Part-1: Data Collection and Preprocessing

Pre-requisite

CHEMBL DATABASE:

The ChEMBL is a database of bioactive drug-like small molecules, it contains 2-D structures, calculated properties (e.g. logP, Molecular Weight, Lipinski Parameters, etc.) and abstracted bioactivities (e.g. binding constants, pharmacology and ADMET data). It consists of more than 2.1 million compounds compiled from around 80000 documents, 1.4 million assays with spanning 2000 cells, 14000 targets and 38000 indicators.

Installing libraries

```
# Installing the CHEMBL Web Package Service to retrieve data from CHEMBL Database
! pip install chembl_webresource_client
```

```
Collecting chembl_webresource_client
  Downloading https://files.pythonhosted.org/packages/c3/75/ccfc66e213d685c623d74609d11
     |████████████████████████████████| 61kB 3.1MB/s
Collecting requests-cache>=0.4.7
  Downloading https://files.pythonhosted.org/packages/3c/b7/ece6951b3ca140c3ff403d4e2aa
Requirement already satisfied: requests>=2.18.4 in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: urllib3 in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: easydict in /usr/local/lib/python3.7/dist-packages (from
Collecting url-normalize>=1.4
  Downloading https://files.pythonhosted.org/packages/65/1c/6c6f408be78692fc850006a2b6d
Requirement already satisfied: itsdangerous in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from url-
Installing collected packages: url-normalize, requests-cache, chembl-webresource-client
Successfully installed chembl-webresource-client-0.10.3 requests-cache-0.6.3 url-normal
```

Importing Libraries

```
# Importing the necessary libraries
import pandas as pd
from chembl_webresource_client.new_client import new_client
```

Data Collection

Searching for the target protein

```
# Target Search for alzheimer's disease
target = new_client.target
target_query = target.search('acetylcholinesterase')
targets = pd.DataFrame.from_dict(target_query)
targets[:10]
```

	cross_references	organism	pref_name	score	species_group_flag	target_
0	[{'xref_id': 'P22303', 'xref_name': None, 'xre...	Homo sapiens	Acetylcholinesterase	27.0	False	CI
1	[]	Homo sapiens	Cholinesterases; ACHE & BCHE	27.0	False	CHEMI
2	[]	Drosophila melanogaster	Acetylcholinesterase	17.0	False	CHEMI
3	[{'xref_id': 'P04058', 'xref_name': None, 'xre...	Torpedo californica	Acetylcholinesterase	15.0	False	CH
4	[{'xref_id': 'P21836', 'xref_name': None, 'xre...	Mus musculus	Acetylcholinesterase	15.0	False	CH
5	[{'xref_id': 'P37136', 'xref_name': None, 'xre...	Rattus norvegicus	Acetylcholinesterase	15.0	False	CH
6	[{'xref_id': 'O42275', 'xref_name': None, 'xre...	Electrophorus electricus	Acetylcholinesterase	15.0	False	CH
7	[{'xref_id': 'P23795', 'xref_name': None, 'xre...	Bos taurus	Acetylcholinesterase	15.0	False	CH

```
targets.shape
```

```
(24, 9)
```

Select and Retrieve Data with target_type == SINGLE PROTEIN and standard_type == IC_{50} measured in nM(nano molar)

Formely known as PROTEIN in the target type, it is now subdivded into a number of categories:

- In the simple case where a compound is believed to interact specifically with a monomeric protein, the target type ‘SINGLE PROTEIN’ is now used.
- In cases where either a compound is known to act non-specifically with all members of a protein family, or the assay conditions are such that it is not possible to determine which member(s) of a protein family the compound is acting on (e.g. a cell-based or tissue-based assay), a target type of ‘PROTEIN FAMILY’ is used.
- If the molecular entity of a compound is known as protein complex and can be precisely defined; with which the our compound interacts, the target type ‘PROTEIN COMPLEX’ is used.
- In a tissue-based format, the exact subunit combinations present are generally not known. In such cases, the target type of ‘PROTEIN COMPLEX GROUP’ is used.
- Other new target types have also been created for approved drugs whose molecular targets are not proteins, e.g. metals.

```
# Assigning the selected target as entry 1
selected_target = targets.target_chembl_id[0]
selected_target
```

```
'CHEMBL220'
```

When identifying compounds that bind to a particular protein target for structure–activity relationship or lead identification studies, it is important to be using comparable data. Thus, for this analysis we are using the standardized value of IC_{50} with the units as nM, i.e. nano molars.

```
# Retriving the bioactivity data with target_type as SINGLE PROTEIN as well as standard type
activity = new_client.activity
result = activity.filter(target_chembl_id=selected_target).filter(standard_type="IC50")
```

```
# Converting dictionary to dataframe
df = pd.DataFrame.from_dict(result)
df.head(3)
```

	activity_comment	activity_id	activity_properties	assay_chembl_id	assay_descripti
0	None	33969	[]	CHEMBL643384	Inhibito concentrati agai acetylcholine
1	None	37563	[]	CHEMBL643384	Inhibito concentrati agai acetylcholine
2	None	37565	[]	CHEMBL643384	Inhibito concentrati agai

```
# Getting the shape of the dataframe created
df.shape
```

```
(7479, 45)
```

```
# Saving the dataframe to a csv file
df.to_csv('bioactivity_data_raw.csv',index=False)
```

Handling Missing Data

```
# Dropping the compounds having missing values for the standard_type and canonical_smiles
df1 = df[df.standard_value.notna()]
df2 = df1[df1.canonical_smiles.notna()]
df2
```

	activity_comment	activity_id	activity_properties	assay_chembl_id	assay_descri
0	None	33969	[]	CHEMBL643384	Inh concen a acetylch
1	None	37563	[]	CHEMBL643384	Inh concen a acetylch
2	None	37565	[]	CHEMBL643384	Inh concen a acetylch
3	None	38902	[]	CHEMBL643384	Inh concen a acetylch
4	None	41170	[]	CHEMBL643384	Inh concen a acetylch
...	...	...	...	...	
					Inhibition of h

```
# Dropping the duplicates
df3 = df2.drop_duplicates(['canonical_smiles'])
df3
```

	activity_comment	activity_id	activity_properties	assay_chembl_id	assay_descri
0	None	33969	[]	CHEMBL643384	Inh concen a acetylch
1	None	37563	[]	CHEMBL643384	Inh concen a acetylch
2	None	37565	[]	CHEMBL643384	Inh concen a acetylch
3	None	38902	[]	CHEMBL643384	Inh concen a acetylch
4	None	41170	[]	CHEMBL643384	Inh concen a acetylch
...	...	...	...	...	

Data Pre-preprocessing

Inhibition of h

Labeling compounds as active, intermediate or inactive.

The bioactivity data is in the IC50 unit. Compounds having values of less than 1000 nM will be considered to be **active** while those greater than 10,000 nM will be considered to be **inactive**. As for those values in between 1,000 and 10,000 nM will be referred to as **intermediate**.

7477 None 19487204 [] CHEMBL4481753 (unknown

```
# Classifying compounds by labeling into active, intermediate and inactive
bioactivity_class = []
for i in df3.standard_value:
  if float(i) >= 10000.0:
    bioactivity_class.append('inactive')
  elif float(i) <= 1000.0:
    bioactivity_class.append('active')
  else:
    bioactivity_class.append('intermediate')
```

Combining the columns molecule_chembl_id,canonical_smiles,standard_value and bioactivity_class into a dataframe

```
# Combining the columns molecule_chembl_id,canonical_smiles,standard_value
columns = ['molecule_chembl_id','canonical_smiles','standard_value']
df4 = df3[columns]
df4[10:50]
```

	molecule_chembl_id	canonical_smiles	standar
10	CHEMBL341437	CCSc1nc(-c2ccc(OC)cc2)nn1C(=O)N1CCOCC1	
11	CHEMBL335033	CSc1nc(-c2ccc3ccccc3c2)nn1C(=O)N(C)C	
12	CHEMBL122983	C[C@H]1C(=O)N(C(=O)NCc2ccccc2)[C@@H]1Oc1ccc(C(...	
13	CHEMBL338720	CSc1nc(-c2ccc(-c3ccccc3)cc2)nn1C(=O)N(C)C	
14	CHEMBL339995	CSc1nc(/C=C/c2ccccc2)nn1C(=O)N(C)C	
15	CHEMBL335158	CCCCCCSc1nc(-c2ccc(Cl)cc2)nn1C(=O)N1CCCCC1	
16	CHEMBL131536	CSc1nc(-c2ccc(Cl)cc2)nn1C(=O)N(C)c1ccccc1	
17	CHEMBL106126	Cc1c(C(C)C)c(=O)on1C(=O)N1CCC[C@H](C)C1	
18	CHEMBL334971	CCSc1nc(-c2ccc(OC)cc2)nn1C(=O)N(C)c1ccccc1	
19	CHEMBL336625	CCCCCCSc1nc(-c2ccc(C)cc2)nn1C(=O)N(C)c1ccccc1	
20	CHEMBL130666	CSc1nc(-c2ccc(Cl)cc2)nn1C(=O)N1CCCCC1	
21	CHEMBL134061	O=C(N1CCOCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F	
22	CHEMBL133388	CSc1nc(-c2ccc(C)cc2)nn1C(=O)N(C)C	
23	CHEMBL130645	CSc1nc(-c2ccc(Cl)cc2)nn1C(=O)N1CCOCC1	
25	CHEMBL133580	CCCCCCSc1nc(-c2ccc(Cl)cc2)nn1C(=O)N(C)c1ccccc1	
26	CHEMBL336524	CCSc1nc(-c2ccc(Cl)cc2)nn1C(=O)N(C)C	
27	CHEMBL336276	CCCCCCSc1nc(-c2ccc(C)cc2)nn1C(=O)N(C)C	

```
# Concatenating the bioactivity_class column
df4['class'] = bioactivity_class
df4
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
```

	molecule_chembl_id	canonical_smiles	standard_value
0	CHEMBL133897	CCOc1nn(-c2cccc(OCc3ccccc3)c2)c(=O)o1	750.0
1	CHEMBL336398	O=C(N1CCCCC1)n1nc(-	100.0

```
# Saving the dataframe to csv
df4.to_csv('bioactivity_data_preprocessed.csv',index=False)
```

Part-2: Descriptor Calculation and Exploratory Data Analysis

Pre-requisite

Installing libraries

7475	CHEMBL4570655	c2nc(NC(=O)C3CCN(Cc4ccccc4C)CC3)sc2-	10000.0	i

```
# Installing 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
```

```
    python-dateutil-2.8.1      |     pyhd3eb1b0_0         221 KB
    pytz-2021.1                |     pyhd3eb1b0_0         181 KB
    rdkit-2020.09.1.0          |   py37hd50e099_1        25.8 MB  rdkit
    xz-5.2.5                   |       h7b6447c_0         341 KB
    zstd-1.4.5                 |       h9ceee32_0         619 KB
    ------------------------------------------------------------
                                           Total:       226.3 MB

The following NEW packages will be INSTALLED:

  blas               pkgs/main/linux-64::blas-1.0-mkl
  bzip2              pkgs/main/linux-64::bzip2-1.0.8-h7b6447c_0
  cairo              pkgs/main/linux-64::cairo-1.16.0-hf32fb01_1
  fontconfig         pkgs/main/linux-64::fontconfig-2.13.1-h6c09931_0
  freetype           pkgs/main/linux-64::freetype-2.10.4-h5ab3b9f_0
  glib               pkgs/main/linux-64::glib-2.68.1-h36276a3_0
  icu                pkgs/main/linux-64::icu-58.2-he6710b0_3
  intel-openmp       pkgs/main/linux-64::intel-openmp-2021.2.0-h06a4308_610
  jpeg               pkgs/main/linux-64::jpeg-9b-h024ee3a_2
  lcms2              pkgs/main/linux-64::lcms2-2.12-h3be6417_0
  libboost           pkgs/main/linux-64::libboost-1.73.0-h3ff78a5_11
  libpng             pkgs/main/linux-64::libpng-1.6.37-hbc83047_0
  libtiff            pkgs/main/linux-64::libtiff-4.2.0-h85742a9_0
  libuuid            pkgs/main/linux-64::libuuid-1.0.3-h1bed415_2
  libwebp-base       pkgs/main/linux-64::libwebp-base-1.2.0-h27cfd23_0
  libxcb             pkgs/main/linux-64::libxcb-1.14-h7b6447c_0
  libxml2            pkgs/main/linux-64::libxml2-2.9.10-hb55368b_3
  lz4-c              pkgs/main/linux-64::lz4-c-1.9.3-h2531618_0
  mkl                pkgs/main/linux-64::mkl-2021.2.0-h06a4308_296
  mkl-service        pkgs/main/linux-64::mkl-service-2.3.0-py37h27cfd23_1
  mkl_fft            pkgs/main/linux-64::mkl_fft-1.3.0-py37h42c9631_2
  mkl_random         pkgs/main/linux-64::mkl_random-1.2.1-py37ha9443f7_2
  numpy              pkgs/main/linux-64::numpy-1.20.1-py37h93e21f0_0
  numpy-base         pkgs/main/linux-64::numpy-base-1.20.1-py37h7d8b39e_0
  olefile            pkgs/main/linux-64::olefile-0.46-py37_0
  pandas             pkgs/main/linux-64::pandas-1.2.4-py37h2531618_0
  pcre               pkgs/main/linux-64::pcre-8.44-he6710b0_0
  pillow             pkgs/main/linux-64::pillow-8.2.0-py37he98fc37_0
  pixman             pkgs/main/linux-64::pixman-0.40.0-h7b6447c_0
  py-boost           pkgs/main/linux-64::py-boost-1.73.0-py37ha9443f7_11
  python-dateutil    pkgs/main/noarch::python-dateutil-2.8.1-pyhd3eb1b0_0
  pytz               pkgs/main/noarch::pytz-2021.1-pyhd3eb1b0_0
  rdkit              rdkit/linux-64::rdkit-2020.09.1.0-py37hd50e099_1
  zstd               pkgs/main/linux-64::zstd-1.4.5-h9ceee32_0

The following packages will be UPDATED:

  ca-certificates                                2020.1.1-0 --> 2021.4.13-h06a4308_1
  certifi                                 2019.11.28-py37_0 --> 2020.12.5-py37h06a430
  conda                                        4.8.2-py37_0 --> 4.10.1-py37h06a4308_1
  libffi                                   3.2.1-hd88cf55_4 --> 3.3-he6710b0_2
  openssl                                 1.1.1d-h7b6447c_4 --> 1.1.1k-h27cfd23_0
  xz                                       5.2.4-h14c3975_4 --> 5.2.5-h7b6447c_0



Downloading and Extracting Packages
conda-4.10.1         | 2.9 MB    | : 100% 1.0/1 [00:00<00:00,  3.29s/it]
```

```
# Importing library and adding path
import sys
sys.path.append('/usr/local/lib/python3.7/site-packages/')
```

Loading Bioactivity Preprocessed Data

```
# Importing required library
import pandas as pd
```

```
# Reading the csv file into DataFrame
df = pd.read_csv('bioactivity_data_preprocessed.csv')
df
```

	molecule_chembl_id	canonical_smiles	standard_value	
0	CHEMBL133897	CCOc1nn(-c2cccc(OCc3ccccc3)c2)c(=O)o1	750.0	
1	CHEMBL336398	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC1CC1	100.0	
2	CHEMBL131588	CN(C(=O)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F)c1ccccc1	50000.0	i
3	CHEMBL130628	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F	300.0	
4	CHEMBL130478	CSc1nc(-c2ccc(OC(F)(F)F)cc2)nn1C(=O)N(C)C	800.0	
...	...	...	...	
5038	CHEMBL4554172	Cc1ccc(-c2nc(NC(=O)C3CCN(Cc4cccc(F)c4)CC3)sc2-...	10000.0	i
5039	CHEMBL4533844	Cc1ccc(-c2nc(NC(=O)C3CCN(Cc4ccccc4C(F)(F)F)CC3...	7570.0	interr
5040	CHEMBL4570655	Cc1ccc(-c2nc(NC(=O)C3CCN(Cc4ccccc4C)CC3)sc2-	10000.0	i

Calculating Lipinski Descriptors

Christopher Lipinski, a scientist at Pfizer, came up with a set of rule-of-thumb for evaluating the **druglikeness** of compounds. Such druglikeness is based on the Absorption, Distribution, Metabolism and Excretion (ADME) that is also known as the pharmacokinetic profile. Lipinski analyzed all orally active FDA-approved drugs in the formulation of what is to be known as the **Rule-of-Five** or **Lipinski's Rule**.

The Lipinski's Rule stated the following:

- Molecular weight < 500 Dalton
- Octanol-water partition coefficient (LogP) < 5
- Hydrogen bond donors < 5
- Hydrogen bond acceptors < 10

The rule is called "Rule of Five", because the border values are 5, 500, 2*5, and 5.

Importing libraries

RDKit is a collection of cheminformatics and machine-learning software written in C++ and Python.

```
# Importing the necessary libraries
import numpy as np
from rdkit import Chem
from rdkit.Chem import Descriptors, Lipinski
```

Calculating the descriptors

```
df_no_smiles = df.drop(columns='canonical_smiles')
```

```
smiles = []

for i in df.canonical_smiles.tolist():
  cpd = str(i).split('.')
  cpd_longest = max(cpd, key = len)
  smiles.append(cpd_longest)

smiles = pd.Series(smiles, name = 'canonical_smiles')
```

```
df_clean_smiles = pd.concat([df_no_smiles,smiles], axis=1)
df_clean_smiles
```

	molecule_chembl_id	standard_value	class	canonical_
0	CHEMBL133897	750.0	active	CCOc1nn(-c2cccc(OCc3ccccc3)c2)c
1	CHEMBL336398	100.0	active	O=C(N1CCCCC1 c2ccc(Cl)cc2)nc1SC
2	CHEMBL131588	50000.0	inactive	CN(C(=O)n1nc(-c2ccc(Cl)cc2)nc1 (F)F)c1
3	CHEMBL130628	300.0	active	O=C(N1CCCCC1 c2ccc(Cl)cc2)nc1SCC
4	CHEMBL130478	800.0	active	CSc1nc(-c2ccc(OC(F)(F)F)cc2)nn1C(=O
...	...	...	...	
5038	CHEMBL4554172	10000.0	inactive	C c2nc(NC(=O)C3CCN(Cc4cccc(F)c4)CC
5039	CHEMBL4533844	7570.0	intermediate	Cc1ccc(-c2nc(NC(=O)C3CCN(Cc4cccc (F)F
5040	CHEMBL4570655	10000.0	inactive	C c2nc(NC(=O)C3CCN(Cc4ccccc4C)C

```
# 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


df_lipinski = lipinski(df_clean_smiles.canonical_smiles)
df_lipinski
```

	MW	LogP	NumHDonors	NumHAcceptors
0	312.325	2.80320	0.0	6.0
1	376.913	4.55460	0.0	5.0

Combining DataFrames

```
# Dataframe 1
df_lipinski
```

	MW	LogP	NumHDonors	NumHAcceptors
0	312.325	2.80320	0.0	6.0
1	376.913	4.55460	0.0	5.0
2	426.851	5.35740	0.0	5.0
3	404.845	4.70690	0.0	5.0
4	346.334	3.09530	0.0	6.0
...	...	...	...	...
5038	499.655	7.08374	1.0	4.0
5039	549.662	7.96344	1.0	4.0
5040	495.692	7.25306	1.0	4.0
5041	576.510	4.06432	6.0	11.0
5042	558.539	6.27724	5.0	9.0

5043 rows × 4 columns

```
# Dataframe 2
df
```

	molecule_chembl_id	canonical_smiles	standard_value	
0	CHEMBL133897	CCOc1nn(-c2cccc(OCc3ccccc3)c2)c(=O)o1	750.0	
1	CHEMBL336398	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC1CC1	100.0	
2	CHEMBL131588	CN(C(=O)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F)c1ccccc1	50000.0	i

```
# Combining the two columns
df_combined = pd.concat([df,df_lipinski],axis=1)
df_combined
```

	molecule_chembl_id	canonical_smiles	standard_value	
0	CHEMBL133897	CCOc1nn(-c2cccc(OCc3ccccc3)c2)c(=O)o1	750.0	
1	CHEMBL336398	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC1CC1	100.0	
2	CHEMBL131588	CN(C(=O)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F)c1ccccc1	50000.0	i
3	CHEMBL130628	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F	300.0	
4	CHEMBL130478	CSc1nc(-c2ccc(OC(F)(F)F)cc2)nn1C(=O)N(C)C	800.0	
...	...	...	...	
5038	CHEMBL4554172	Cc1ccc(-c2nc(NC(=O)C3CCN(Cc4cccc(F)c4)CC3)sc2-...	10000.0	i
5039	CHEMBL4533844	Cc1ccc(-c2nc(NC(=O)C3CCN(Cc4ccccc4C(F)(F)F)CC3...	7570.0	interr
5040	CHEMBL4570655	Cc1ccc(-c2nc(NC(=O)C3CCN(Cc4ccccc4C)CC3)sc2-	10000.0	i

```
# Saving this to csv file
df_combined.to_csv('bioactivity_combined_dataframe.csv',index=False)
```

Converting IC$_{50}$ to $_{p}$IC$_{50}$

To allow IC$_{50}$ to be more evenly distributed, we will convert it to $_{p}$IC$_{50}$, which is negative logarithm of IC$_{50}$, i.e. -log$_{10}$(IC$_{50}$).

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

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

```
# Creating custom pIC50()

def pIC50(input):
    pIC50 = []

    for i in input['standard_value_norm']:

        # Converts nM to M
        molar = i*(10**-9)

        pIC50.append(-np.log10(molar))

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

    return x
```

```
# Analysing the standard_value column
df_combined.standard_value.describe()
```

```
     count    5.043000e+03
     mean     2.989356e+12
     std      1.147822e+14
     min      5.000000e-06
     25%      1.218000e+02
     50%      2.090000e+03
     75%      1.540000e+04
     max      5.888437e+15
     Name: standard_value, dtype: float64
```

```
# Noting the min and max values of log
max = -np.log10( (10**-9)* 100000000 )
min = -np.log10( (10**-9)* 10000000000 )
max,min
```

```
     (1.0, -1.0)
```

```
# Capping the standard value to 10000000000
```

```
def norm_value(input):
    norm = []

    for i in input['standard_value']:
        if i > 100000000:
          i = 100000000
        norm.append(i)

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

    return x


# Calling the function to cap values to 10000000000
df_norm = norm_value(df_combined)
df_norm
```

	molecule_chembl_id	canonical_smiles	class	MW
0	CHEMBL133897	CCOc1nn(-c2cccc(OCc3ccccc3)c2)c(=O)o1	active	312.325
1	CHEMBL336398	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC1CC1	active	376.913
2	CHEMBL131588	CN(C(=O)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F)c1ccccc1	inactive	426.851
3	CHEMBL130628	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F	active	404.845
4	CHEMBL130478	CSc1nc(-c2ccc(OC(F)(F)F)cc2)nn1C(=O)N(C)C	active	346.334
...	...	...	...	...
5038	CHEMBL4554172	Cc1ccc(-c2nc(NC(=O)C3CCN(Cc4cccc(F)c4)CC3)sc2-...	inactive	499.655
5039	CHEMBL4533844	Cc1ccc(-c2nc(NC(=O)C3CCN(Cc4ccccc4C(F)(F)F)CC3...	intermediate	549.662
5040	CHEMBL4570655	Cc1ccc(-c2nc(NC(=O)C3CCN(Cc4ccccc4C)CC3)sc2-	inactive	495.692

```
# Re-analyzing the max and min values of the column standard_value
df_norm.standard_value_norm.describe()
```

```
count    5.043000e+03
mean     3.063436e+05
std      4.553341e+06
min      5.000000e-06
25%      1.218000e+02
50%      2.090000e+03
75%      1.540000e+04
```

```
    max      1.000000e+08
    Name: standard_value_norm, dtype: float64
```

```
# Converting pIC50 from df
df_final = pIC50(df_norm)
df_final
```

	molecule_chembl_id	canonical_smiles	class	MW
0	CHEMBL133897	CCOc1nn(-c2cccc(OCc3ccccc3)c2)c(=O)o1	active	312.325
1	CHEMBL336398	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC1CC1	active	376.913
2	CHEMBL131588	CN(C(=O)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F)c1ccccc1	inactive	426.851
3	CHEMBL130628	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F	active	404.845
4	CHEMBL130478	CSc1nc(-c2ccc(OC(F)(F)F)cc2)nn1C(=O)N(C)C	active	346.334
...	...	...	...	...
5038	CHEMBL4554172	Cc1ccc(-c2nc(NC(=O)C3CCN(Cc4cccc(F)c4)CC3)sc2-...	inactive	499.655
5039	CHEMBL4533844	Cc1ccc(-c2nc(NC(=O)C3CCN(Cc4ccccc4C(F)(F)F)CC3...	intermediate	549.662
5040	CHEMBL4570655	Cc1ccc(-c2nc(NC(=O)C3CCN(Cc4ccccc4C)CC3)sc2-	inactive	495.692

```
# Re-analysing the pIC50 column
df_final.pIC50.describe()
```

```
    count    5043.000000
    mean        5.863164
    std         1.612500
    min         1.000000
    25%         4.812479
    50%         5.679854
    75%         6.914353
    max        14.301030
    Name: pIC50, dtype: float64
```

```
# Saving to csv file
df_final.to_csv('bioactivity_class_pic50.csv',index=False)
```

```
import pandas as pd
df_final = pd.read_csv('bioactivity_class_pic50.csv')
```

```
df_final.drop(['Unnamed: 0'],inplace=True,axis=1)
```

Removing the intermediate bioactivity class

```
# Removing the intermediate class
df_2class = df_final[df_final['class'] != 'intermediate']
df_2class
```

	molecule_chembl_id	canonical_smiles	class	MW	L
0	CHEMBL133897	CCOc1nn(-c2cccc(OCc3ccccc3)c2)c(=O)o1	active	312.325	2.80
1	CHEMBL336398	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC1CC1	active	376.913	4.55
2	CHEMBL131588	CN(C(=O)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F)c1ccccc1	inactive	426.851	5.35
3	CHEMBL130628	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F	active	404.845	4.70
4	CHEMBL130478	CSc1nc(-c2ccc(OC(F)(F)F)cc2)nn1C(=O)N(C)C	active	346.334	3.09
...	...	...	...	...	
5036	CHEMBL4578266	Cc1ccc(-c2nc(NC(=O)C3CCN(Cc4ccc(F)cc4Cl)CC3)sc...	inactive	534.100	7.73
5038	CHEMBL4554172	Cc1ccc(-c2nc(NC(=O)C3CCN(Cc4cccc(F)c4)CC3)sc2-...	inactive	499.655	7.08
5040	CHEMBL4570655	Cc1ccc(-c2nc(NC(=O)C3CCN(Cc4ccccc4C)CC3)sc2-	inactive	495.692	7.25

```
# Saving to csv file
df_2class.to_csv('bioactivity_2class_df.csv',index=False)
```

Exploratory Data Analysis

Importing libraries

```
# Importing the required libraries
import seaborn as sns
sns.set(style='ticks')
import matplotlib.pyplot as plt
```

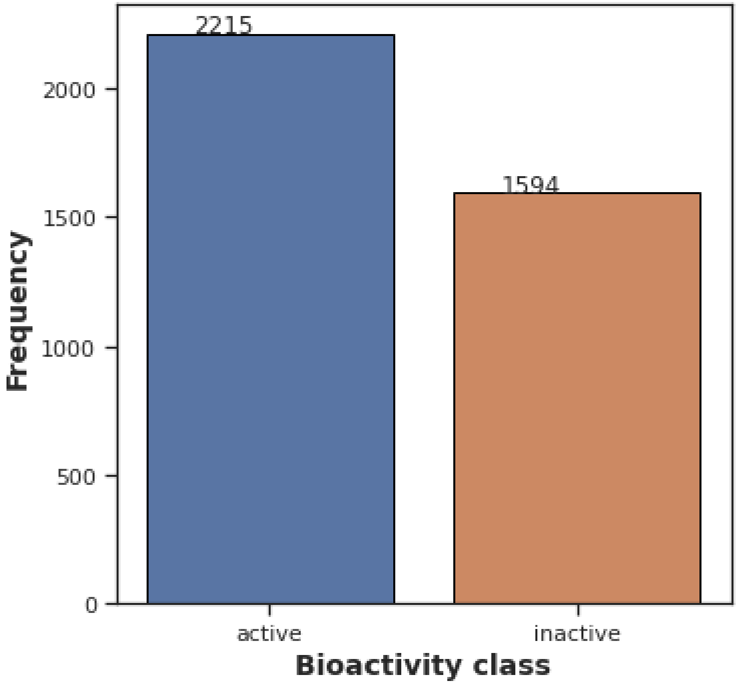
Frequency Plot of 2 bioactivity classes

```
plt.figure(figsize=(5.5, 5.5))
ax = sns.countplot(x='class',data=df_2class,edgecolor='black')

plt.xlabel('Bioactivity class', fontsize=14, fontweight='bold')
plt.ylabel('Frequency', fontsize=14, fontweight='bold')

for p in ax.patches:
    ax.annotate('{:}'.format(p.get_height()), (p.get_x()+0.15, p.get_height()+1))

plt.savefig('plot_bioactivity_class.png')
```

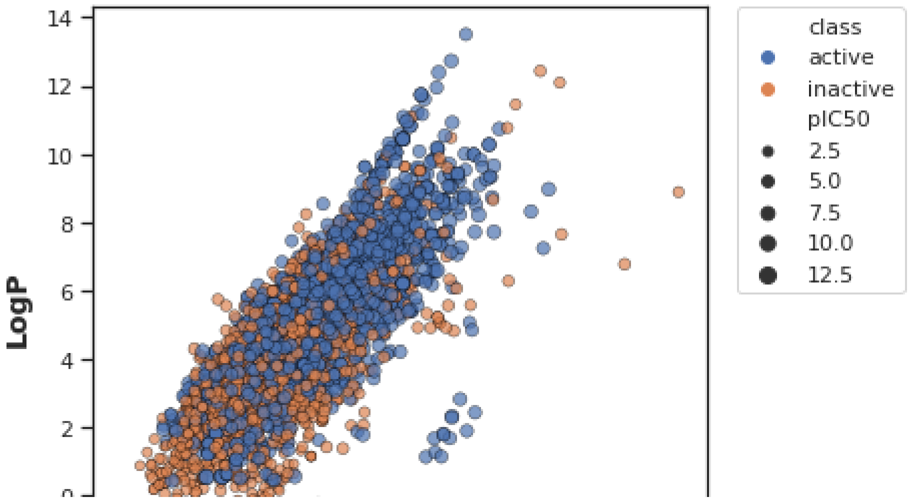


Scatter plot of MW versus LogP

```
plt.figure(figsize=(5.5, 5.5))

sns.scatterplot(x='MW', y='LogP', data=df_2class, hue='class', size='pIC50', edgecolor='black

plt.xlabel('MW', fontsize=14, fontweight='bold')
plt.ylabel('LogP', fontsize=14, fontweight='bold')
plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0)
plt.savefig('plot_MW_vs_LogP.png')
```



Box Plots

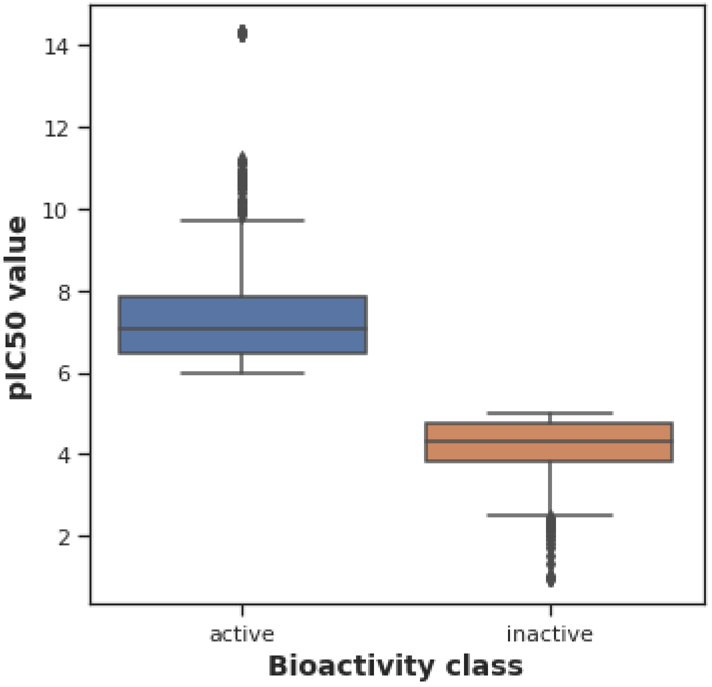
$_{p}$IC$_{50}$ Value

```
plt.figure(figsize=(5.5, 5.5))

sns.boxplot(x = 'class', y = 'pIC50', data = df_2class)

plt.xlabel('Bioactivity class', fontsize=14, fontweight='bold')
plt.ylabel('pIC50 value', fontsize=14, fontweight='bold')

plt.savefig('plot_ic50.png')
```



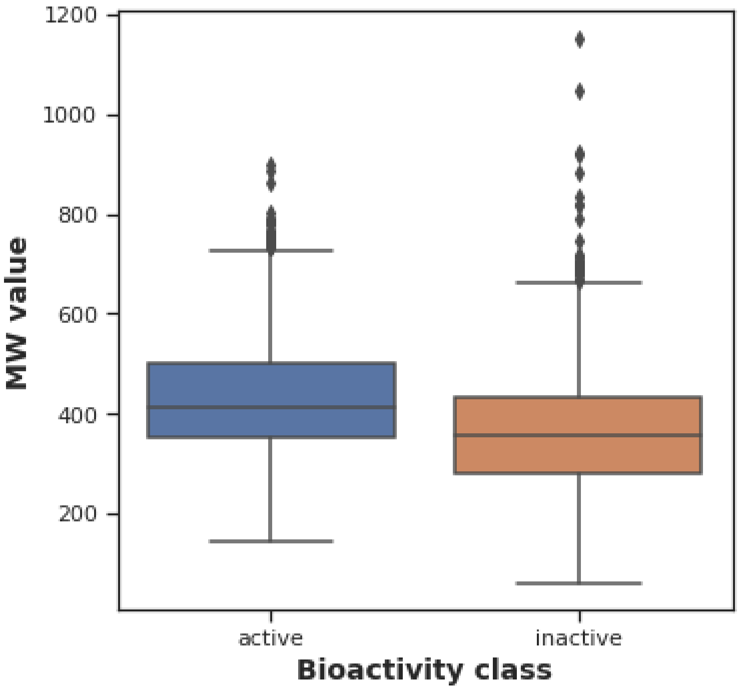
MW Value

```
plt.figure(figsize=(5.5, 5.5))

sns.boxplot(x = 'class', y = 'MW', data = df_2class)

plt.xlabel('Bioactivity class', fontsize=14, fontweight='bold')
plt.ylabel('MW value', fontsize=14, fontweight='bold')

plt.savefig('plot_MW.png')
```



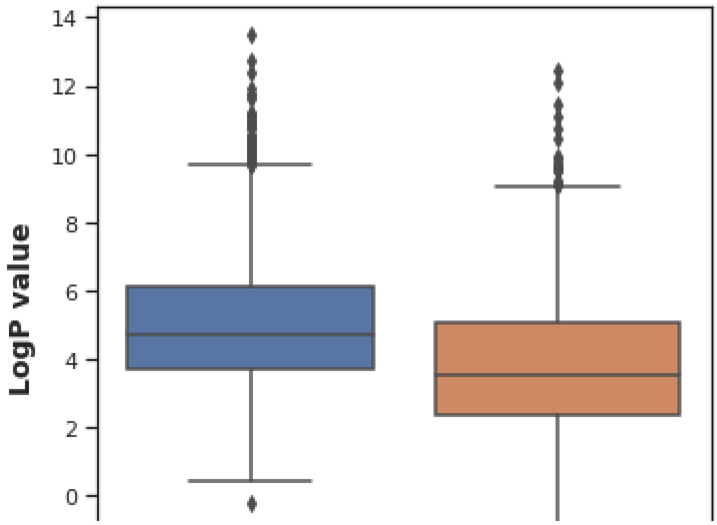
Log P Value

```
plt.figure(figsize=(5.5, 5.5))

sns.boxplot(x = 'class', y = 'LogP', data = df_2class)

plt.xlabel('Bioactivity class', fontsize=14, fontweight='bold')
plt.ylabel('LogP value', fontsize=14, fontweight='bold')

plt.savefig('plot_LogP.png')
```



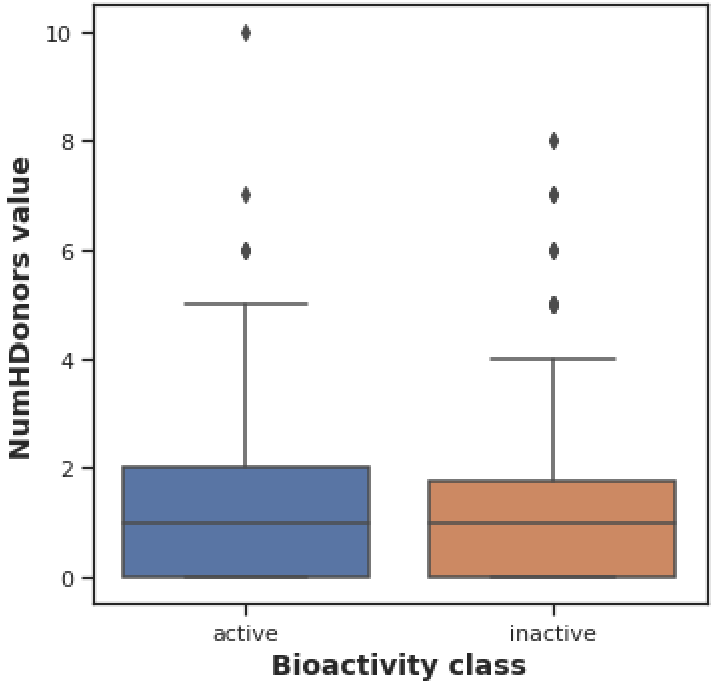
NumHDonors Value

```
plt.figure(figsize=(5.5, 5.5))

sns.boxplot(x = 'class', y = 'NumHDonors', data = df_2class)

plt.xlabel('Bioactivity class', fontsize=14, fontweight='bold')
plt.ylabel('NumHDonors value', fontsize=14, fontweight='bold')

plt.savefig('plot_NumHDonors.png')
```



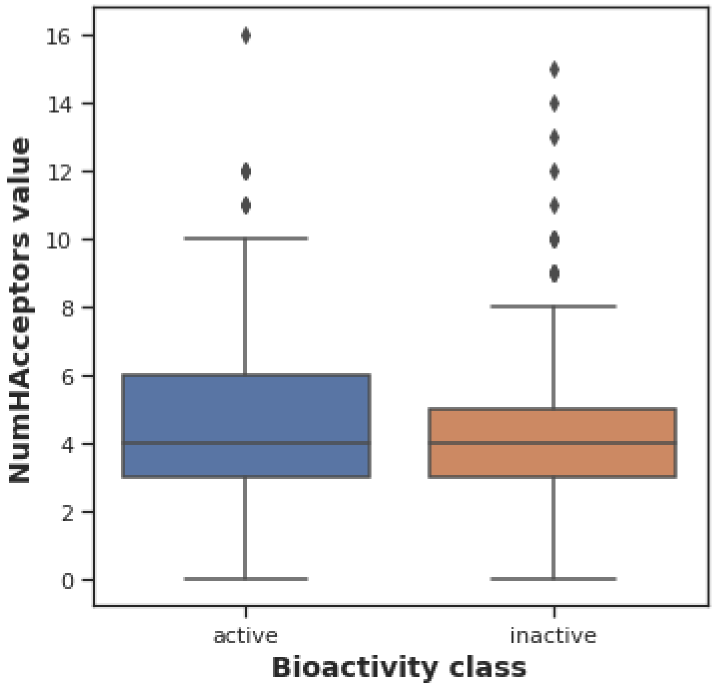
NumHAcceptors Value

```
plt.figure(figsize=(5.5, 5.5))
plt.figure(figsize=(5.5, 5.5))

sns.boxplot(x = 'class', y = 'NumHAcceptors', data = df_2class)

plt.xlabel('Bioactivity class', fontsize=14, fontweight='bold')
plt.ylabel('NumHAcceptors value', fontsize=14, fontweight='bold')

plt.savefig('plot_NumHAcceptors.png')
```



Statistical Analysis: Mann-Whitney U Test

```
# Creating MannWhitney Function

def mannwhitney(descriptor, verbose=False):

  from numpy.random import seed
  from numpy.random import randn
  from scipy.stats import mannwhitneyu

# seed the random number generator
  seed(1)

# actives and inactives
  selection = [descriptor, 'class']
  df = df_2class[selection]
  active = df[df['class'] == 'active']
  active = active[descriptor]

  selection = [descriptor, 'class']
  df = df_2class[selection]
  inactive = df[df['class'] == 'inactive']
  inactive = inactive[descriptor]

# compare samples
  stat, p = mannwhitneyu(active, inactive)

# interpret
  alpha = 0.05
  if p > alpha:
    interpretation = 'Same distribution (fail to reject H0)'
  else:
    interpretation = 'Different distribution (reject H0)'

  results = pd.DataFrame({'Descriptor':descriptor,
                          'Statistics':stat,
                          'p':p,
                          'alpha':alpha,
                          'Interpretation':interpretation}, index=[0])
  filename = 'mannwhitneyu_' + descriptor + '.csv'
  results.to_csv(filename)

  return results


# MannWhitney tests
print(mannwhitney('pIC50'))
print(mannwhitney('MW'))
print(mannwhitney('LogP'))
print(mannwhitney('NumHDonors'))
print(mannwhitney('NumHAcceptors'))
```

```
  Descriptor  Statistics    p  alpha                      Interpretation
0      pIC50         0.0  0.0   0.05  Different distribution (reject H0)
  Descriptor  Statistics  ...  alpha                      Interpretation
0         MW   1207259.5  ...   0.05  Different distribution (reject H0)

[1 rows x 5 columns]
  Descriptor  Statistics  ...  alpha                      Interpretation
0       LogP   1176292.5  ...   0.05  Different distribution (reject H0)

[1 rows x 5 columns]
   Descriptor  Statistics  ...  alpha                      Interpretation
0  NumHDonors   1524372.5  ...   0.05  Different distribution (reject H0)

[1 rows x 5 columns]
      Descriptor  Statistics  ...  alpha                      Interpretation
0  NumHAcceptors   1636225.0  ...   0.05  Different distribution (reject H0)

[1 rows x 5 columns]
```

Interpretation of Statistical Results

pIC50 values

Taking a look at pIC50 values, the **actives** and **inactives** displayed ***statistically significant difference***, which is to be expected since threshold values (`IC50 < 1,000 nM = Actives while IC50 > 10,000 nM = Inactives`, corresponding to `pIC50 > 6 = Actives and pIC50 < 5 = Inactives`) were used to define actives and inactives.

Lipinski's descriptors

All of the 4 Lipinski's descriptors exhibited ***statistically significant difference*** between the **actives** and **inactives**.

```
# Zipping the results
! zip -r results.zip . -i *.csv *.png
```

```
      adding: mannwhitneyu_pIC50.csv (deflated 14%)
      adding: plot_LogP.png (deflated 12%)
      adding: mannwhitneyu_NumHAcceptors.csv (deflated 10%)
      adding: plot_NumHAcceptors.png (deflated 12%)
      adding: mannwhitneyu_MW.csv (deflated 9%)
      adding: bioactivity_class_pic50.csv (deflated 76%)
      adding: plot_bioactivity_class.png (deflated 17%)
      adding: plot_MW.png (deflated 14%)
      adding: bioactivity_2class_df.csv (deflated 77%)
      adding: plot_NumHDonors.png (deflated 13%)
      adding: mannwhitneyu_LogP.csv (deflated 8%)
      adding: mannwhitneyu_NumHDonors.csv (deflated 11%)
      adding: plot_MW_vs_LogP.png (deflated 1%)
      adding: plot_ic50.png (deflated 13%)
```

Part-3: Descriptor Calculation and Dataset Preparation

Pre-requisite

Downloading Padel Descriptor

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

```
    --2021-05-07 09:25:10--  https://github.com/dataprofessor/bioinformatics/raw/master/pade
    Resolving github.com (github.com)... 192.30.255.112
    Connecting to github.com (github.com)|192.30.255.112|:443... connected.
    HTTP request sent, awaiting response... 302 Found
    Location: https://raw.githubusercontent.com/dataprofessor/bioinformatics/master/padel.z
```

```
--2021-05-07 09:25:10--  https://raw.githubusercontent.com/dataprofessor/bioinformatics/
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443
HTTP request sent, awaiting response... 200 OK
Length: 25768637 (25M) [application/zip]
Saving to: ‘padel.zip’

padel.zip           100%[===================>]  24.57M  54.9MB/s    in 0.4s

2021-05-07 09:25:11 (54.9 MB/s) - ‘padel.zip’ saved [25768637/25768637]

--2021-05-07 09:25:11--  https://github.com/dataprofessor/bioinformatics/raw/master/pade
Resolving github.com (github.com)... 192.30.255.112
Connecting to github.com (github.com)|192.30.255.112|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://raw.githubusercontent.com/dataprofessor/bioinformatics/master/padel.sh
--2021-05-07 09:25:11--  https://raw.githubusercontent.com/dataprofessor/bioinformatics/
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443
HTTP request sent, awaiting response... 200 OK
Length: 231 [text/plain]
Saving to: ‘padel.sh’

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

2021-05-07 09:25:12 (10.2 MB/s) - ‘padel.sh’ saved [231/231]
```

```
! unzip padel.zip
```

```
  inflating: __MACOSX/PaDEL-Descriptor/lib/._libPaDEL-Descriptor(2).jar
  inflating: PaDEL-Descriptor/lib/jgrapht-0.6.0(3).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._jgrapht-0.6.0(3).jar
  inflating: PaDEL-Descriptor/lib/jama(7).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._jama(7).jar
  inflating: PaDEL-Descriptor/lib/ambit2-core-2.4.7-SNAPSHOT.jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._ambit2-core-2.4.7-SNAPSHOT.jar
  inflating: PaDEL-Descriptor/lib/commons-cli-1.2(6).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._commons-cli-1.2(6).jar
  inflating: PaDEL-Descriptor/lib/libPaDEL-Descriptor(1).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._libPaDEL-Descriptor(1).jar
  inflating: PaDEL-Descriptor/lib/jama(4).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._jama(4).jar
  inflating: PaDEL-Descriptor/lib/libPaDEL-Jobs(2).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._libPaDEL-Jobs(2).jar
  inflating: PaDEL-Descriptor/lib/ambit2-smarts-2.4.7-SNAPSHOT(3).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._ambit2-smarts-2.4.7-SNAPSHOT(3).jar
  inflating: PaDEL-Descriptor/lib/ambit2-smarts-2.4.7-SNAPSHOT(2).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._ambit2-smarts-2.4.7-SNAPSHOT(2).jar
  inflating: PaDEL-Descriptor/lib/ambit2-smarts-2.4.7-SNAPSHOT.jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._ambit2-smarts-2.4.7-SNAPSHOT.jar
  inflating: PaDEL-Descriptor/lib/libPaDEL-Jobs(3).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._libPaDEL-Jobs(3).jar
  inflating: PaDEL-Descriptor/lib/l2fprod-common-all(1).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._l2fprod-common-all(1).jar
  inflating: PaDEL-Descriptor/lib/jama.jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._jama.jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._jama.jar
  inflating: PaDEL-Descriptor/lib/l2fprod-common-all.jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._l2fprod-common-all.jar
  inflating: PaDEL-Descriptor/lib/jama(5).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._jama(5).jar

  inflating: PaDEL-Descriptor/lib/jgrapht-0.6.0(1).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._jgrapht-0.6.0(1).jar
  inflating: PaDEL-Descriptor/lib/commons-cli-1.2(7).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._commons-cli-1.2(7).jar
  inflating: PaDEL-Descriptor/lib/libPaDEL-Descriptor.jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._libPaDEL-Descriptor.jar
  inflating: PaDEL-Descriptor/lib/libPaDEL-Jobs(4).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._libPaDEL-Jobs(4).jar
  inflating: PaDEL-Descriptor/lib/cdk-1.4.15.jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._cdk-1.4.15.jar
  inflating: PaDEL-Descriptor/lib/ambit2-smarts-2.4.7-SNAPSHOT(5).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._ambit2-smarts-2.4.7-SNAPSHOT(5).jar
  inflating: PaDEL-Descriptor/lib/ambit2-core-2.4.7-SNAPSHOT(1).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._ambit2-core-2.4.7-SNAPSHOT(1).jar
  inflating: PaDEL-Descriptor/lib/libPaDEL-Jobs(8).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._libPaDEL-Jobs(8).jar
  inflating: PaDEL-Descriptor/lib/jgrapht-0.6.0(6).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._jgrapht-0.6.0(6).jar
  inflating: PaDEL-Descriptor/lib/jama(2).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._jama(2).jar
  inflating: PaDEL-Descriptor/lib/jama(3).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._jama(3).jar
  inflating: PaDEL-Descriptor/lib/commons-cli-1.2(1).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._commons-cli-1.2(1).jar
  inflating: PaDEL-Descriptor/lib/guava-17.0.jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._guava-17.0.jar
  inflating: PaDEL-Descriptor/lib/ambit2-smarts-2.4.7-SNAPSHOT(4).jar
  inflating: __MACOSX/PaDEL-Descriptor/lib/._ambit2-smarts-2.4.7-SNAPSHOT(4).jar
  inflating: PaDEL-Descriptor/lib/libPaDEL-Jobs(5).jar
```

Loading the bioactivity dataframe

```
# Importing required libraries
import pandas as pd
```

```
df = pd.read_csv('/content/bioactivity_class_pic50.csv')
df.drop(['Unnamed: 0'],inplace=True,axis=1)
df
```

	molecule_chembl_id	canonical_smiles	class	MW
0	CHEMBL133897	CCOc1nn(-c2cccc(OCc3ccccc3)c2)c(=O)o1	active	312.325
1	CHEMBL336398	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC1CC1	active	376.913
2	CHEMBL131588	CN(C(=O)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F)c1ccccc1	inactive	426.851
3	CHEMBL130628	O=C(N1CCCCC1)n1nc(-c2ccc(Cl)cc2)nc1SCC(F)(F)F	active	404.845
4	CHEMBL130478	CSc1nc(-c2ccc(OC(F)(F)F)cc2)nn1C(=O)N(C)C	active	346.334

```
# Selecting particular columns
selection = ['canonical_smiles','molecule_chembl_id']
df_selection = df[selection]
df_selection.to_csv('molecule.smi', sep='\t', index=False, header=False)
```

```
                                                                  (F)F)CC3...
```

```
# Viewing the first 5 rows
! 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
```

```
# Viewing the length
! cat molecule.smi | wc -l
```

```
5043
```

Calculating Fingerprint Descriptors

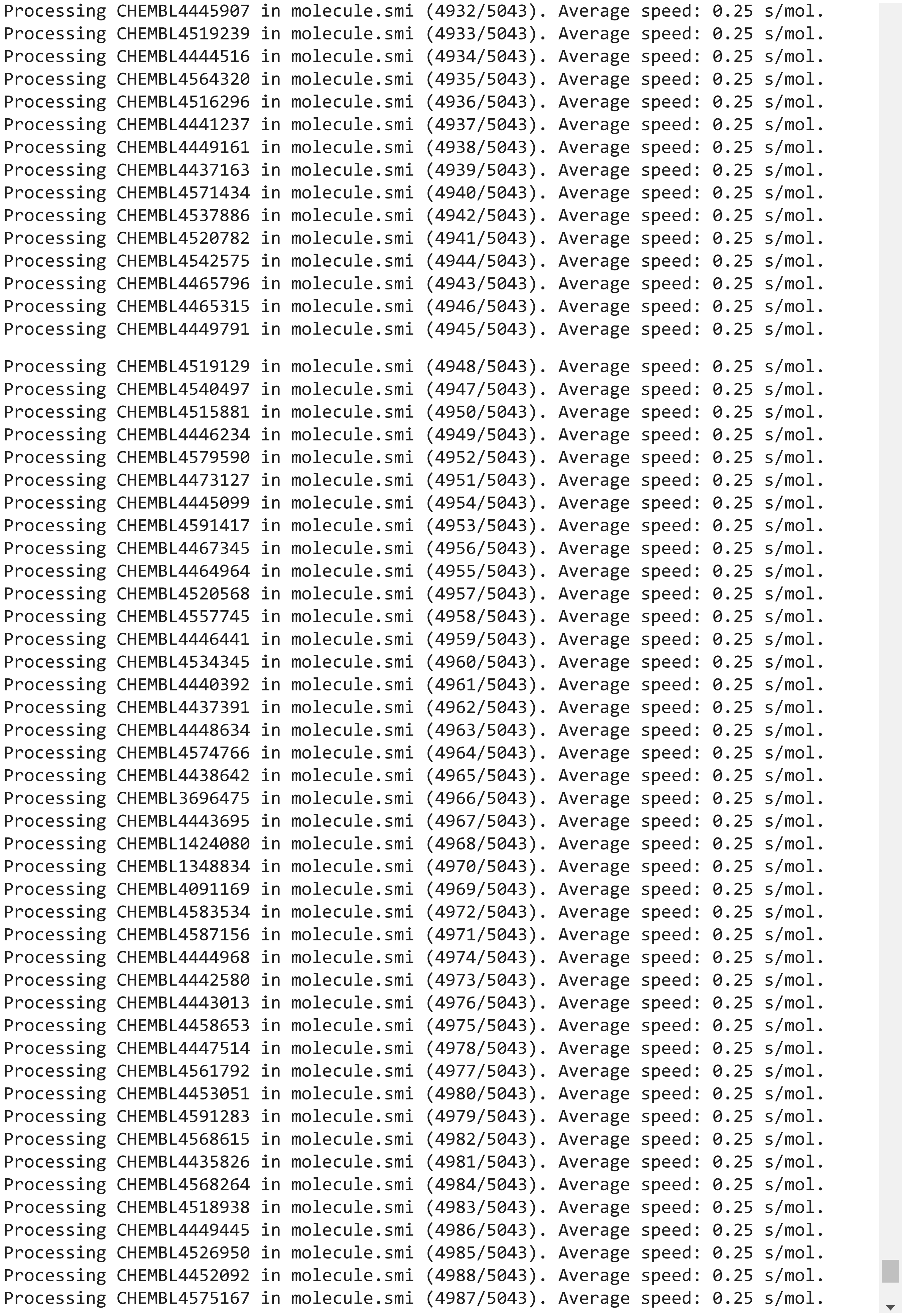
Calculate PaDEL Descriptors

```
! cat padel.sh
```

```
java -Xms1G -Xmx1G -Djava.awt.headless=true -jar ./PaDEL-Descriptor/PaDEL-Descriptor.jar
```

```
! bash padel.sh
```

```
Processing CHEMBL4467130 in molecule.smi (4930/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4436169 in molecule.smi (4931/5043). Average speed: 0.25 s/mol.
```



```
Processing CHEMBL4445907 in molecule.smi (4932/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4519239 in molecule.smi (4933/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4444516 in molecule.smi (4934/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4564320 in molecule.smi (4935/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4516296 in molecule.smi (4936/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4441237 in molecule.smi (4937/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4449161 in molecule.smi (4938/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4437163 in molecule.smi (4939/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4571434 in molecule.smi (4940/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4537886 in molecule.smi (4942/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4520782 in molecule.smi (4941/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4542575 in molecule.smi (4944/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4465796 in molecule.smi (4943/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4465315 in molecule.smi (4946/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4449791 in molecule.smi (4945/5043). Average speed: 0.25 s/mol.

Processing CHEMBL4519129 in molecule.smi (4948/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4540497 in molecule.smi (4947/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4515881 in molecule.smi (4950/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4446234 in molecule.smi (4949/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4579590 in molecule.smi (4952/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4473127 in molecule.smi (4951/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4445099 in molecule.smi (4954/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4591417 in molecule.smi (4953/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4467345 in molecule.smi (4956/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4464964 in molecule.smi (4955/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4520568 in molecule.smi (4957/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4557745 in molecule.smi (4958/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4446441 in molecule.smi (4959/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4534345 in molecule.smi (4960/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4440392 in molecule.smi (4961/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4437391 in molecule.smi (4962/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4448634 in molecule.smi (4963/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4574766 in molecule.smi (4964/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4438642 in molecule.smi (4965/5043). Average speed: 0.25 s/mol.
Processing CHEMBL3696475 in molecule.smi (4966/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4443695 in molecule.smi (4967/5043). Average speed: 0.25 s/mol.
Processing CHEMBL1424080 in molecule.smi (4968/5043). Average speed: 0.25 s/mol.
Processing CHEMBL1348834 in molecule.smi (4970/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4091169 in molecule.smi (4969/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4583534 in molecule.smi (4972/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4587156 in molecule.smi (4971/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4444968 in molecule.smi (4974/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4442580 in molecule.smi (4973/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4443013 in molecule.smi (4976/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4458653 in molecule.smi (4975/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4447514 in molecule.smi (4978/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4561792 in molecule.smi (4977/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4453051 in molecule.smi (4980/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4591283 in molecule.smi (4979/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4568615 in molecule.smi (4982/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4435826 in molecule.smi (4981/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4568264 in molecule.smi (4984/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4518938 in molecule.smi (4983/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4449445 in molecule.smi (4986/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4526950 in molecule.smi (4985/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4452092 in molecule.smi (4988/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4575167 in molecule.smi (4987/5043). Average speed: 0.25 s/mol.
Processing CHEMBL4566242 in molecule.smi (4990/5043). Average speed: 0.25 s/mol
```

Preparing the Data Matrices

X Matrix

```
# Reading the op file from padel step
df_X = pd.read_csv('descriptors_output.csv')
df_X
```

	Name	PubchemFP0	PubchemFP1	PubchemFP2	PubchemFP3	PubchemFP4	Pubch
0	CHEMBL336398	1	1	1	0	0	
1	CHEMBL133897	1	1	1	0	0	
2	CHEMBL130628	1	1	1	0	0	
3	CHEMBL131588	1	1	0	0	0	
4	CHEMBL130478	1	1	0	0	0	
...	...	...	...	...	...	...	
5038	CHEMBL4554172	1	1	1	0	0	
5039	CHEMBL4533844	1	1	1	0	0	
5040	CHEMBL4570655	1	1	1	1	0	
5041	CHEMBL4571704	1	1	1	0	0	
5042	CHEMBL4556664	1	1	1	0	0	

5043 rows × 882 columns

```
# Dropping the name column
df_X = df_X.drop(columns=['Name'])
df_X
```

	PubchemFP0	PubchemFP1	PubchemFP2	PubchemFP3	PubchemFP4	PubchemFP5	PubchemFP6
0	1	1	1	0	0	0	0
1	1	1	1	0	0	0	0
2	1	1	1	0	0	0	0
3	1	1	0	0	0	0	0
4	1	1	0	0	0	0	0
...	...	...	...	...	...	...	...
5038	1	1	1	0	0	0	0
5040	1	1	1	1	0	0	0

Y Column

```
# Using the pic50 as y column
df_Y = df['pIC50']
df_Y
```

```
0       6.124939
1       7.000000
2       4.301030
3       6.522879
4       6.096910
          ...
5038    5.000000
5039    5.120904
5040    5.000000
5041    4.809668
5042    4.165579
Name: pIC50, Length: 5043, dtype: float64
```

Combining the X and Y to prepare a dataset

```
# Concatenating X and Y variables
dataset = pd.concat([df_X,df_Y],axis=1)
dataset
```

	PubchemFP0	PubchemFP1	PubchemFP2	PubchemFP3	PubchemFP4	PubchemFP5	PubchemFP6
0	1	1	1	0	0	0	0
1	1	1	1	0	0	0	0
2	1	1	1	0	0	0	0
3	1	1	0	0	0	0	0
4	1	1	0	0	0	0	0
...	...	...	...	...	...	...	...
5038	1	1	1	0	0	0	0

```
# Saving to csv file
dataset.to_csv('bioactivity_data_class_pIC50_pubchem_fp.csv',index=False)
```

Part-4: Regression Model with Random Forest

5043 rows × 882 columns

Pre-requisite

Importing libraries

```
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
```

Loading the dataset

```
df = pd.read_csv('bioactivity_data_class_pIC50_pubchem_fp.csv')
df
```

```
      PubchemFP0  PubchemFP1  PubchemFP2  PubchemFP3  PubchemFP4  PubchemFP5  PubchemFP6
   0           1           1           1           0           0           0           0
   1           1           1           1           0           0           0           0
   2           1           1           1           0           0           0           0
   3           1           1           0           0           0           0           0
   4           1           1           0           0           0           0           0
 ...         ...         ...         ...         ...         ...         ...         ..
5038           1           1           1           0           0           0           0
5039           1           1           1           0           0           0           0
```

Pre-process Data

Input features

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

```
      PubchemFP0  PubchemFP1  PubchemFP2  PubchemFP3  PubchemFP4  PubchemFP5  PubchemFP6
   0           1           1           1           0           0           0           0
   1           1           1           1           0           0           0           0
   2           1           1           1           0           0           0           0
   3           1           1           0           0           0           0           0
   4           1           1           0           0           0           0           0
 ...         ...         ...         ...         ...         ...         ...         ..
5038           1           1           1           0           0           0           0
5039           1           1           1           0           0           0           0
5040           1           1           1           1           0           0           0
5041           1           1           1           0           0           0           0
5042           1           1           1           0           0           0           0

5043 rows × 881 columns
```

Output Feature

```
Y = df.pIC50
Y
```

```
0       6.124939
1       7.000000
2       4.301030
3       6.522879
4       6.096910
          ...
5038    5.000000
5039    5.120904
5040    5.000000
5041    4.809668
5042    4.165579
Name: pIC50, Length: 5043, dtype: float64
```

Checking the data dimensions

```
# X dimension
X.shape
```

```
(5043, 881)
```

```
# Y dimension
Y.shape
```

```
(5043,)
```

Remove low variance features

```
from sklearn.feature_selection import VarianceThreshold

selection = VarianceThreshold(threshold=(.8 * (1 - .8)))
X = selection.fit_transform(X)
X
```

```
array([[0, 1, 1, ..., 0, 0, 0],
       [0, 1, 0, ..., 0, 0, 0],
       [0, 1, 1, ..., 0, 0, 0],
       ...,
       [1, 1, 0, ..., 1, 0, 0],
       [0, 0, 0, ..., 1, 1, 0],
       [0, 0, 0, ..., 1, 1, 0]])
```

Data Split

```
# Train Test Split into 80-20
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2)
```

```
# Checking dimensions of train set
X_train.shape, Y_train.shape
```

```
     ((4034, 140), (4034,))
```

```
# Checking dimensions of test set
X_test.shape, Y_test.shape
```

```
     ((1009, 140), (1009,))
```

Building a Regression Model: Random Forest

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

```
     0.5532211320295918
```

```
# Predicting the values using model
Y_pred = model.predict(X_test)
```

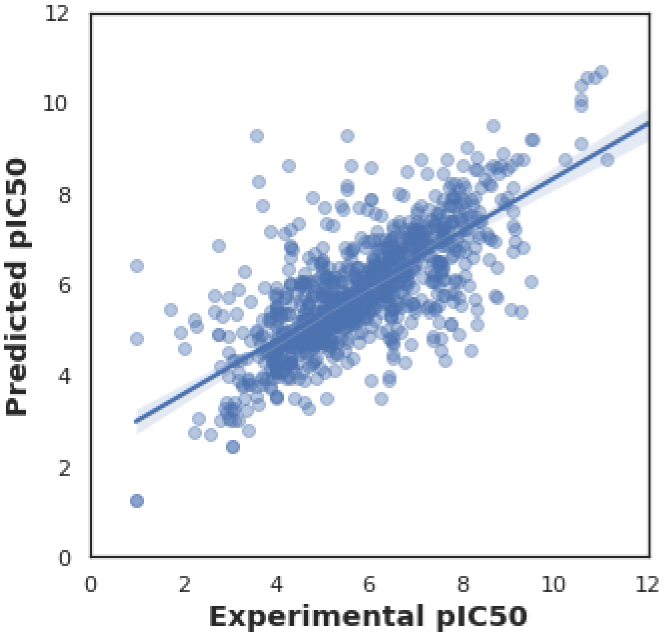
Scatter Plot of Experimental vs Predicted ${}_{p}IC_{50}$ Values

```
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)
ax.figure.set_size_inches(5, 5)
plt.show()
```

```
    /usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
      FutureWarning
```



Part-5: Comparing Regressors

Pre-requisite

Installing and Importing libraries

```
# Downloading the libraries
! pip install lazypredict
```

```
Collecting lazypredict
  Downloading https://files.pythonhosted.org/packages/97/38/cadb2b79268c7f82f6b027bf0b2
Collecting tqdm==4.56.0
  Downloading https://files.pythonhosted.org/packages/80/02/8f8880a4fd6625461833abcf679
     |████████████████████████████████| 81kB 4.1MB/s
Collecting scipy==1.5.4
  Downloading https://files.pythonhosted.org/packages/dc/7e/8f6a79b102ca1ea928bae8998b0
     |████████████████████████████████| 25.9MB 1.6MB/s
Requirement already satisfied: click==7.1.2 in /usr/local/lib/python3.7/dist-packages (
Collecting joblib==1.0.0
  Downloading https://files.pythonhosted.org/packages/34/5b/bd0f0fb5564183884d8e35b81d0
     |████████████████████████████████| 307kB 45.1MB/s
Collecting lightgbm==2.3.1
  Downloading https://files.pythonhosted.org/packages/0b/9d/ddcb2f43aca194987f1a99e27ed
     |████████████████████████████████| 1.2MB 40.2MB/s
Collecting scikit-learn==0.23.1
  Downloading https://files.pythonhosted.org/packages/b8/7e/74e707b66490d4eb05f702966ad
     |████████████████████████████████| 6.8MB 44.2MB/s
Requirement already satisfied: six==1.15.0 in /usr/local/lib/python3.7/dist-packages (f
Collecting PyYAML==5.3.1
  Downloading https://files.pythonhosted.org/packages/64/c2/b80047c7ac2478f9501676c988a
     |████████████████████████████████| 276kB 50.4MB/s
Collecting pytest==5.4.3
  Downloading https://files.pythonhosted.org/packages/9f/f3/0a83558da436a081344aa6c8b85
     |████████████████████████████████| 256kB 42.5MB/s
Collecting pandas==1.0.5
  Downloading https://files.pythonhosted.org/packages/af/f3/683bf2547a3eaeec15b39cef86f
     |████████████████████████████████| 10.1MB 45.1MB/s
Collecting numpy==1.19.1
  Downloading https://files.pythonhosted.org/packages/50/8f/29d5688614f9bba59931683d5d3
     |████████████████████████████████| 14.5MB 335kB/s
Collecting xgboost==1.1.1
  Downloading https://files.pythonhosted.org/packages/7c/32/a11befbb003e0e6b7e062a77f01
     |████████████████████████████████| 127.6MB 99kB/s
Collecting threadpoolctl>=2.0.0
  Downloading https://files.pythonhosted.org/packages/f7/12/ec3f2e203afa394a14991172935
Requirement already satisfied: attrs>=17.4.0 in /usr/local/lib/python3.7/dist-packages 
Requirement already satisfied: more-itertools>=4.0.0 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: packaging in /usr/local/lib/python3.7/dist-packages (fro
Requirement already satisfied: wcwidth in /usr/local/lib/python3.7/dist-packages (from 
Requirement already satisfied: py>=1.5.0 in /usr/local/lib/python3.7/dist-packages (fro
Collecting pluggy<1.0,>=0.12
  Downloading https://files.pythonhosted.org/packages/a0/28/85c7aa31b80d150b772fbe4a229
Requirement already satisfied: importlib-metadata>=0.12; python_version < "3.8" in /usr
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: python-dateutil>=2.6.1 in /usr/local/lib/python3.7/dist-
Requirement already satisfied: pyparsing>=2.0.2 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: typing-extensions>=3.6.4; python_version < "3.8" in /usr
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (fro
Building wheels for collected packages: PyYAML
  Building wheel for PyYAML (setup.py) ... done
  Created wheel for PyYAML: filename=PyYAML-5.3.1-cp37-cp37m-linux_x86_64.whl size=4462
  Stored in directory: /root/.cache/pip/wheels/a7/c1/ea/cf5bd31012e735dc1dfea3131a2d5ea
Successfully built PyYAML
ERROR: tensorflow 2.4.1 has requirement numpy~=1.19.2, but you'll have numpy 1.19.1 whi
ERROR: google-colab 1.0.0 has requirement pandas~=1.1.0; python_version >= "3.0", but y
ERROR: datascience 0.10.6 has requirement folium==0.2.1, but you'll have folium 0.8.3 w
```

```
ERROR: albumentations 0.1.12 has requirement imgaug<0.2.7,>=0.2.5, but you'll have imgau
Installing collected packages: tqdm, numpy, scipy, joblib, threadpoolctl, scikit-learn,
  Found existing installation: tqdm 4.41.1
    Uninstalling tqdm-4.41.1:
      Successfully uninstalled tqdm-4.41.1
  Found existing installation: numpy 1.19.5
    Uninstalling numpy-1.19.5:
      Successfully uninstalled numpy-1.19.5
  Found existing installation: scipy 1.4.1
    Uninstalling scipy-1.4.1:
      Successfully uninstalled scipy-1.4.1
  Found existing installation: joblib 1.0.1
    Uninstalling joblib-1.0.1:
      Successfully uninstalled joblib-1.0.1
  Found existing installation: scikit-learn 0.22.2.post1
    Uninstalling scikit-learn-0.22.2.post1:
      Successfully uninstalled scikit-learn-0.22.2.post1
  Found existing installation: lightgbm 2.2.3
    Uninstalling lightgbm-2.2.3:
      Successfully uninstalled lightgbm-2.2.3
  Found existing installation: PyYAML 3.13
    Uninstalling PyYAML-3.13:
      Successfully uninstalled PyYAML-3.13
  Found existing installation: pluggy 0.7.1
    Uninstalling pluggy-0.7.1:
      Successfully uninstalled pluggy-0.7.1
  Found existing installation: pytest 3.6.4
    Uninstalling pytest-3.6.4:
      Successfully uninstalled pytest-3.6.4
  Found existing installation: pandas 1.1.5
```

```
# Importing libraries
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
import lazypredict
from lazypredict.Supervised import LazyRegressor
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:143: FutureWarning:
  warnings.warn(message, FutureWarning)
```

Loading dataset

```
# Reading the dataset
df = pd.read_csv('bioactivity_data_class_pIC50_pubchem_fp.csv')
df
```

	PubchemFP0	PubchemFP1	PubchemFP2	PubchemFP3	PubchemFP4	PubchemFP5	PubchemFP6
0	1	1	1	0	0	0	0
1	1	1	1	0	0	0	0
2	1	1	1	0	0	0	0
3	1	1	0	0	0	0	0
4	1	1	0	0	0	0	0
...	...	...	...	...	...	...	..
5038	1	1	1	0	0	0	0
5039	1	1	1	0	0	0	0
5040	1	1	1	1	0	0	0
5041	1	1	1	0	0	0	0

```
# Loading the input and output features
X = df.drop('pIC50', axis=1)
Y = df.pIC50
```

Pre-processing Data

```
# Remove low variance features
from sklearn.feature_selection import VarianceThreshold
selection = VarianceThreshold(threshold=(.8 * (1 - .8)))
X = selection.fit_transform(X)
X.shape
```

```
    (5043, 140)
```

```
# Perform data splitting using 80/20 ratio
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=42)
```

```
X_train.shape, Y_train.shape
```

```
    ((4034, 140), (4034,))
```

```
X_test.shape, Y_test.shape
```

```
    ((1009, 140), (1009,))
```

Comparing Using ML Algorithms

```
# Defines and builds the lazyclassifier
clf = LazyRegressor(verbose=0,ignore_warnings=True, custom_metric=None)
models_train,predictions_train = clf.fit(X_train, X_train, Y_train, Y_train)
models_test,predictions_test = clf.fit(X_train, X_test, Y_train, Y_test)
```

```
     100%|██████████| 42/42 [01:07<00:00,  1.61s/it]
     100%|██████████| 42/42 [00:00<00:00, 243652.51it/s]
```

```
# Performance table of the training set (80% subset)
predictions_train
```

Model	Adjusted R-Squared	R-Squar
DecisionTreeRegressor	0.86	0.
ExtraTreeRegressor	0.86	0.
ExtraTreesRegressor	0.86	0.
GaussianProcessRegressor	0.86	0.
RandomForestRegressor	0.83	0.
XGBRegressor	0.82	0.
BaggingRegressor	0.81	0.
MLPRegressor	0.79	0.
HistGradientBoostingRegressor	0.67	0.
LGBMRegressor	0.67	0.
KNeighborsRegressor	0.64	0.
SVR	0.54	0.
NuSVR	0.53	0.
GradientBoostingRegressor	0.44	0.
Ridge	0.31	0.
ElasticNetCV	0.30	0.
RidgeCV	0.30	0.
LassoCV	0.30	0.

Visualise the data of each model performance

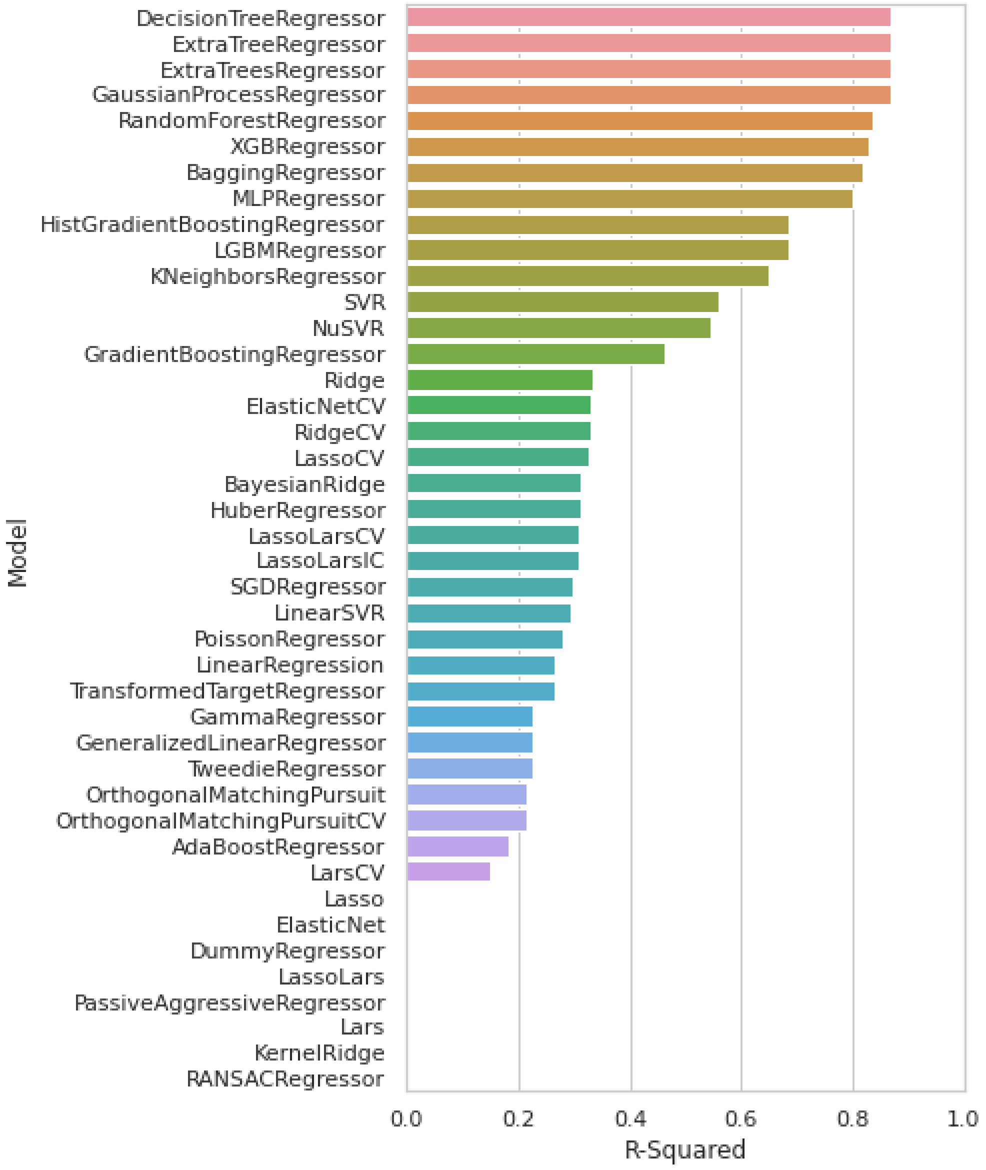
LassoLarsCV	0.28	0.

Bar-Plot of R-squared values

```
# Bar plot of R-squared values
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(5, 10))
sns.set_theme(style="whitegrid")
ax = sns.barplot(y=predictions_train.index, x="R-Squared", data=predictions_train)
ax.set(xlim=(0, 1))
```

```
[(0.0, 1.0)]
```

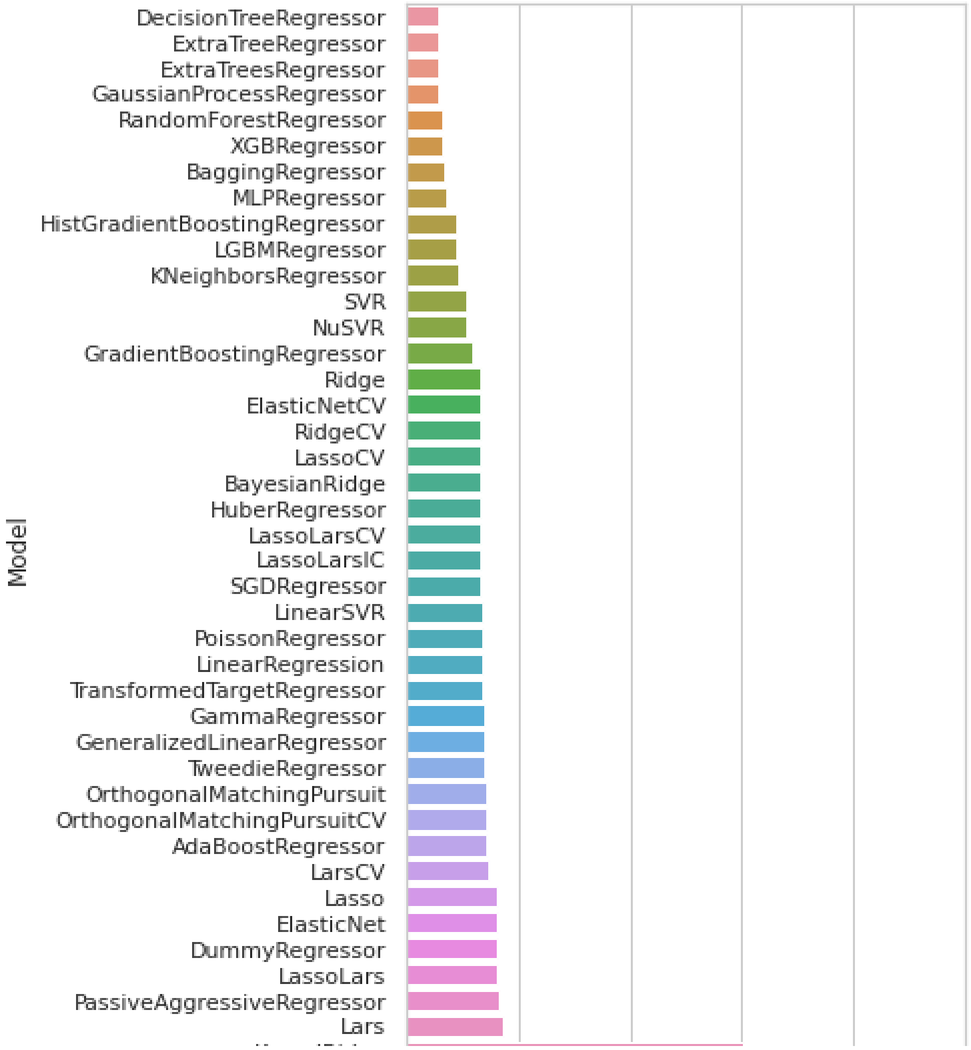


Bar-Plot of RMSE Values

```
# Bar plot of RMSE values
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(5, 10))
sns.set_theme(style="whitegrid")
ax = sns.barplot(y=predictions_train.index, x="RMSE", data=predictions_train)
ax.set(xlim=(0, 10))
```

```
[(0.0, 10.0)]
```

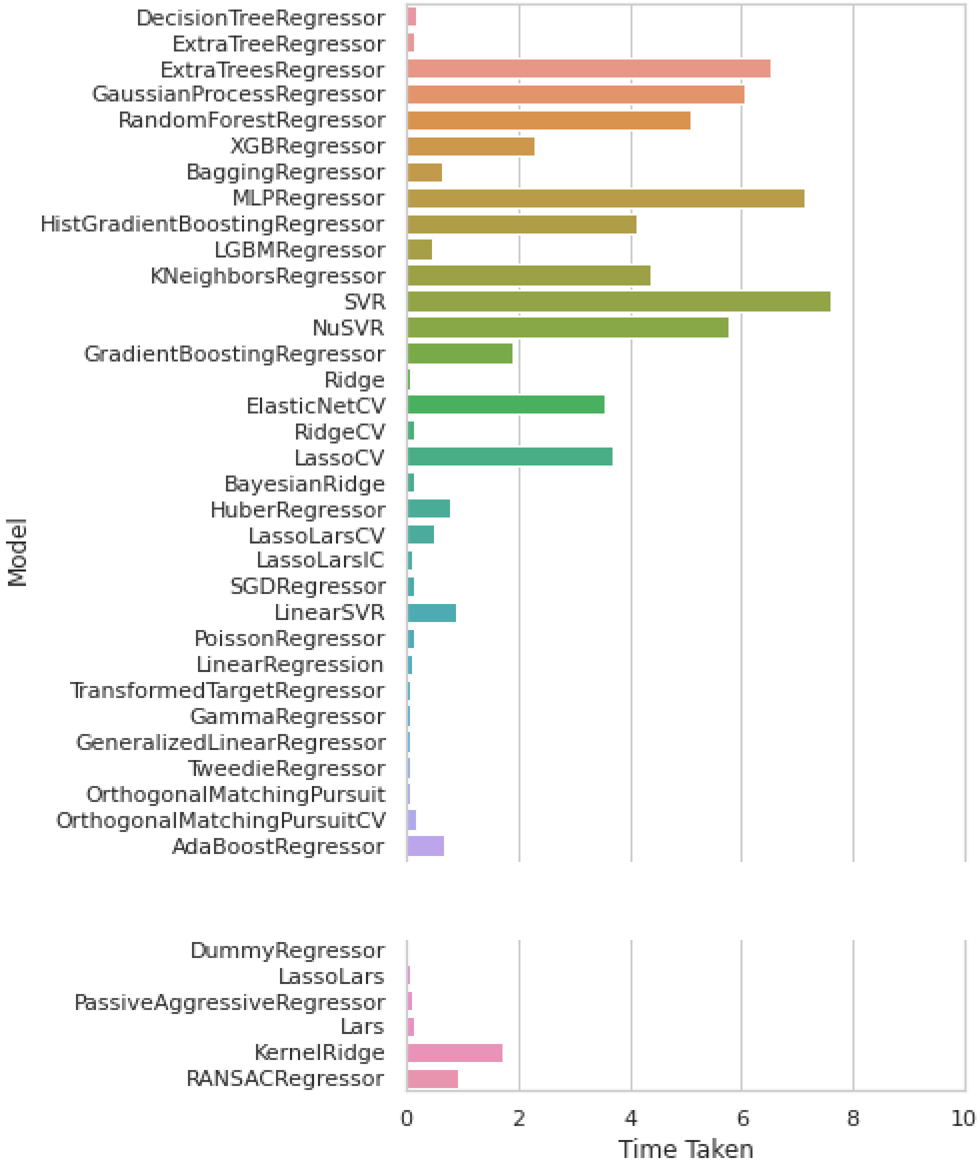


Bar-Plot of Calculation Time

```
# Bar plot of calculation time
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(5, 10))
sns.set_theme(style="whitegrid")
ax = sns.barplot(y=predictions_train.index, x="Time Taken", data=predictions_train)
ax.set(xlim=(0, 10))
```

```
[(0.0, 10.0)]
```



✓ 0s completed at 3:34 PM