEMBL4848527	CC1=CC2Cc3nc4cc(Cl)ccc4c(NCCCCCCCCC(=O)NCc4cc[...	active	547.143	7.0315	3.0	4.0	7.120331
4006	CHEMBL4872514	COc1cc(CNC(=O)CCCCCCCCNc2c3c(nc4cc(Cl)ccc24)CC...	active	561.170	7.7468	2.0	5.0	7.053057
4007	CHEMBL3304306	CC1=CC2Cc3nc4cc(Cl)ccc4c(NCCCCCCCCCNC(=O)c4cc(...	active	692.256	8.6434	4.0	7.0	6.207608
4008	CHEMBL140476	CC1=CC2Cc3nc4cc(Cl)ccc4c(N)c3C(C1)C2	active	284.790	4.4664	1.0	2.0	6.742321
4009	CHEMBL502	COc1cc2c(cc1OC)C(=O)C(CC1CCN(Cc3ccccc3)CC1)C2	intermediate	379.500	4.3611	0.0	4.0	5.156145

4010 rows × 8 columns

```
df_final.pIC50.describe()
```

```
count    4010.000000
mean             inf
std              NaN
min        -7.640000
25%         4.697237
50%         5.494850
75%         6.721589
max              inf
Name: pIC50, dtype: float64
```

در ادامه برای هر یک از مقادیر بدست آمده پلات های زیر را رسم میکنیم.



خروجی این مرحله را یکبار به صورت دیتای بیواکتیویتای که مقدار **standard_value** آن نرمال شده تا سقف ۱۰۰ میلیون و با در نظر گرفتن کلاس **intermediate** و یکبار بدون نرمال کردن و بدون کلاس **intermediate** ذخیره میکنیم.

برای مرحله سوم دیسکریپتور **padel.zip** را از گیت هاب دانلود و از حالت فشرده خارج میکنیم.

Download PaDEL-Descriptor

```
! wget https://github.com/dataprofessor/bioinformatics/raw/master/padel.zip
! wget https://github.com/dataprofessor/bioinformatics/raw/master/padel.sh
```

```
--2023-02-06 12:04:36--  https://github.com/dataprofessor/bioinformatics/raw/master/padel.zip
Resolving github.com (github.com)... 140.82.114.4
Connecting to github.com (github.com)|140.82.114.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://raw.githubusercontent.com/dataprofessor/bioinformatics/master/padel.zip [following]
--2023-02-06 12:04:36--  https://raw.githubusercontent.com/dataprofessor/bioinformatics/master/padel.zip
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 25768637 (25M) [application/zip]
Saving to: ‘padel.zip’

padel.zip           100%[===================>]  24.57M   155MB/s    in 0.2s

2023-02-06 12:04:37 (155 MB/s) - ‘padel.zip’ saved [25768637/25768637]

--2023-02-06 12:04:37--  https://github.com/dataprofessor/bioinformatics/raw/master/padel.sh
Resolving github.com (github.com)... 140.82.114.3
Connecting to github.com (github.com)|140.82.114.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://raw.githubusercontent.com/dataprofessor/bioinformatics/master/padel.sh [following]
--2023-02-06 12:04:37--  https://raw.githubusercontent.com/dataprofessor/bioinformatics/master/padel.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 231 [text/plain]
Saving to: ‘padel.sh’

padel.sh            100%[===================>]     231  --.-KB/s    in 0s

2023-02-06 12:04:37 (11.6 MB/s) - ‘padel.sh’ saved [231/231]
```

```
! unzip padel.zip
```

در ادامه دیتای بیواکتیویتی خروجی مرحله قبل را که بدون نرمال کردن مقدار **standard_value** تا سقف ۱۰۰ میلیون و بدون کلاس **intermediate** ذخیره کرده بودیم را لود میکنیم.

Load bioactivity data

Download the curated ChEMBL bioactivity data that has been pre-processed from Parts 1 and 2 of this Bioinformatics Project series. Here we will be using the **bioactivity_data_3class_pIC50.csv** file that essentially contain the pIC50 values that we will be using for building a regression model.

```
[5] import pandas as pd
```

```
[7] #df3 = pd.read_csv('/content/drive/MyDrive/DrugDiscovery/Data/Alzheimer/cholinesterase_0/cholinesterase__data_2class_pIC50.csv')
    df3 = pd.read_csv('/content/drive/MyDrive/DrugDiscovery/Data/Alzheimer/cholinesterase_0/cholinesterase__data_2class_pIC50_withoutnorm_withintermediate.csv')
```

```
df3
```

	Unnamed: 0	molecule_chembl_id	canonical_smiles	bioactivity_class	MW	LogP	NumHDonors	NumHAcceptors	pIC50
0	0	CHEMBL133897	CCOc1nn(-c2cccc(OCc3ccccc3)c2)c(=O)o1	active	312.325	2.8032	0.0	6.0	6.036212
1	1	CHEMBL336398	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC1CC1	active	376.913	4.5546	0.0	5.0	6.045757
2	2	CHEMBL131588	CN(C(=O)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F)c1ccccc1	inactive	426.851	5.3574	0.0	5.0	4.301030
3	3	CHEMBL130628	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F	active	404.845	4.7069	0.0	5.0	6.000000
4	4	CHEMBL130478	CSc1nc(-c2ccc(OC(F)(F)F)cc2)nn1C(=O)N(C)C	active	346.334	3.0953	0.0	6.0	6.698970
...	...	...	...	...	...	...	...	...	...
4005	4005	CHEMBL4848527	CC1=CC2Cc3nc4cc(Cl)ccc4c(NCCCCCCCCC(=O)NCc4cc[...	active	547.143	7.0315	3.0	4.0	7.120331
4006	4006	CHEMBL4872514	COc1cc(CNC(=O)CCCCCCCCNc2c3c(nc4cc(Cl)ccc24)CC...	active	561.170	7.7468	2.0	5.0	7.053057
4007	4007	CHEMBL3304306	CC1=CC2Cc3nc4cc(Cl)ccc4c(NCCCCCCCCCNC(=O)c4cc(...	active	692.256	8.6434	4.0	7.0	6.207608
4008	4008	CHEMBL140476	CC1=CC2Cc3nc4cc(Cl)ccc4c(N)c3C(C1)C2	active	284.790	4.4664	1.0	2.0	6.742321
4009	4009	CHEMBL502	COc1cc2c(cc1OC)C(=O)C(CC1CCN(Cc3ccccc3)CC1)C2	intermediate	379.500	4.3611	0.0	4.0	5.156145

در مرحله بعد ستون های **'canonical_smiles','molecule_chembl_id'** را در فایل **molecule.smi** ذخیره می کنیم. سپس با استفاده از دیسکریپتور **padel**، برای دیتا **Fingerprint** میسازیم.

```
[9] selection = ['canonical_smiles','molecule_chembl_id']
    df3_selection = df3[selection]
    df3_selection.to_csv('molecule.smi', sep='\t', index=False, header=False)
```

```
[10] ! cat molecule.smi | head -5

CCOc1nn(-c2cccc(OCc3ccccc3)c2)c(=O)o1   CHEMBL133897
O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC1CC1      CHEMBL336398
CN(C(=O)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F)c1ccccc1       CHEMBL131588
O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F   CHEMBL130628
CSc1nc(-c2ccc(OC(F)(F)F)cc2)nn1C(=O)N(C)C       CHEMBL130478
```

```
[11] ! cat molecule.smi | wc -l

4010
```

Calculate fingerprint descriptors

Calculate PaDEL descriptors

```
[12] ! cat padel.sh

java -Xms1G -Xmx1G -Djava.awt.headless=true -jar ./PaDEL-Descriptor/PaDEL-Descriptor.jar -removesalt -standardizenitro -fingerprints -descriptortypes .
```

```
! bash padel.sh

Processing CHEMBL336398 in molecule.smi (2/4010).
Processing CHEMBL133897 in molecule.smi (1/4010).
Processing CHEMBL131588 in molecule.smi (3/4010). Average speed: 8.02 s/mol.
Processing CHEMBL130628 in molecule.smi (4/4010). Average speed: 4.10 s/mol.
Processing CHEMBL130478 in molecule.smi (5/4010). Average speed: 2.98 s/mol.
Processing CHEMBL130098 in molecule.smi (6/4010). Average speed: 2.32 s/mol.
Processing CHEMBL337486 in molecule.smi (7/4010). Average speed: 1.89 s/mol.
Processing CHEMBL336538 in molecule.smi (8/4010). Average speed: 1.64 s/mol.
Processing CHEMBL131051 in molecule.smi (9/4010). Average speed: 1.71 s/mol.
```

خروجی فایل دیسکریپتور شامل ۸۸۱ ستون مقدار **Fingerprint** به صورت زیر می باشد.

Preparing the X and Y Data Matrices

X data matrix

```
[16] df3_X = pd.read_csv('/content/drive/MyDrive/DrugDiscovery/Data/Alzheimer/cholinesterase_0/without_desc/descriptors_output.csv')
```

```
df3_X
```

	Name	PubchemFP0	PubchemFP1	PubchemFP2	PubchemFP3	PubchemFP4	PubchemFP5	PubchemFP6	PubchemFP7	PubchemFP8	...	PubchemFP871	PubchemFP872	PubchemFP873	PubchemFP874	PubchemFP875	Pubch
0	CHEMBL336398	1	1	1	0	0	0	0	0	0	...	0	0	0	0	0	
1	CHEMBL133897	1	1	1	0	0	0	0	0	0	...	0	0	0	0	0	
2	CHEMBL130628	1	1	1	0	0	0	0	0	0	...	0	0	0	0	0	
3	CHEMBL131588	1	1	0	0	0	0	0	0	0	...	0	0	0	0	0	
4	CHEMBL130478	1	1	0	0	0	0	0	0	0	...	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
4005	CHEMBL4848527	1	1	1	1	0	0	0	0	0	...	0	0	0	0	0	
4006	CHEMBL4872514	1	1	1	1	0	0	0	0	0	...	0	0	0	0	0	
4007	CHEMBL140476	1	1	1	0	0	0	0	0	0	...	0	0	0	0	0	
4008	CHEMBL502	1	1	1	0	0	0	0	0	0	...	0	0	0	0	0	
4009	CHEMBL3304306	1	1	1	1	0	0	0	0	0	...	0	0	0	0	0	

4010 rows × 882 columns

ستون **pIC50** را به ستون آخر فایل دیسکریپتور الحاق کرده و فایل جدید را برای مراحل بعدی ذخیره می کنیم.

در این مرحله دیتایی که از مرحله قبل ذخیره کردیم را لود می کنیم. دیتا دارای ۸۸۱ فیچر و یک تارگت که **'pIC50'** میباشد.

```
[3] df = pd.read_csv('/content/drive/MyDrive/DrugDiscovery/Data/Alzheimer/cholinesterase_0/cholinesterase_data_3class_pIC50_pubchem_fp___without___.csv')
```

3. Input features

The ***Acetylcholinesterase*** data set contains 881 input features and 1 output variable (pIC50 values).

3.1. Input features

```
X = df.drop('pIC50', axis=1)
X
```

	PubchemFP0	PubchemFP1	PubchemFP2	PubchemFP3	PubchemFP4	PubchemFP5	PubchemFP6	PubchemFP7	PubchemFP8	PubchemFP9	...	PubchemFP871	PubchemFP872	PubchemFP873	Pubc
0	1	1	1	0	0	0	0	0	0	1	...	0	0	0	
1	1	1	1	0	0	0	0	0	0	1	...	0	0	0	
2	1	1	1	0	0	0	0	0	0	1	...	0	0	0	
3	1	1	0	0	0	0	0	0	0	1	...	0	0	0	
4	1	1	0	0	0	0	0	0	0	1	...	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
005	1	1	1	1	0	0	0	0	0	1	...	0	0	0	
006	1	1	1	1	0	0	0	0	0	1	...	0	0	0	
007	1	1	1	0	0	0	0	0	0	1	...	0	0	0	
008	1	1	1	0	0	0	0	0	0	1	...	0	0	0	
009	1	1	1	1	0	0	0	0	0	1	...	0	0	0	

10 rows × 881 columns

در این مرحله دیتا را به دو بخش **train , test** تقسیم میکنیم. سپس یک مدل رگرسیون با استفاده از الگوریتم **Random Forest** میسازیم. همانطور که مشاهده میکنید دقت **r2** مدل روی داده تست **0.62** درصد می باشد.

4. Data split (80/20 ratio)

```
[33] X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2)
```

```
[34] X_train.shape, Y_train.shape
```

```
((2646, 144), (2646,))
```

```
[35] X_test.shape, Y_test.shape
```

```
((662, 144), (662,))
```

5. Building a Regression Model using Random Forest

```
[36] import numpy as np

     np.random.seed(100)
     model = RandomForestRegressor(n_estimators=60)
     model.fit(X_train, Y_train)
     r2 = model.score(X_test, Y_test)
     r2
```

```
0.626456574929644
```

```
[37] Y_pred = model.predict(X_test)
     Y_pred
```

```
array([6.53033127, 4.25475723, 6.76937115, 6.0320372 , 4.89024359,
       5.81764296, 4.30357721, 5.023344  , 5.19858911, 6.77392121,
       8.23140094, 4.13520265, 4.01467269, 4.24649655, 6.76937115,
       4.64991627, 4.39072278, 4.49163842, 7.6039684 , 3.89928215,
       3.73339275, 3.93072835, 4.50506563, 8.07533356, 4.65033119,
       4.68544291, 6.22625956, 4.98485942, 4.58101165, 6.78533063,
       4.64839144, 4.46798635, 4.68332755, 6.04421056, 5.08791022,
       4.86030494, 4.62035859, 5.78182302, 4.03433346, 7.51447024,
       7.44127988, 4.35671262, 4.8477565 , 4.78337461, 4.45303889,
       5.65139271, 7.16609915, 7.44517972, 4.25908435, 4.06190698,
```

Scatter plot pic50 را برای مقدار پیش بینی شده و مقدار **experimental** محاسبه میکنیم.

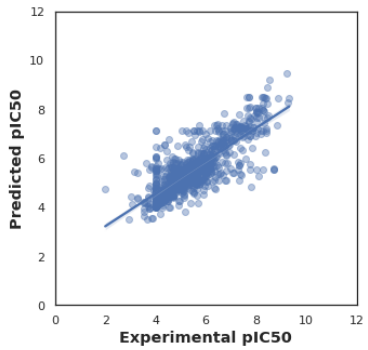
6. Scatter Plot of Experimental vs Predicted pIC50 Values

```
import seaborn as sns
import matplotlib.pyplot as plt

sns.set(color_codes=True)
sns.set_style("white")

ax = sns.regplot(Y_test, Y_pred, scatter_kws={'alpha':0.4})
ax.set_xlabel('Experimental pIC50', fontsize='large', fontweight='bold')
ax.set_ylabel('Predicted pIC50', fontsize='large', fontweight='bold')
ax.set_xlim(0, 12)
ax.set_ylim(0, 12)
ax.figure.set_size_inches(5, 5)
plt.show
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variables as keyword args: x,
  warnings.warn(
<function matplotlib.pyplot.show(*args, **kw)>
```



یک بار دیگر دیتای بیواکتیویتی خروجی مرحله قبل را که مقدار **standard_value** تا سقف ۱۰۰ میلیون نرمال شده بود و کلاس **intermediate** آن حذف شده بود را لود میکنیم. مدل رگرسیونی را مجدد برای این دیتا میسازیم و خروجی مدل دقت**r2= 0.63** درصد را گزارش میکند.

5. Building a Regression Model using Random Forest

```
model = RandomForestRegressor(n_estimators=100)
model.fit(X_train, Y_train)
r2 = model.score(X_test, Y_test)
r2
```

```
0.6324574315953669
```

```
Y_pred = model.predict(X_test)
```

```
[16] Y_pred
```

```
4.43851787, 9.4561642 , 4.04711433, 6.39929354, 4.70094886,
6.59469789, 5.9606147 , 7.45488441, 6.62001244, 4.59710464,
7.83882338, 6.00138703, 4.03996259, 5.30582911, 4.74395215,
4.18000253, 4.5985925 , 6.71375363, 4.43785069, 5.08285218,
4.80691788, 5.37872283, 4.93015719, 6.38951077, 4.70396967,
4.59621473, 7.89456801, 7.06369808, 4.71532935, 4.46228961,
4.40925237, 6.62001244, 4.62169581, 6.9464217 , 6.20819375,
6.23564   , 6.06973687, 5.5929828 , 7.65292603, 6.62001244,
4.4831534 , 5.01655894, 4.98860097, 4.82332013, 4.72966428,
5.08285218, 6.12942443, 4.61010301, 6.34561641, 4.75604132,
4.38574271, 5.00066342, 8.36564889, 7.04201384, 8.01228211,
4.15910915, 7.74507647, 7.57657433, 7.19090984, 7.41792585,
4.66409878, 8.14829036, 5.51841897, 4.96510782, 4.68044645,
5.77295675, 4.14874467, 4.34617775, 4.03386641, 4.88266196,
```

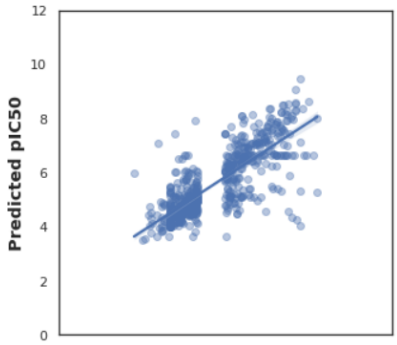
+ Code + Text

```
[17] import seaborn as sns
import matplotlib.pyplot as plt

sns.set(color_codes=True)
sns.set_style("white")

ax = sns.regplot(Y_test, Y_pred, scatter_kws={'alpha':0.4})
ax.set_xlabel('Experimental pIC50', fontsize='large', fontweight='bold')
ax.set_ylabel('Predicted pIC50', fontsize='large', fontweight='bold')
ax.set_xlim(0, 12)
ax.set_ylim(0, 12)
ax.figure.set_size_inches(5, 5)
plt.show
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variables a
  warnings.warn(
<function matplotlib.pyplot.show(*args, **kw)>
```



منابع

/https://www.ebi.ac.uk/chembl

https://m.youtube.com/c/DataProfessor