2_IntroToChemoinformatics

October 23, 2019

1 Introduction to cheminformatics

Andrea Volkamer

Basic handling of molecules

- Reading & writing of molecules
- Molecular descriptors & fingerprints
- Molecular similarity

Using RDKit: open source cheminformatics software More information can be found here:

- http://www.rdkit.org/docs/index.html
- http://www.rdkit.org/docs/api/index.html

[1]:
```
# The majority of the basic molecular functionality is found in module rdkit.
 ↪Chem library
from rdkit import Chem
from rdkit.Chem import AllChem
```

1.1 Representation of molecules

1.1.1 SMILES (Simplified Molecular Input Line Entry Specification)

- Atoms are represented by atomic symbols: C, N, O, F, S, Cl, Br, I
- Double bonds are =, triple bonds are #
- Branching is indicated by parenthesis
- Ring closures are indicated by pairs of matching digits

More information can be found here: http://www.daylight.com/dayhtml/doc/theory/theory.smiles.html

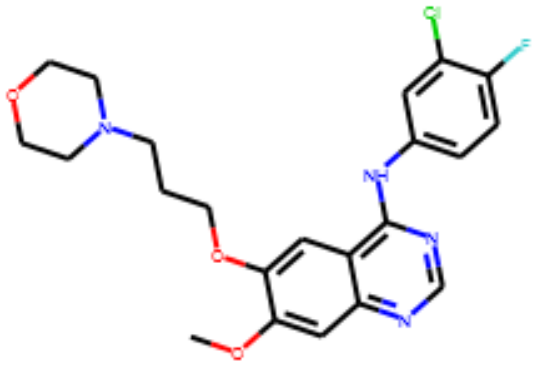
[2]:
```
# Individual molecules can be constructed using a variety of approaches
# FDA approved EGFR inhibitors: Gefitinib, Erlotinib

mol1 = Chem.MolFromSmiles('COc1cc2ncnc(Nc3ccc(F)c(Cl)c3)c2cc1OCCCN1CCOCC1')
mol2 = Chem.MolFromSmiles('C#Cc1cccc(Nc2ncnc3cc(OCCOC)c(OCCOC)cc23)c1')
```

Drawing molecules

[3]:
```
from rdkit.Chem.Draw import IPythonConsole
from rdkit.Chem import Draw
```

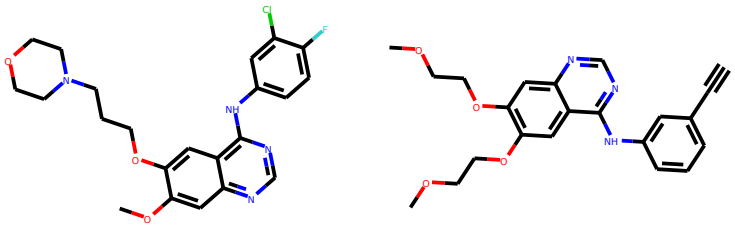
[4]:
```
# Single molecule
mol1
```

[4]:



[5]:
```
# List of molecules
Draw.MolsToGridImage([mol1,mol2], useSVG=True)
```

[5]:



Molecule representation

[6]:
```
# Molecule representation
print(Chem.MolToMolBlock(mol1))
```

```
     RDKit          2D

 31 34  0  0  0  0  0  0  0  0999 V2000
    0.7500   -6.4952    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
    1.5000   -5.1962    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
```

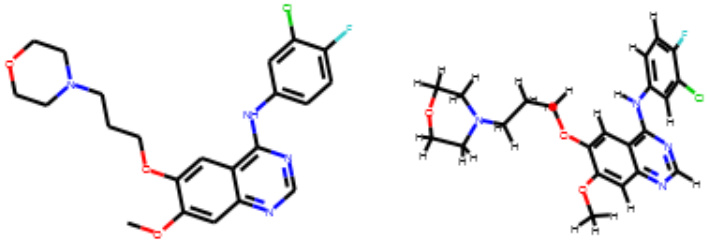
```
   0.7500   -3.8971    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   1.5000   -2.5981    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   0.7500   -1.2990    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   1.5000    0.0000    0.0000 N   0  0  0  0  0  0  0  0  0  0  0  0
   0.7500    1.2990    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -0.7500    1.2990    0.0000 N   0  0  0  0  0  0  0  0  0  0  0  0
  -1.5000    0.0000    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -3.0000    0.0000    0.0000 N   0  0  0  0  0  0  0  0  0  0  0  0
  -3.7500    1.2990    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -3.0000    2.5981    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -3.7500    3.8971    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -5.2500    3.8971    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -6.0000    5.1962    0.0000 F   0  0  0  0  0  0  0  0  0  0  0  0
  -6.0000    2.5981    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -7.5000    2.5981    0.0000 Cl  0  0  0  0  0  0  0  0  0  0  0  0
  -5.2500    1.2990    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -0.7500   -1.2990    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -1.5000   -2.5981    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -0.7500   -3.8971    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -1.5000   -5.1962    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
  -3.0000   -5.1962    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -3.7500   -6.4952    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -5.2500   -6.4952    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -6.0000   -7.7942    0.0000 N   0  0  0  0  0  0  0  0  0  0  0  0
  -7.5000   -7.7942    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -8.2500   -9.0933    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -7.5000  -10.3923    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
  -6.0000  -10.3923    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -5.2500   -9.0933    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
 1  2  1  0
 2  3  1  0
 3  4  2  0
 4  5  1  0
 5  6  2  0
 6  7  1  0
 7  8  2  0
 8  9  1  0
 9 10  1  0
10 11  1  0
11 12  2  0
12 13  1  0
13 14  2  0
14 15  1  0
14 16  1  0
16 17  1  0
16 18  2  0
 9 19  2  0
19 20  1  0
```

```
 20 21  2  0
 21 22  1  0
 22 23  1  0
 23 24  1  0
 24 25  1  0
 25 26  1  0
 26 27  1  0
 27 28  1  0
 28 29  1  0
 29 30  1  0
 30 31  1  0
 21  3  1  0
 31 26  1  0
 19  5  1  0
 18 11  1  0
M  END
```

1.1.2 Generating 3D coordinates

[7]:
```
m_3D = Chem.AddHs(mol1)
AllChem.EmbedMolecule(m_3D)
#AllChem.UFFOptimizeMolecule(m_3D) # Improves the quality of the conformation; this step should not be necessary since v2018.09: default conformations use ETKDG
Draw.MolsToGridImage([mol1,m_3D])
```

[7]:



[8]:
```
print(Chem.MolToMolBlock(m_3D))
```

```
     RDKit          3D

 55 58  0  0  0  0  0  0  0  0999 V2000
    1.0048   -4.1973    2.6656 C   0  0  0  0  0  0  0  0  0  0  0  0
```

```
 0.5503   -2.9338    2.2181 O   0  0  0  0  0  0  0  0  0  0  0  0
 1.2439   -2.2646    1.2038 C   0  0  0  0  0  0  0  0  0  0  0  0
 2.3643   -2.8202    0.6375 C   0  0  0  0  0  0  0  0  0  0  0  0
 3.0797   -2.1758   -0.3787 C   0  0  0  0  0  0  0  0  0  0  0  0
 4.1727   -2.7529   -0.9039 N   0  0  0  0  0  0  0  0  0  0  0  0
 4.8813   -2.1500   -1.8825 C   0  0  0  0  0  0  0  0  0  0  0  0
 4.4701   -0.9614   -2.3192 N   0  0  0  0  0  0  0  0  0  0  0  0
 3.3991   -0.2992   -1.8677 C   0  0  0  0  0  0  0  0  0  0  0  0
 3.0237    0.9636   -2.3435 N   0  0  0  0  0  0  0  0  0  0  0  0
 3.6356    1.7267   -3.3724 C   0  0  0  0  0  0  0  0  0  0  0  0
 3.4448    3.1188   -3.3917 C   0  0  0  0  0  0  0  0  0  0  0  0
 3.9688    3.9430   -4.3436 C   0  0  0  0  0  0  0  0  0  0  0  0
 4.7273    3.3689   -5.3400 C   0  0  0  0  0  0  0  0  0  0  0  0
 5.2731    4.1376   -6.3130 F   0  0  0  0  0  0  0  0  0  0  0  0
 4.9382    2.0037   -5.3574 C   0  0  0  0  0  0  0  0  0  0  0  0
 5.8929    1.2638   -6.6114 Cl  0  0  0  0  0  0  0  0  0  0  0  0
 4.3868    1.1821   -4.3661 C   0  0  0  0  0  0  0  0  0  0  0  0
 2.6814   -0.9478   -0.8523 C   0  0  0  0  0  0  0  0  0  0  0  0
 1.5581   -0.3883   -0.2870 C   0  0  0  0  0  0  0  0  0  0  0  0
 0.8465   -1.0430    0.7336 C   0  0  0  0  0  0  0  0  0  0  0  0
-0.2622   -0.4487    1.2648 O   0  0  0  0  0  0  0  0  0  0  0  0
-0.8083    0.7972    0.8890 C   0  0  0  0  0  0  0  0  0  0  0  0
-2.0244    1.0560    1.7607 C   0  0  0  0  0  0  0  0  0  0  0  0
-3.0582   -0.0168    1.6055 C   0  0  0  0  0  0  0  0  0  0  0  0
-4.2127    0.2027    2.4334 N   0  0  0  0  0  0  0  0  0  0  0  0
-4.9815    1.3159    2.0070 C   0  0  0  0  0  0  0  0  0  0  0  0
-6.1141    1.6583    2.9507 C   0  0  0  0  0  0  0  0  0  0  0  0
-6.4800    0.5661    3.7333 O   0  0  0  0  0  0  0  0  0  0  0  0
-6.3573   -0.6325    3.0417 C   0  0  0  0  0  0  0  0  0  0  0  0
-4.9244   -0.9950    2.7314 C   0  0  0  0  0  0  0  0  0  0  0  0
 0.2170   -4.7367    3.2392 H   0  0  0  0  0  0  0  0  0  0  0  0
 1.9729   -4.1338    3.1791 H   0  0  0  0  0  0  0  0  0  0  0  0
 1.1702   -4.8466    1.7550 H   0  0  0  0  0  0  0  0  0  0  0  0
 2.7291   -3.7918    0.9723 H   0  0  0  0  0  0  0  0  0  0  0  0
 5.7498   -2.6707   -2.2602 H   0  0  0  0  0  0  0  0  0  0  0  0
 2.1816    1.4339   -1.9051 H   0  0  0  0  0  0  0  0  0  0  0  0
 2.8408    3.5514   -2.5917 H   0  0  0  0  0  0  0  0  0  0  0  0
 3.7956    5.0197   -4.3204 H   0  0  0  0  0  0  0  0  0  0  0  0
 4.5696    0.1298   -4.4393 H   0  0  0  0  0  0  0  0  0  0  0  0
 1.2128    0.5693   -0.6316 H   0  0  0  0  0  0  0  0  0  0  0  0
-1.0095    0.8496   -0.1808 H   0  0  0  0  0  0  0  0  0  0  0  0
-0.0363    1.5841    1.1298 H   0  0  0  0  0  0  0  0  0  0  0  0
-2.4184    2.0406    1.4282 H   0  0  0  0  0  0  0  0  0  0  0  0
-1.7003    1.1528    2.7961 H   0  0  0  0  0  0  0  0  0  0  0  0
-3.3573   -0.1002    0.5192 H   0  0  0  0  0  0  0  0  0  0  0  0
-2.5764   -1.0109    1.8171 H   0  0  0  0  0  0  0  0  0  0  0  0
-4.3366    2.2132    1.9615 H   0  0  0  0  0  0  0  0  0  0  0  0
-5.3763    1.1907    0.9534 H   0  0  0  0  0  0  0  0  0  0  0  0
```

```
  -5.8803    2.5575    3.5541 H   0  0  0  0  0  0  0  0  0  0  0  0
  -6.9972    1.9098    2.3290 H   0  0  0  0  0  0  0  0  0  0  0  0
  -6.9390   -0.5612    2.1242 H   0  0  0  0  0  0  0  0  0  0  0  0
  -6.7409   -1.4333    3.7102 H   0  0  0  0  0  0  0  0  0  0  0  0
  -4.4644   -1.5018    3.6077 H   0  0  0  0  0  0  0  0  0  0  0  0
  -4.9268   -1.6927    1.8782 H   0  0  0  0  0  0  0  0  0  0  0  0
 1  2  1  0
 2  3  1  0
 3  4  2  0
 4  5  1  0
 5  6  2  0
 6  7  1  0
 7  8  2  0
 8  9  1  0
 9 10  1  0
10 11  1  0
11 12  2  0
12 13  1  0
13 14  2  0
14 15  1  0
14 16  1  0
16 17  1  0
16 18  2  0
 9 19  2  0
19 20  1  0
20 21  2  0
21 22  1  0
22 23  1  0
23 24  1  0
24 25  1  0
25 26  1  0
26 27  1  0
27 28  1  0
28 29  1  0
29 30  1  0
30 31  1  0
21  3  1  0
31 26  1  0
19  5  1  0
18 11  1  0
 1 32  1  0
 1 33  1  0
 1 34  1  0
 4 35  1  0
 7 36  1  0
10 37  1  0
12 38  1  0
13 39  1  0
```

```
 18 40  1  0
 20 41  1  0
 23 42  1  0
 23 43  1  0
 24 44  1  0
 24 45  1  0
 25 46  1  0
 25 47  1  0
 27 48  1  0
 27 49  1  0
 28 50  1  0
 28 51  1  0
 30 52  1  0
 30 53  1  0
 31 54  1  0
 31 55  1  0
M  END
```

1.1.3 Writing molecules to *sdf* (structure data files)

[9]:
```
w = Chem.SDWriter('./data/mytest_mol3D.sdf')
w.write(m_3D)
w.close()
```

1.1.4 Descriptors

Molecular descriptors (global)

[10]:
```
from rdkit.Chem import Descriptors
```

[11]:
```
print ('Heavy atoms:', Descriptors.HeavyAtomCount(mol1))
print ('H-bond donors:', Descriptors.NumHDonors(mol1))
print ('H-bond acceptors:', Descriptors.NumHAcceptors(mol1))
print ('Molecular weight:', Descriptors.MolWt(mol1))
print ('LogP:', Descriptors.MolLogP(mol1))
```

```
Heavy atoms: 31
H-bond donors: 1
H-bond acceptors: 7
Molecular weight: 446.9100000000004
LogP: 4.275600000000003
```

[12]:
```
print ('Heavy atoms:', Descriptors.HeavyAtomCount(mol2))
print ('H-bond donors:', Descriptors.NumHDonors(mol2))
print ('H-bond acceptors:', Descriptors.NumHAcceptors(mol2))
print ('Molecular weight:', Descriptors.MolWt(mol2))
print ('LogP:', Descriptors.MolLogP(mol2))
```

```
Heavy atoms: 29
H-bond donors: 1
H-bond acceptors: 7
Molecular weight: 393.4430000000002
LogP: 3.405100000000002
```

Better for similarity search: Molecular fingerprints GL: I don't see any utility in introducing MACCS keys to beginners. These days I think they are primarily of historic interest and it would be better to use something the Morgan FP or RDKit FP here. Or, if you want something simple, atom pairs/topological torsions. If you're going to use them, you might as well pull the SMARTS from the RDKit directly instead of retyping them in. I've changed that.

MACCS keys

- There is a SMARTS-based implementation of the 166 public MACCS keys (certain substructure/SMARTS keys which are expected to be found)
- The MACCS keys are a set of questions about a chemical structure
- Based on counting substructural features

[13]:
```
# Example MACCS keys
from rdkit.Chem import MACCSkeys

smarts = [MACCSkeys.smartsPatts[x][0] for x in (132, 133, 135)]
print(smarts)

mols = [Chem.MolFromSmarts(x) for x in smarts]

# A detail: get the molecules ready to be drawn:
for m in mols:
    m.UpdatePropertyCache()

Draw.MolsToGridImage(mols)
```

```
['[#8]~*~[CH2]~*', '*@*!@[#7]', '[#7]!:*:*']
```

[13]:

[14]:
```
# Calculation of MACCS fingerprint

fp1 = MACCSkeys.GenMACCSKeys(mol1)
fp2 = MACCSkeys.GenMACCSKeys(mol2)
```

[15]:
```
fp1.ToBitString()
```

[15]:
```
'00000000000000000000000000000000000001000100000000000010000000100000010010100
10000011100000101011011010101010101000010111001111100111101100011111110111011111
11111110'
```

[16]:
```
fp2.ToBitString()
```

[16]:
```
'00000000000000000100000000000000000001000000000000000000000000000010000001000010
01000001000000100011001000100010001011010110001110000110101101011111101111101111
11111110'
```

1.1.5 Molecular similarity

[17]:
```
from rdkit import DataStructs
```

[18]:
```
# Tanimoto
commonBits = fp1&fp2
print('fp1:',fp1.GetNumOnBits(),'fp2:',fp2.GetNumOnBits(),'num in common:
 ↪',commonBits.GetNumOnBits())
print(commonBits.GetNumOnBits()/(fp1.GetNumOnBits()+fp2.
 ↪GetNumOnBits()-commonBits.GetNumOnBits()))
print('Tanimoto:', DataStructs.TanimotoSimilarity(fp1,fp2))
```

```
fp1: 60 fp2: 50 num in common: 45
0.6923076923076923
Tanimoto: 0.6923076923076923
```

[ ]:

[ ]: