ADME-favourable

November 5, 2022

1 ADME Favourable Compounds

Aim: Isolate compounds fulfilling the Lipinski rules, so they will be able to enter the body

1.1 Importing Modules

[1]:
```
try:
    from rdkit import Chem
    from rdkit.Chem import Descriptors, Draw, PandasTools
except:
    !pip install rdkit

from pathlib import Path
import math

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.lines import Line2D
import matplotlib.patches as mpatches
from rdkit import Chem
from rdkit.Chem import Descriptors, Draw, PandasTools
```

```
Collecting rdkit
  Using cached
rdkit-2022.9.1-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (29.7
MB)
Requirement already satisfied: numpy in /opt/conda/lib/python3.9/site-packages
(from rdkit) (1.21.6)
Requirement already satisfied: Pillow in /opt/conda/lib/python3.9/site-packages
(from rdkit) (9.1.1)
Installing collected packages: rdkit
Successfully installed rdkit-2022.9.1
```

[2]:
```
# Set path to this notebook
HERE = Path(_dh[-1])
DATA = HERE / "data"
```

[3]:
```
# Load Data
molecules = pd.read_csv(DATA/"PTP1B_compounds.csv")
PandasTools.AddMoleculeColumnToFrame(molecules, "smiles")
molecules
```

[3]:

```
      Unnamed: 0 molecule_chembl_id         IC50 units  \
0              0      CHEMBL4524071         0.42    nM
1              1       CHEMBL505512         0.67    nM
2              2       CHEMBL444092         1.17    nM
3              3       CHEMBL430266         1.20    nM
4              4       CHEMBL280487         1.24    nM
…            …                  …           …    …
2978        2978       CHEMBL324473  12200000.00    nM
2979        2979       CHEMBL169826  13000000.00    nM
2980        2980       CHEMBL100267  20000000.00    nM
2981        2981        CHEMBL95668  28000000.00    nM
2982        2982       CHEMBL330395  29000000.00    nM

                                                 smiles     pIC50  \
0     C/C=C1\C(=O)N[C@@H](C(=O)O)[C@H](C)C(=O)N[C@@H…  9.376751
1     CO[C@H]([C@H](O)CC(=O)[C@@H](C)[C@@H](O)CC[C@@…  9.173925
2     C=C1C(=O)N[C@H](C)C(=O)N[C@@H](CC(C)C)C(=O)N[C…  8.931814
3     COC[C@@H]([C@H](O)[C@H](O)C(=O)NCC[C@H](C)c1nc…  8.920819
4     C=C1[C@@H]([C@@H](O)C[C@H](C)[C@H]2O[C@@]3(CCC…  8.906578
…                                                   …        …
2978      C=CCOC(=O)/C=C\c1cccc(C(F)(F)P(=O)(O)O)c1.N.N  1.913640
2979  CC(=O)N[C@@H](CC(=O)O)C(=O)N[C@@H](C)C(=O)N[C@…  1.886057
2980                    COc1ccc2cc(C(=O)O)ccc2c1C(=O)O  1.698970
2981  O=S(=O)([O-])c1cccc2c(S(=O)(=O)[O-])cccc12.[Na…  1.552842
2982              CS(=O)(=O)Nc1cccc2cc(S(=O)(=O)O)ccc12  1.537602

                                                 ROMol
0     <rdkit.Chem.rdchem.Mol object at 0x7f6f0193e160>
1     <rdkit.Chem.rdchem.Mol object at 0x7f6f0193e100>
2     <rdkit.Chem.rdchem.Mol object at 0x7f6f0193e0a0>
3     <rdkit.Chem.rdchem.Mol object at 0x7f6f0193e040>
4     <rdkit.Chem.rdchem.Mol object at 0x7f6f015b1220>
…                                                   …
2978  <rdkit.Chem.rdchem.Mol object at 0x7f6f0150f400>
2979  <rdkit.Chem.rdchem.Mol object at 0x7f6f0150f460>
2980  <rdkit.Chem.rdchem.Mol object at 0x7f6f0150f4c0>
2981  <rdkit.Chem.rdchem.Mol object at 0x7f6f0150f520>
2982  <rdkit.Chem.rdchem.Mol object at 0x7f6f0150f580>

[2983 rows x 7 columns]
```

2 Molecular Properties

[4]:
```
# Clipping molecular properties to the dataframe, properties of the␣
 ↪Lipinski-rule
molecules["molecular_weight"] = molecules["ROMol"].apply(Descriptors.ExactMolWt)
molecules["n_hba"] = molecules["ROMol"].apply(Descriptors.NumHAcceptors)
molecules["n_hbd"] = molecules["ROMol"].apply(Descriptors.NumHDonors)
molecules["logp"] = molecules["ROMol"].apply(Descriptors.MolLogP)
# Colors are used for plotting the molecules later
# NBVAL_CHECK_OUTPUT
molecules
```

[4]:

	Unnamed: 0	molecule_chembl_id	IC50	units	\
0	0	CHEMBL4524071	0.42	nM	
1	1	CHEMBL505512	0.67	nM	
2	2	CHEMBL444092	1.17	nM	
3	3	CHEMBL430266	1.20	nM	
4	4	CHEMBL280487	1.24	nM	
…	…	…	…	…	
2978	2978	CHEMBL324473	12200000.00	nM	
2979	2979	CHEMBL169826	13000000.00	nM	
2980	2980	CHEMBL100267	20000000.00	nM	
2981	2981	CHEMBL95668	28000000.00	nM	
2982	2982	CHEMBL330395	29000000.00	nM	

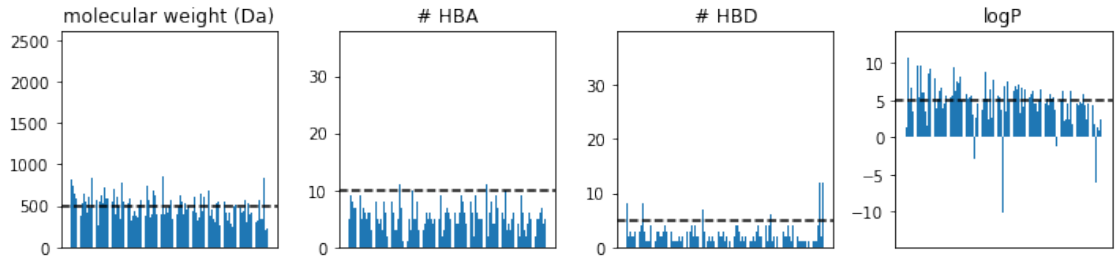
	smiles	pIC50	\
0	C/C=C1\C(=O)N[C@@H](C(=O)O)[C@H](C)C(=O)N[C@@H…	9.376751	
1	CO[C@H]([C@H](O)CC(=O)[C@@H](C)[C@@H](O)CC[C@@…	9.173925	
2	C=C1C(=O)N[C@H](C)C(=O)N[C@@H](CC(C)C)C(=O)N[C…	8.931814	
3	COC[C@@H]([C@H](O)[C@H](O)C(=O)NCC[C@H](C)c1nc…	8.920819	
4	C=C1[C@@H]([C@@H](O)C[C@H](C)[C@H]2O[C@@]3(CCC…	8.906578	
…	…	…	
2978	C=CCOC(=O)/C=C\c1cccc(C(F)(F)P(=O)(O)O)c1.N.N	1.913640	
2979	CC(=O)N[C@@H](CC(=O)O)C(=O)N[C@@H](C)C(=O)N[C@…	1.886057	
2980	COc1ccc2cc(C(=O)O)ccc2c1C(=O)O	1.698970	
2981	O=S(=O)([O-])c1cccc2c(S(=O)(=O)[O-])cccc12.[Na…	1.552842	
2982	CS(=O)(=O)Nc1cccc2cc(S(=O)(=O)O)ccc12	1.537602	

	ROMol	molecular_weight	\
0	<rdkit.Chem.rdchem.Mol object at 0x7f6f0193e160>	824.443240	
1	<rdkit.Chem.rdchem.Mol object at 0x7f6f0193e100>	766.450342	
2	<rdkit.Chem.rdchem.Mol object at 0x7f6f0193e0a0>	994.548768	
3	<rdkit.Chem.rdchem.Mol object at 0x7f6f0193e040>	1008.543605	
4	<rdkit.Chem.rdchem.Mol object at 0x7f6f015b1220>	804.465992	
…	…	…	
2978	<rdkit.Chem.rdchem.Mol object at 0x7f6f0150f400>	352.099965	
2979	<rdkit.Chem.rdchem.Mol object at 0x7f6f0150f460>	823.323578	

```
2980  <rdkit.Chem.rdchem.Mol object at 0x7f6f0150f4c0>        246.052823
2981  <rdkit.Chem.rdchem.Mol object at 0x7f6f0150f520>        331.940118
2982  <rdkit.Chem.rdchem.Mol object at 0x7f6f0150f580>        301.007864

      n_hba  n_hbd     logp
0         9      9  1.18257
1        13      3  4.78530
2        11     11  0.82807
3        16      8  4.68898
4        12      5  5.21360
…     …    …       …
2978      5      4  2.98000
2979     12     12 -3.14400
2980      3      2  2.24480
2981      6      0 -5.34400
2982      4      2  1.45800

[2983 rows x 11 columns]
```

[5]:
```
# Define the Lipinski rule
ro5_properties = {
    "molecular_weight": (500, "molecular weight (Da)"),
    "n_hba": (10, "# HBA"),
    "n_hbd": (5, "# HBD"),
    "logp": (5, "logP"),
}
```

[6]:
```
# Start 1x4 plot frame
fig, axes = plt.subplots(figsize=(10, 2.5), nrows=1, ncols=4)
x = np.arange(1, len(molecules) + 1)

# Create subplots
for index, (key, (threshold, title)) in enumerate(ro5_properties.items()):
    axes[index].bar(molecules['molecule_chembl_id'],molecules[key])
    axes[index].axhline(y=threshold, color="black", linestyle="dashed")
    axes[index].set_title(title)
    axes[index].set_xticks([])

# Fit subplots and legend into figure
plt.tight_layout()
plt.show()
```



Most compounds do not adhere to the LogP rule and molecular weight rule.

[7]:
```python
def calculate_ro5_properties(smiles):
    """
    Test if input molecule (SMILES) fulfills Lipinski's rule of five.

    Parameters
    ----------
    smiles : str
        SMILES for a molecule.

    Returns
    -------
    pandas.Series
        Molecular weight, number of hydrogen bond acceptors/donor and logP value
        and Lipinski's rule of five compliance for input molecule.
    """
    # RDKit molecule from SMILES
    molecule = Chem.MolFromSmiles(smiles)
    # Calculate Ro5-relevant chemical properties
    molecular_weight = Descriptors.ExactMolWt(molecule)
    n_hba = Descriptors.NumHAcceptors(molecule)
    n_hbd = Descriptors.NumHDonors(molecule)
    logp = Descriptors.MolLogP(molecule)
    # Check if Ro5 conditions fulfilled
    conditions = [molecular_weight <= 500, n_hba <= 10, n_hbd <= 5, logp <= 5]
    ro5_fulfilled = sum(conditions) >= 3
    # Return True if no more than one out of four conditions is violated
    return pd.Series(
        [molecular_weight, n_hba, n_hbd, logp, ro5_fulfilled],
        index=["molecular_weight", "n_hba", "n_hbd", "logp", "ro5_fulfilled"],
    )
```

[8]:
```python
# This takes a couple of seconds
ro5_properties = molecules["smiles"].apply(calculate_ro5_properties)
molecules = pd.concat([molecules, ro5_properties], axis=1)
molecules = molecules[molecules['ro5_fulfilled']]
```

```
molecules
```

[8]:

	Unnamed: 0	molecule_chembl_id	IC50	units	\
14	14	CHEMBL411295	7.0	nM	
16	16	CHEMBL377141	10.0	nM	
18	18	CHEMBL1938829	11.0	nM	
22	22	CHEMBL604457	12.6	nM	
23	23	CHEMBL592290	14.0	nM	
…	…	…	…	…	
2977	2977	CHEMBL99971	12000000.0	nM	
2978	2978	CHEMBL324473	12200000.0	nM	
2980	2980	CHEMBL100267	20000000.0	nM	
2981	2981	CHEMBL95668	28000000.0	nM	
2982	2982	CHEMBL330395	29000000.0	nM	

	smiles	pIC50	\
14	O=C(O)c1cccc(/C=C/c2ccc3cc(Br)c(C(F)(F)P(=O)(O…	8.154902	
16	O=C1CC(c2ccc(C[C@H](NS(=O)(=O)c3cccc(C(F)(F)F)…	8.000000	
18	O=C(O)COc1ccc(S(=O)(=O)N(Cc2ccc(-c3csnn3)cc2)C…	7.958607	
22	CS(=O)(=O)OC1C(Cl)CCCC1Cc1ccc(N2CC(=O)CS2(=O)=…	7.899629	
23	O=C1CN(c2c(O)cc(CC3CCCCC3)cc2F)S(=O)(=O)C1	7.853872	
…	…	…	
2977	O=C(O)c1ccc2c(C(=O)O)c(O)ccc2c1	1.920819	
2978	C=CCOC(=O)/C=C\c1cccc(C(F)(F)P(=O)(O)O)c1.N.N	1.913640	
2980	COc1ccc2cc(C(=O)O)ccc2c1C(=O)O	1.698970	
2981	O=S(=O)([O-])c1cccc2c(S(=O)(=O)[O-])cccc12.[Na…	1.552842	
2982	CS(=O)(=O)Nc1cccc2cc(S(=O)(=O)O)ccc12	1.537602	

	ROMol	molecular_weight	\
14	<rdkit.Chem.rdchem.Mol object at 0x7f6f015b14c0>	482.968278	
16	<rdkit.Chem.rdchem.Mol object at 0x7f6f015b1b20>	652.024097	
18	<rdkit.Chem.rdchem.Mol object at 0x7f6f015b1c40>	694.056877	
22	<rdkit.Chem.rdchem.Mol object at 0x7f6f015b1dc0>	451.052622	
23	<rdkit.Chem.rdchem.Mol object at 0x7f6f015b1e20>	341.109707	
…	…	…	
2977	<rdkit.Chem.rdchem.Mol object at 0x7f6f0150f3a0>	232.037173	
2978	<rdkit.Chem.rdchem.Mol object at 0x7f6f0150f400>	352.099965	
2980	<rdkit.Chem.rdchem.Mol object at 0x7f6f0150f4c0>	246.052823	
2981	<rdkit.Chem.rdchem.Mol object at 0x7f6f0150f520>	331.940118	
2982	<rdkit.Chem.rdchem.Mol object at 0x7f6f0150f580>	301.007864	

	n_hba	n_hbd	logp	molecular_weight	n_hba	n_hbd	logp	\
14	3	3	5.0929	482.968278	3	3	5.0929	
16	6	2	2.3154	652.024097	6	2	2.3154	
18	9	3	4.9739	694.056877	9	3	4.9739	
22	7	1	1.4059	451.052622	7	1	1.4059	
23	4	1	2.3730	341.109707	4	1	2.3730	

```
…     …    …     …                …     …    …     …
2977     3     3  1.9418        232.037173     3     3  1.9418
2978     5     4  2.9800        352.099965     5     4  2.9800
2980     3     2  2.2448        246.052823     3     2  2.2448
2981     6     0 -5.3440        331.940118     6     0 -5.3440
2982     4     2  1.4580        301.007864     4     2  1.4580

     ro5_fulfilled
14            True
16            True
18            True
22            True
23            True
…              …
2977          True
2978          True
2980          True
2981          True
2982          True

[1992 rows x 16 columns]
```

2.1 Save Lipinski Compounds

[9]:
```
molecules.drop('ROMol', axis=1, inplace=True)
molecules.to_csv(DATA / "PTP1B_compounds_lipinski.csv")
```

[ ]:
```

```