DOI: 10.15255/KUI.2017.052
KUI-26/2018
Stručni rad
Prispjelo 21. prosinca 2017.
Prihvaćeno 1. svibnja 2018.

Molekulsko modeliranje odnosa strukturnih svojstava i aktivnosti molekula s pomoću programskog jezika Python (prvi dio)

M. Lovrić[a,b*]

[a] Know-Center, Inffeldgasse 13/6, 8010 Graz, Austrija
[b] Centar za NMR, Institut Ruđer Bošković, Bijenička cesta 54, 10 000 Zagreb

Sažetak

Danas se količina podataka znatno povećava, a podatcima se pridaje sve veća vrijednost, kao i poznavanju njihove manipulacije i crpljenja vrijednih informacija. Poznat primjer crpljenja informacija je pretraživanje poznatih kemijskih spojeva i dizajniranje novih spojeva na osnovi znanja iz modela u svrhu istraživanja potencijalnih lijekova. Stoga je studentu kemije važno biti dobro pripremljen za trenutačno digitalno doba, gdje nije više dovoljno biti samo spretan u laboratoriju, nego je potrebno znati modelirati i raditi s podatcima. Ovaj rad pokriva osnove molekulskog modeliranja i QSAR-a te osnove rukovanja podatcima pomoću besplatnog programskog jezika Python i njegove biblioteke za molekulsko modeliranje RDKit. Ostale Pythonove biblioteke koje će se primjenjivati u radu su Pandas, za rukovanje i obradu svih vrsta podataka; statsmodels, Numpy, Scipy i SKLearn za matematičke i statističke operacije te linearnu algebru i Matplotlib i Seaborn za ispisivanje grafova. Programski jezik Python je sa svojim navedenim bibliotekama integriran u program Anaconda. Anaconda korisniku omogućuje jednostavnu primjenu i upravljanje bibliotekama te upotrebu sučelja Jupyter Notebook za programiranje i ispis grafičkih prikaza i rezultata analiza. U ovom, prvom dijelu rada analizirat će se problem predviđanja topljivosti u vodi na skupu organskih kemijskih spojeva pomoću univarijatne linearne regresije. Cilj rada je približiti kemičarima programiranje u jeziku Python, primjenu njegovih biblioteka i praktično rješavanje problema u molekulskom modeliranju.

Ključne riječi

QSAR, Python, Jupyter Notebook, molekulsko modeliranje, RDKit

Uvod

Današnjem kemičaru više nije dovoljna samo spretnost u laboratoriju, nego se u svijetu sve veće količine podataka[1] javlja potreba za poznavanjem programiranja, pretraživanja baza i modeliranja. Učestala ključna riječ u današnjem svijetu znanosti je izraz *big data*.[2] Primjer *big data* za slučaj kemijskih podataka i aktivnosti je baza PubChem[3], koja trenutačno sadrži preko 90 milijuna spojeva.[4] Ozbiljno pretraživanje zahtijeva poznavanje osnova programiranja, baza i obrade podataka, a za izvlačenje znanja potrebno je poznavati modeliranje i domenu željenih svojstava. Ovaj prvi u serijalu radova namijenjen je prije svega studentima i doktorandima koji žele započeti svoj put u svijetu molekulskog modeliranja, ali i kemometrike primjenjive na laboratorijskim eksperimentima u znanosti i industriji. Znanje i praksa stečeno u serijalu radova vezano uz obradu podataka i samo modeliranje lako je prenosivo u bilo koji kemijski problem gdje korisnik raspolaže s mogućnošću numeričkog modeliranja. U serijalu radova proći će se kroz put same obrade podataka, računanja molekulskih deskriptora i matematičkog modeliranja te osnove programskog jezika Python[5] i njegovih programskih biblioteka koje su dobro dokumentirane i za koje postoji velik broj priručnika i aktivna zajednica. Korisnicima iz industrije bit će posebno zanimljivo jer svoje podatke mogu analizirati u besplatnim softverskim alatima bez plaćanja skupih licencija. Teorijske osnove pojedinih kemometričkih metoda, grafičkih prikaza i statističkih alata kao što su linearna regresija, *F*- i *p*-vrijednosti dobro su opisane u literaturi na hrvatskom[6,7] i engleskom jeziku[8] i nisu u opsegu ovog rada. Ovo nije prvi rad ovakve vrste iz područja kemoinformatike, jedan opširniji koncept priručnika objavljen je na engleskom jeziku.[9]

Uvod u modeliranje QSAR

Molekulsko modeliranje odnosa strukturnih svojstava i aktivnosti ili QSAR (engl. *quantitative structure–activity relationship*) niz je radnji koje imaju cilj predvidjeti neku aktivnost molekula poznavajući njezina strukturna svojstva.[10] Potreba za modeliranjem nastaje iz činjenice da je poznat velik broj kemijskih spojeva za koje još nisu izmjerene eksperimentalne vrijednosti mogućih aktivnosti. Primjer takvih aktivnosti su fizikalno-kemijska svojstva kao što su topljivost u vodi log*S* i particijski koeficijent oktanol-voda[11] log*P*. Modelirati se također mogu i biološke aktivnosti kao antivirusne i antitumorske.[12] Pomoću molekulskog modeliranja mogu se uštedjeti vrijeme i sredstva za ispitivanje velikog broja spojeva kao i cijelih baza poput PubChema. Strukturna svojstva opisuju se deskriptorima. Deskriptori mogu biti jednodimenzionalni te dvo-, trodimenzionalni ili višedimenzionalni. Primjer 1D bio bi broj nekih funkcionalnih skupina kao što je skupina −OH, a primjer 2D deskriptora bili bi topološki indeksi derivirani iz matrica udaljenosti atoma. Potrebno je znati koja aktivnost se želi računati, tzv. ciljna varijabla. U ovom radu to je topljivost u vodi, izražena kao log*S*. Za dobivanje funkcije koja prevodi strukturno ili neko drugo fizikalno-kemijsko svojstvo u aktivnost, potrebno je imati skup molekula koje već imaju traženu izmjerenu aktivnost kako bi model mogao učiti (trenirati) iz poznatih strukturnih parametara

* Mario Lovrić, e-pošta: mlovric@know-center.at

i poznate aktivnosti, tj. odrediti kakva je to funkcija koja prevodi deskriptore u aktivnost. Zatim se model može vrednovati (provjeriti) na dodatnom skupu mjerenih aktivnosti. Obvezatne predradnje su priprema skupa spojeva u smislu pregleda struktura i pretvaranja u potrebne formate prihvatljive računalima, kao što su zapisi SMILES[13] i MOL[14]. Zapis molekula SMILES (engl. *simplified molecular-input line-entry system*) jednodimenzionalni je kodirani zapis ASCII koji prihvaća gotovo sav kemijski softver. Primjer takvog zapisa za molekulu askorbinske kiseline je "OC[C@H](O)[C@H]1OC(=O)C(O)=C1O". Zapis MOL je tablični zapis strukture koji sadrži informacije o atomima, vezama među njima, a može pospremiti i koordinate atoma u dvo- i trodimenzionalnom prostoru. Za slučaj ovog rada uzeta je već pripremljena baza kemijskih struktura i mjerene topljivosti iz literature.[15,16] U nastavcima ovog serijala radova bit će više riječi o pripremi skupa spojeva i drugim kemometričkim metodama.

Python

Python je skriptni objektno-orijentirani programski jezik visoke razine. Njegove velike prednosti su otvoren kod, iznimno čitka sintaksa, cijena (besplatan je), velika zajednica korisnika i time razvijene programske biblioteke. Neke od Pythonovih biblioteka kao što su Pandas,[17] Scipy,[18] statsmodels[19] i SKLearn[20] posebno su namijenjene obradi podataka i statističkoj analizi. Za grafičke prikaze ponajviše se primjenjuju programske biblioteke Matplotlib[21] i Seaborn.[22] Na internetskim stranicama svake od biblioteka mogu se naći primjeri i upute za primjenu. Slike i grafovi u ovom radu odgovaraju ispisanom kodu, a za dodatnu obradu i dekoracije korisnik se može informirati u dokumentaciji pojedine biblioteke. Na internetskoj stranici https://stackoverflow.com mogu se postaviti pitanja u vezi s kodom i naći mnoga gotova rješenja. Za svrhu modeliranja QSAR primijenit će se biblioteka RDKit,[23] koja je također dobro dokumentirana. Internetska stranica biblioteke http://www.rdkit.org/ nudi uvodni materijal i popis metoda za molekulsko modeliranje. Bitan koncept u Pythonu je da je sve objekt. Objekti imaju metode (funkcije koje se vrše nad objektima), klase (skupine metoda) i atribute (svojstva). U programskom kodu u ovom radu često će se pozivati metode nad pojedinim objektima.

Instalacija okruženja Python

Anaconda je besplatna distribucija Pythona za analizu podataka. Uz intuitivno razvojno okruženje pogodna je i za upravljanje programskim bibliotekama. Prije instalacije korisno je provjeriti koji je operativni sustav instaliran na računalu (32- ili 64-bit). Instalacijske upute za računala Windows nalaze se na adresi https://docs.anaconda.com/anaconda/install/windows. Pri samom preuzimanju potrebno je odabrati verziju Pythona, 3.6 za 64-bitna računala Windows te verziju 2.7 za 32-bitna računala. Anacondu je potrebno instalirati. Nakon završene instalacije, potrebno je instalirati biblioteke koje nedostaju. Među programima na računalu potrebno je pronaći "Anaconda Prompt". Kod klasičnih izbornika Windows to se čini odlaskom na *Start izbornik -> Programi -> Anaconda3 -> Anaconda Prompt*. Nakon toga otvorit će se komandno sučelje. Na primjeru instalacije biblioteke RDKit koja služi za molekulsko modeliranje vidjet će se način instalacije programskih biblioteka. U Anaconda Prompt potrebno je upisati "conda install -c rdkit rdkit" te pritisnuti tipku *enter*. Ukoliko se pojavi upit "Proceed ([y]/n)?", potrebno je upisati slovo "y" i pritisnuti tipku *enter*, slika 1. Biblioteka će potom biti instalirana, što može potrajati. Nakon instalacije sučelje je potrebno zatvoriti.

```
(C:\Users\mr.T\Anaconda3) C:\Users\mr.T>conda install -c rdkit rdkit
Fetching package metadata .............
Solving package specifications: .

Package plan for installation in environment C:\Users\mr.T\Anaconda3:

The following NEW packages will be INSTALLED:

    boost: 1.59.0-py36_3    rdkit
    rdkit: 2017.09.1-py36_1 rdkit
    vc:    14-0

The following packages will be UPDATED:

    conda: 4.3.21-py36_0          --> 4.3.30-py36h7e176b0_0

Proceed ([y]/n)? y
```

Slika 1 – Primjer instalacije biblioteke RDKit u naredbenom sučelju

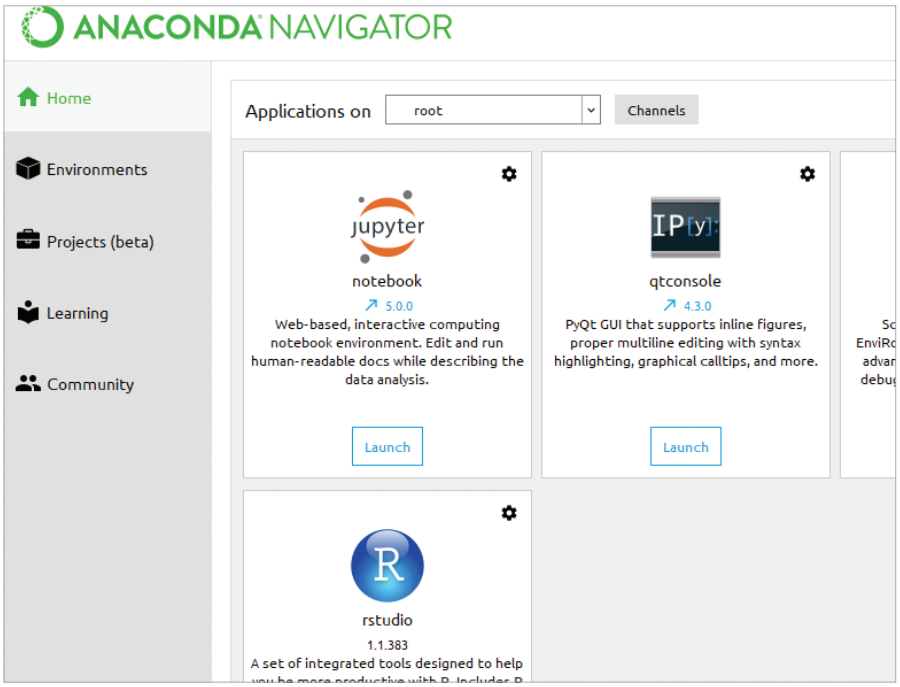
Fig. 1 – Screenshot of RDKit library installation in the command window

Pokretanje okruženja Python

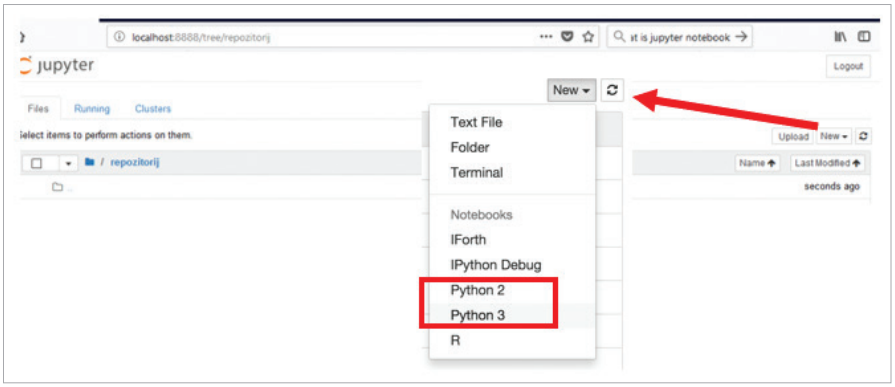
Korisniku se prije pokretanja programa Anaconda preporučuje stvaranje novog radnog pretinca u kojem će pospremati svoj budući rad i podatke. Preporuka je otvoriti pretinac s nazivom "repozitorij", zatim u njemu pretinac s nazivom "data". Radi jednostavnosti, preporuka je taj repozitorij otvoriti na adresi **"C:\Users\<ime korisnika>".** Adresa pretinca "data" će dakle biti **"C:\Users\<ime korisnika>\repozitorij\data"**, a podatci skupa molekula za modeliranje i već pripremljeni *notebook* su na adresi https://github.com/mariolovric/modeliranje-tutorial. Podatke za praktični primjer, datoteku s adrese **"solubility.txt"** potrebno je preuzeti s poveznice i spremiti u data pretinac.

Nakon instalacije potrebno je pokrenuti Anaconda Navigator. Kod klasičnih izbornika Windows to se čini odlaskom na **Start -> Program -> Anaconda3 -> Anaconda Navigator**. Potom se otvara sučelje programa koje izgleda kao na slici 2.

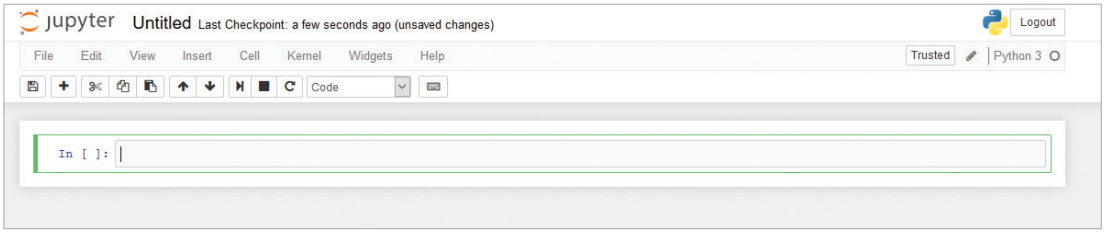
Potrebno je izabrati okruženje Jupyter Notebook, koje će se nakon odabira opcije "Launch" otvoriti u postavljenom internetskom pregledniku. Jupyter Notebook je interaktivno okruženje u obliku internetske aplikacije za pisanje i izvršavanje koda. Nakon što se Jupyter Notebook automatski otvori u internetskom pregledniku (stranica s adresom http://localhost:8888/tree), potrebno je izabrati radni pretinac (npr. prethodno stvoreni repozitorij). Sučelje bi trebalo izgledati kao na slici 3. Za stvaranje novog *notebooka* potrebno je na kontrolnoj ploči u desnom gornjem kutu izabrati opciju "New" i zatim u padajućem izborniku oda-



Slika 2 – Sučelja programa Anaconda
Fig. 2 – The Anaconda user interface



Slika 3 – Sučelje Jupyter Notebooka
Fig. 3 – The Jupyter Notebook interface



Slika 4 – Početno sučelje u aplikaciji Jupyter Notebook
Fig. 4 – Starting interface in Jupyter Notebook

brati "Python 2" ili "Python 3" ovisno o instaliranoj verziji. Otvorit će se novi *notebook* s nazivom "Untitled", slika 4.

Korisniku se preporučuje da pregleda padajuće izbornike na kontrolnoj traci i isproba mogućnosti Jupyter Notebooka. Naredbe se u *notebook* upisuju u prazna polja na sredini označena s "*In [ ]:*". Nakon upisa svih naredbi u polje, polja (engl. *cell*) se izvršavaju pritiskom na tipke *Shift* + *Enter*. Sučelje ostaje otvoreno za korake koji slijede. Bitno je točno prepisati kod iz praktičnog primjera. Pogreške će se događati, a korisniku se preporučuje da prostudira ispis pogrešaka i sam pokuša ući u trag izvoru pogreške.

Osnove programiranja u Pythonu Jupyter Notebooka

Učitavanje programskih biblioteka

Za obradu podataka i rad s kemijskim strukturama potrebno je pozvati nekoliko Pythonovih programskih biblioteka koje su uključene u Anacondi. Popis dodatnih biblioteka može se naći na https://anaconda.org/anaconda/repo. Numpy je biblioteka za rad s matricama i algebarskim operacijama, u ovom serijalu primijenit će se za pretvaranje iz matrica i u matrice (*numpy array*). Pandas je biblioteka za rad s podatkovnim okvirima (*pandas dataframe*), strukturama za tablično skladištenje svih vrsta podataka, ne samo numeričkih. Mogu biti i jednostupčani tablični zapisi tzv. podatkovni vektor (*pandas series)*. Podatkovni okviri sastoje se od podataka, indeksa (rednih brojeva redaka) i stupaca (*column*) i njima se vrlo lako može manipulirati i pretraživati ih. Stupci i redci mogu se izdvajati u podatkovne vektore metodama za indeksiranje.

Eksperimentalni dio

Praktični primjer

Za učitavanje biblioteka primjenjuje se naredba *import*, a uz pomoćnu naredbu *as* mogu se definirati kratice pojedine biblioteke. Kompletni kod za upis i izvršavanje upisan je u poljima teksta ispisanim kurzivom. Potrebno je učitati biblioteke Numpy, Pandas i RDKit.

```
# In[1]: import pandas as pd
import numpy as np
from rdkit import Chem
from rdkit.Chem import AllChem, Draw, Descriptors
```

Učitavanje i pregled sirovih podataka

Podatci se učitavaju metodom biblioteke Pandas *read_csv*, koja učitava datoteku .csv u podatkovni okvir koji se dodjeljuje objektu *sirovi_podaci*. Postoji niz argumenata koji se mogu proslijediti toj metodi, koji specificiraju način učitavanja, a mogu se pogledati u dokumentaciji.

Argumenti za metodu *read_csv* su adresa datoteke **‚data/solubility.txt‘**, *delim_whitespace* koji definira kako tretirati prazna polja u podatcima i *names* koji definira listu u uglatim zagradama s nazivima stupaca podatkovnog okvira pisanih u navodnim znakovima koje znamo iz pregleda sirovih podataka. Podatkovni okviri kao objekti u Pythonu imaju pripadajuće metode. Jedna od njih je *head()* koja prikazuje sve stupce prvih pet redova podatkovnog okvira.

```
# In[2]: sirovi_podaci=pd.read_csv(‚data/
solubility.txt‘,\
delim_whitespace=True,names=[‚cas‘,‘smiles‘,‘lo
gs‘])
# In[3]: sirovi_podaci.head()
```

Tablica 1 – Ispis polja In[3]
Table 1 – Output result of In[3]

	cas	smiles	logs
0	60-35-5	CC(N)=O	1,6
1	60-34-4	CNN	1,3
2	64-19-7	CC(O)=O	1,2
3	123-75-1	C1CCCN1	1,2
4	127-07-1	NC([NH]O)=O	1,1

Pretvaranje zapisa SMILES u MOL

Podatkovni okvir kao objekt ima metodu *apply()* koja primjenjuje neku funkciju ili metodu na retke ili stupce podatkovnih okvira. Za pretvaranje zapisa SMILES iz datoteke u MOL potrebnog za daljnje računanje, pozvat će se metoda *apply()* koja će primijeniti metodu *Chem.MolFromSmiles()* iz biblioteke RDKit. Za lakše upravljanje podatcima u nastavku je predana metoda *rename()* koja preimenuje vektor sa zapisom molekula MOL u naziv "mol". Argument *inplace=True* primjenjuje se kad se želi zamijeniti postojeći objekt na kojem se izvodi operacije novim.

```
# In[4]:
mol_column=sirovi_podaci.smiles.apply(Chem.
MolFromSmiles).rename(‚mol‘, inplace=True)
```

Računanje deskriptora

Za računanje deskriptora na skupu molekula u zapisu MOL upotrebljava se modul biblioteke RDKit *Descriptors* kojim se računa niz deskriptora. Za svrhu ovog rada računat će se deskriptori iz niza koje modul nudi, a mogu se naći u dokumentaciji. To su particijski koeficijent (*Descriptors.MolLogP*), molekulska masa (*Descriptors.MolWt*), topološki Balabanov indeks (*Descriptors.BalabanJ*) i ukupna polarna površina molekule (*Descriptors.TPSA*). Za računanje deskriptora ponovno se primjenjuje metoda *apply()*, koja u ovom slučaju zadaje metode za računanje deskriptora za pojedinu molekulu u stupcu i sprema ih u nove podatkovne vektore s novim nazivom (*‚logp‘, ‚molwt‘, ‚balabanj‘, ‚TPSA‘*).

```
# In[5]:
logp=mol_column.apply(Descriptors.MolLogP).
rename(‚logp‘, inplace=True)
molwt=mol_column.apply(Descriptors.MolWt).
rename(‚molwt‘, inplace=True)
balabanj=mol_column.apply(Descriptors.BalabanJ).
rename(‚balabanj‘, inplace=True)
tpsa=mol_column.apply(Descriptors.TPSA).
rename(‚tpsa‘, inplace=True)
```

Spajanje sirovih podataka i izračunatih deskriptora

Nakon što su izračunati svi potrebni molekulski deskriptori, pojedine stupce s deskriptorima korisno je združiti u jedan jedinstven podatkovni okvir. To se čini metodom *concat()*, kojoj je potrebno predati popis vektora koji se združuju u uglatoj zagradi i os po kojoj se vektori spajaju. U ovom slu-

Tablica 2 – Ispis polja In[6]
Table 2 – Output result of In[6]

	cas	smiles	logs	mol	logp	molwt	balabanj	tpsa
0	60-35-5	CC(N)=O	1,58	<rdkit.Chem.rdchem.Mol object...	−0,51	59,07	2,80	43,09
1	60-34-4	CNN	1,34	<rdkit.Chem.rdchem.Mol object...	−0,92	46,07	1,63	38,05
2	64-19-7	CC(O)=O	1,22	<rdkit.Chem.rdchem.Mol object...	0,09	60,05	2.80	37,30
3	123-75-1	C1CCCN1	1,15	<rdkit.Chem.rdchem.Mol object...	0,37	71,12	2,08	12,03

čaju za horizontalno spajanje vektora to je os (*axis*) 1. Moguće je spremiti novi podatkovni okvir nazvan *final_data* u csv datoteku metodom *to_csv()*.

```
# In[6]:
final_data=pd.concat([sirovi_podaci, mol_column,
logp, molwt, balabanj, tpsa], axis=1)
final_data.head(4)
# In[]:final_data.to_csv(,data/solubility_all_
data.csv‘)
```

Razdvajanje podataka na skupove za učenje i vrednovanje modela

U modeliranju je korisno učiti modele koji imaju primjenu i izvan skupa ili skupa na kojem se uči, tj. pokušati učiti što univerzalniji model. U tu svrhu se objekti na kojima se modelira odvajaju minimalno na skup za učenje (engl. *training set*) i skup za vrednovanje modela (engl. *test set*), kako bi se mogao provjeriti model na nepoznatim mu podatcima. Biblioteka SKLearn ima metodu za odvajanje skupova, *train_test_split*. Argumenti za metodu su podatkovni okvir koji se razdvaja (*final_data*), udio skupa za vrednovanje u podatcima *test_size* (0,2 ili 20 %) i *random_state* koji osigurava da podjela skupa bude jednaka u svakoj iteraciji. Metodom *shape* ispisuju se dimenzije podatkovnih okvira za oba skupa. U skupu za učenje je 1048 molekula, a u skupu za vrednovanje su 263 molekule.

```
# In[7]:
from sklearn.model_selection import train_test_
split
train_set,test_set=train_test_split(final_
data,test_size=.2,random_state=42)
train_set.shape, test_set.shape
Out: ((1048, 8), (263, 8))
```

Prikaz varijabli

Preporučuje se prije svakog učenja modela ispisati i grafički prikazati varijable i njihove međusobne odnose. Za pregled korelacijske matrice traženih varijabli potrebno je izabrati stupce od interesa. Izbor varijabli definira se u uglatoj zagradi. Metodom *corr()* dobiva se ispis korelacijske matrice.

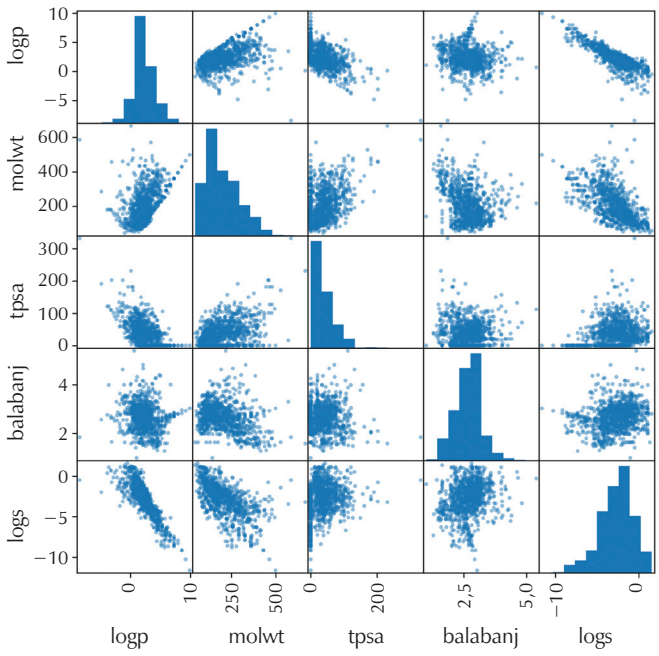
```
# In[8]:
train_
set[[,logp‘,‘molwt‘,‘tpsa‘,‘balabanj‘,‘logs‘]].
corr()
```

Tablica 3 – Ispis polja In[8]
Table 3 – Output result of In[8]

	logp	molwt	tpsa	balabanj	logs
logp	1	0,45	−0,51	−0,16	−0,84
molwt	0,45	1	0,41	−0,36	−0,64
tpsa	−0,51	0,41	1	−0,07	0,19
balabanj	−0,16	−0,36	−0,07	1	0,23
logs	−0,84	−0,64	0,19	0,23	1

Za grafički prikaz u samom *notebooku* unosi se naredba *%matplotlib inline*, koja definira ispis slika i grafova. Iz modula *pandas.plotting* učita se metoda *scatter_matrix()*, koja služi za grafički prikaz korelacijske matrice željenih varijabli i raspodjelu varijabli u matrici raspršenja (slika 5).

```
# In[9]: %matplotlib inline
from pandas.plotting import scatter_matrix
scatter_matrix(train_
set[[,logp‘,‘molwt‘,‘tpsa‘,‘balabanj‘,‘logs‘]],
figsize=(10,12))
```



Slika 5 – Matrica raspršenje svih varijabli
Fig. 5 – Scatter matrix for all variables

Argumenti metode *scatter_matrix* su podatkovni okvir s odabranim varijablama zadani u uglatoj zagradi i opcionalno argumenti; ovdje veličina slike (*figsize)*. Iz slike se može naslutiti mogućnost postojanja linearne korelacije varijabli log*S* i log*P*.

Odabir najutjecajnije predikcijske varijable za jednostruku linearnu regresiju

U slučajevima s velikim brojem varijabli koje se primjenjuju za predikciju (za rad su računate samo četiri), često se primjenjuju tehnike smanjivanja broja predikcijskih varijabli prema nekom od zadanih kriterija. Ovdje će se primijeniti odabir predikcijske varijable koja najviše korelira s ciljnom varijablom u jednostrukoj linearnoj regresiji, pomoću modula *feature_selection* biblioteke SKLearn i metode *f_regression()*. Za bolje upoznavanje funkcionalnosti pojedinih metoda i klasa može se pozvati metoda *help()*.

```
# In[10]: from sklearn.feature_selection import
f_regression
# In[11]: help(f_regression)
```

Parametri za metode iz modula *feature_selection* obvezatno su ciljna varijabla y (‚*logs*') i postojeće prediktorske varijable X (‚*logp*', ‚*molwt*', ‚*tpsa*', ‚*balabanj*'). Rezultati metode spremaju se u objekt *featsel*, a to su *F*-vrijednosti i *p*-vrijednosti korelacijskih koeficijenata. Iz ispisa drugog elementa objekta *featsel* indeksiranog pomoću uglate zagrade i broja 1 vidi se da prva varijabla log*P* najbolje opisuje ciljnu varijablu log*S* na temelju najniže *p*-vrijednosti, zato je izabrana kao predikcijska varijabla.

```
In[12]: featsel=f_regression(X=train_
set[[,logp',‘molwt',‘tpsa',‘balabanj']], y=train_
set[,logs'])
F_values=featsel[0]
P_values=featsel[1]
Out:array([5.79397574e-281, 1.94076230e-121,
1.42230093e-009, 2.57225427e-014])
```

Učenje modela

Za učenje modela univarijatne linearne regresije primjenjuje se metoda *LinearRegression* iz modula *sklearn.linear_model*, koji je potrebno učitati.

```
# In[13]:
from sklearn.linear_model import LinearRegression
as linreg
```

Podatkovni okvir, koji je prethodno podijeljen u skup za učenje i skup za vrednovanje, potrebno je prevesti u matrični zapis metodom *np.array()*. Metoda *LinearRegression* zahtijeva kao ulazni podatak matricu X (*array*) koja mora biti višedimenzionalna i y koji mora biti jednodimenzionalan. Za dodavanje dimenzija matrici prosljeđuje se metoda *np.newaxis*. Korisnika se poziva da ovdje pokuša ispisati matrice i njihove dimenzije metodom *shape*. U objekt *y_true* spremljena je matrica eksperimentalnih vrijednosti za log*S* molekula iz skupa za vrednovanje.

```
# In[14]: xtrain=np.array(train_set.logp)[:,
np.newaxis]
ytrain=np.array(train_set.logs)
xtest=np.array(test_set.logp)[:, np.newaxis]
y_true=np.array(test_set.logs)
```

Za učenje modela ulazni podatci dolaze iz skupa za učenje (*xtrain, ytrain*). Model se računa prema jednadžbi za univarijatnu linearnu regresiju. Detalji u vezi s linearnom regresijom opisani su u lit.[24] U Pythonu je sve objekt, pa stoga i regresijski model koji je dodijeljen pomoću metode *fit()* (na skupu za učenje) objektu s nazivom *fitter*. Objekt *fitter* sadrži parametre regresijskog modela kao objekte *coef_* i *intercept_* koji su dodijeljeni objektima *a* i *b* (nagib i odsječak regresijskog pravca).

```
# In[15]: fitter=linreg().fit(X=xtrain, y=ytrain)
a=fitter.coef_
b=fitter.intercept_
a, b
Out: (array([-0.95544944]), -0.55785964516979547)
```

fitter ima metodu *predict* koja se izvodi na podatcima za koje se želi izračunati željeno svojstvo. U ovom slučaju to je matrica *xtest* koja sadrži podatke za log*P* skupa za vrednovanje spojeva.

Dobiveni vektor *y_pred* sadrži izračunate, teoretske log*S* vrijednosti za skup za vrednovanje spojeva.

```
# In[16]: y_pred=fitter.predict(X=xtest)
```

Rezultati i rasprava

Grafičko prikazivanje rezultata predikcije na skupu za vrednovanje

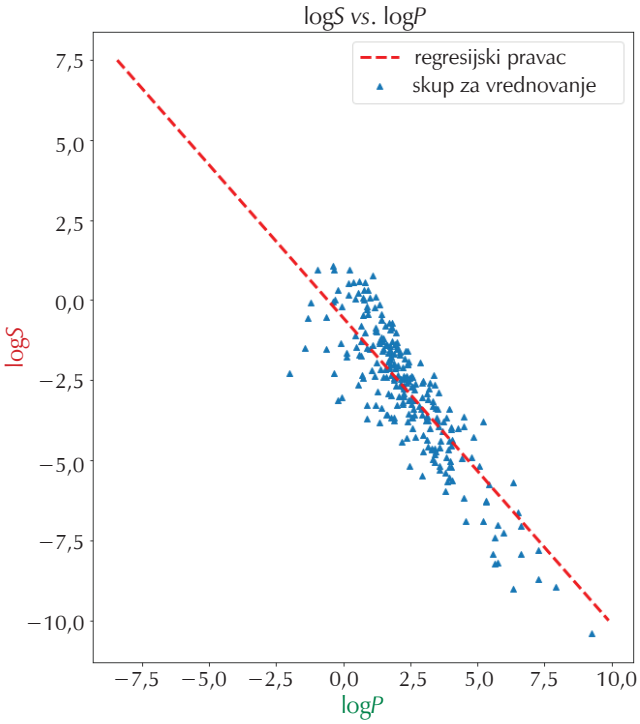
Biblioteka matplotlib poziva se s modulom *pyplot*. Radi jednostavnosti poziva se kao kratica *plt*. U *pyplotu* su definirane klase *figure* i *axes* (slika i osi). Svaki od ta dva objekta ima pripadajuća svojstva i metode. Klasa *figure* (*fig* u primjeru) služi za postavke na razini cijele slike (npr. veličina). Klasa *axes* ponajprije služi za postavke višestrukih grafova, ali se neće primjenjivati u ovom primjeru.

Za prikaz točaka na dvije osi (log*P* i log*S*) upotrebljava se dijagram raspršenja (engl. *scatter plot*). Izvršava se metodom *plt.scatter()* ili *ax.scatter()*, a parametri metode su varijable koje se prikazuju (x,y), markeri kao znakovi kojima se prikazuju točke u grafu, u ovom slučaju trokuti (‚^') te natpisi (engl. *label*) i drugi dekoratori.

Za prikaz regresijskog pravca unutar postojećeg dijagrama raspršenja primjenjuje se metoda *ax.plot()*. Podatci potrebni za prikaz regresijskog pravca su domena x-osi (u kodu *x_space* koji je vektor vrijednosti log*P* skupa za učenje) i izračunate odgovarajuće vrijednosti na y-osi iz modelne jednadžbe (*y_fit*) i parametara modela (*a* i *b*). Metodi *plot()* mogu se predati dodatni argumenti, dekoratori za boju (*c*), tip linije (*linestyle*), debljina linije (*lw*) i naziv pravca (*label*). Metode *plt.xlabel(), plt.title()* i *plt.legend()* definiraju ostale

dekoratore, vidljive na slici 6. Korisnike se poziva da sami ispitaju njihova svojstva i mogućnosti te eksperimentiraju s kodom.

```
# In[17]: import matplotlib.pyplot as plt
fig, ax =plt.subplots(figsize=(10,12))
plt.scatter(x=test_set.logp, y=test_set.logs,
marker='^', label='skup za vrednovanje)
x_space=np.array([min(train_set.logp),max(train_
set.logp)])
y_fit=x_space*fitter.coef_ + fitter.intercept_
ax.plot(x_space,y_fit, linestyle='--', c='r',
lw=3, label="regresijski pravac")
plt.ylabel(,logS', fontsize=20, color='red')
plt.xlabel(,logP', fontsize=20, color='green')
plt.title(,logS vs. logP', fontsize=20)
plt.legend(fontsize=20)
```

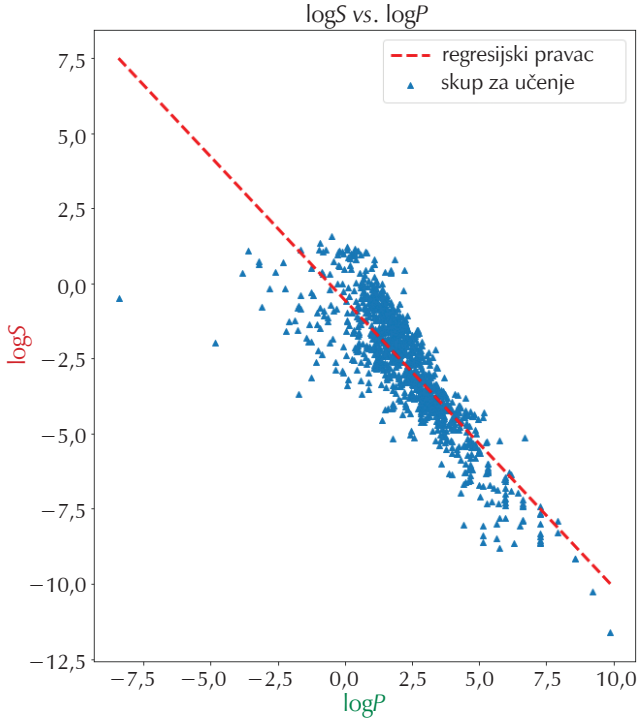


Slika 6 – Prikaz dijagrama raspršenja i regresijskog pravca za skup za vrednovanje

Fig. 6 – Scatter plot and the regression line for the test set

Istim skupom naredbi kao pri ispisivanju grafa skupa za vrednovanje modela može se ispisati skup za učenje, slika 7.

```
# In[19]: fig, ax =plt.subplots(figsize=(10,12))
plt.scatter(x=train_set.logp, y=train_set.logs,
marker='^', label='skup za učenje')
x_space=np.array([min(train_set.logp),max(train_
set.logp)])
y_fit=x_space*fitter.coef_ + fitter.intercept_
plt.plot(x_space,y_fit, linestyle='--', c='r',
lw=3, label="regresijski pravac")
plt. ylabel(,logS', fontsize=20, color='red')
plt. xlabel(,logP', fontsize=20, color='green')
plt. title(,logS vs. logP', fontsize=20)
plt.legend(fontsize=20)
```

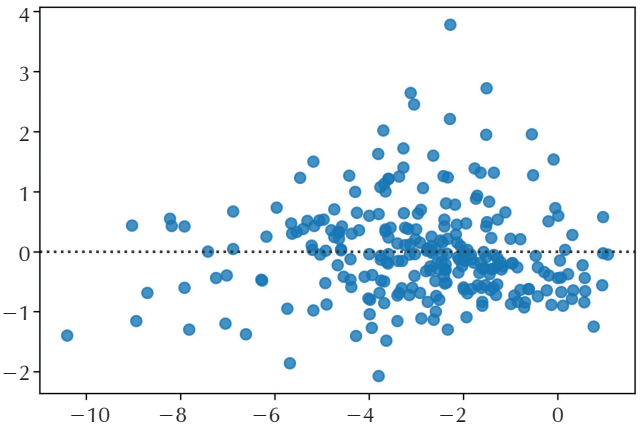


Slika 7 – Prikaz dijagrama raspršenja i regresijskog pravca na skupu za učenje

Fig. 7 – Scatter plot and the regression line for the training set

Dijagnostika modela

Svaki model sadrži određenu pogrešku te vrijednosti koje odstupaju i koje utječu na kvalitetu modela (engl. *outlier*). Već prvim pregledom grafa (slika 7) vidljivo je da postoje vrijednosti koje odstupaju od regresijskog pravca. Za vizualni pregled i dijagram raspršenja oko regresijskog pravca, primjenjuje se biblioteka *seaborn*, koja se uvozi kao kratica *sns* s metodom *residplot* čiji su argumenti modelom izračunata, teorijska (*y_pred*) i eksperimentalna vrijednost za log*S* (*y_true*) (slika 8). Iz vizualne inspekcije čini se da nema grupiranja vrijednosti ili drugih obrazaca u ostatnim odstupanjima oko regresijskog pravca što upućuje na moguću



Slika 8 – Grafički prikaz ostatnih odstupanja modela od regresijskog pravca na skupu za vrednovanje

Fig. 8 – Residual plot for predicted data from the regression line

homoskedastičnost,[24] što je jedna od pretpostavki linearne regresije.

```
# In[18]: import seaborn as sns
sns.residplot(y_true,y_pred)
from sklearn.metrics import mean_squared_error as
MSE
,MSE:',MSE(y_true, y_pred)
Out: (,MSE:', 1.071405731558055)
```

Za pregled osnovnih parametara deskriptivne statistike postoji metoda *describe()* koja se predaje podatkovnom okviru.

```
# In[20]: train_set[[,logs','logp']].describe()
```

Tablica 4 – Ispis polja In[20]
Table 4 – Output result of In[20]

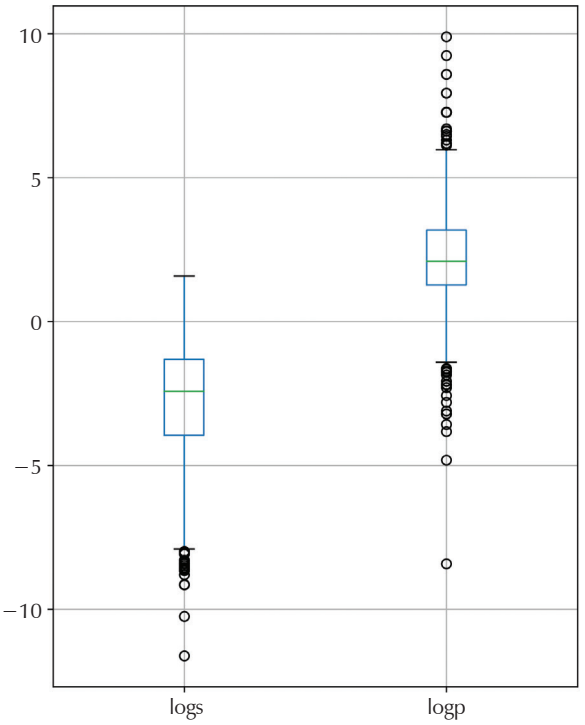
	logs	logp
count	1048,00	1048,00
mean	−2,71	2,26
std	2,03	1,79
min	−11,62	−8,42
25 %	−3,96	1,27
50 %	−2,44	2,09
75 %	−1,32	3,17
max	1,58	9,89

Za provjeru vrijednosti koje značajnije odstupaju u modelu može se primijeniti pravokutni ili B-P dijagram (engl. *boxplot*). Takvo izravno crtanje bez pozivanja biblioteke matplotlib moguće je putem biblioteke Pandas, gdje se metoda *plot()* predaje kao metoda izravno podatkovnom okviru s odgovarajućim parametrima kao što su veličina slike (*figsize*) i mreža (*grid*). U pravokutnom dijagramu vidljivo je postojanje niza vrijednosti koje odstupaju od medijana (zelena linija na slici 9). Posebno uočljive su dvije vrijednosti za log*P* na približno −5 i −8.

Korisniku se preporučuje da o pravokutnom dijagramu više pročita u literaturi te da ga tumači s oprezom.

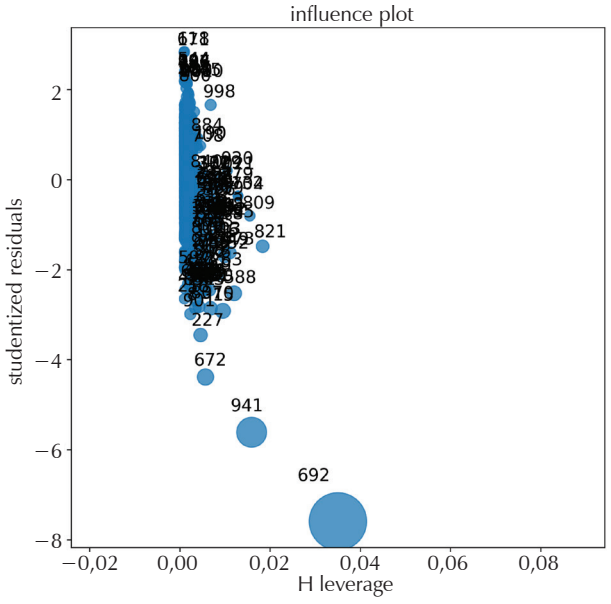
```
# In[21]:
train_set[[,logs','logp']].plot.
box(figsize=(5,10), grid=True)
```

Još jedan vizualni način provjere kvalitete modela je dijagram utjecaja (engl. *leverage plot* ili *influence plot*). Služi isticanju vrijednosti koje imaju velik utjecaj na model; u ovom slučaju regresijski. Za ispisivanje dijagrama utjecaja primjenjuje se biblioteka statsmodels. Slično kao u biblioteci SKLearn uči se model. Međutim ovdje je potrebno dodati konstantu u matrici koja daje prostor za modelni parametar b. To se čini metodom *add_constant()*. Metodom *sm.graphics.influence_plot()* ispisuje se ugođeni (engl. *fitted*) model (slika 10).



Slika 9 – Pravokutni dijagram izabrane prediktorske i ciljne varijable
Fig. 9 – Boxplot for target and predictive variables

```
# In[23]: import statsmodels.api as sm
x_sm=sm.add_constant(xtrain)
model = sm.OLS(ytrain, x_sm)
fitted_model  = model.fit()
fig, ax = plt.subplots(figsize=(12,8))
fig = sm.graphics.influence_plot(fitted_model, alpha
= 0.05, ax=ax, criterion="cooks")
```



Slika 10 – Dijagram utjecaja modela
Fig. 10 – Influence plot for trained model

Iz dijagrama utjecaja vidi se nekoliko vrijednosti koje imaju veći utjecaj u određivanju regresijskog pravca. A to su vrijednosti s indeksom 941 i 692. O kojim vrijednostima se radi, može se saznati iz matrice skupa za učenje. Matrici *xtrain* preda se indeks spoja u uglatoj zagradi *xtrain[]*.

```
# In[24]:xtrain[941]
Out: array([-4.8194])
# In[25]:xtrain[692]
Out: array([-8.4242])
```

Dobivene su dvije vrijednosti za dvije uočljive točke, 941 i 692. Iste dvije vrijednosti bile su uočljive u pravokutnom dijagramu na slici 9.

Podatkovni okviri mogu se filtrirati prema vrijednostima u stupcima, tako da se u uglatoj zagradi dodijeli stupac i granična vrijednost. Ovdje je izabrana gornja granična vrijednost za odstupajuće vrijednosti, log*P* −4,8. Zatim se podatkovnom okviru dodijele stupci od interesa u dvostrukoj uglatoj zagradi i navodnim znakovima. Cilj je ispisati odstupajući log*P* i SMILES strukturu.

```
# In[26]: train_set[train_set.logp<-4.8]
[[,smiles',,logp']]
```

U tablici 5 je ispis dvije molekule koje će se spremiti kao dva objekta za daljnju obradu i označiti kao *outlier1* i *outlier2*. Metodom *loc[]* za indeksiranje podatkovnog okvira izabrat će se reci s odstupajućim vrijednostima iz stupca *train_set.smiles*.

```
# In[27]: outlier1=train_set.smiles.loc[1056]
outlier2=train_set.smiles.loc[105]
```

Pomoću zapisa MOL moguće je izračunati koordinate atoma u prostoru. Nakon prevođenja strukture iz oblika SMILES u MOL metodom *Chem.MolFromSmiles()* i spremanja zapisa MOL u objekt *o1_mol,* metodom *AllChem.Compute2Dcoords()* računaju se koordinate atoma i veza u dvije dimenzije, koje se zatim mogu na ekranu ispisati metodom *print(),* (slika 11).

```
# In[28]: o1_mol=Chem.MolFromSmiles(outlier1)
AllChem.Compute2DCoords(o1_mol)
print(Chem.MolToMolBlock(o1_mol))
```

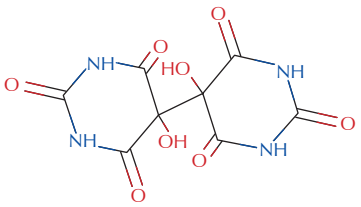
Grafički prikaz strukture molekule ispisuje se pomoću modula *Draw*, a slika se može spremiti ili prikazati na zaslonu. Za grafički prikaz dvaju molekula primjenjuje se metoda *MolToImage()*, a argument te metode je zapis strukture MOL.

```
RDKit          2D

20 21  0  0  0  0  0  0  0  0999 V2000
  -0.1603   -1.1950    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
  -0.6637   -0.3494    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   0.6637    0.3494    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   1.1213   -1.0791    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   2.5872   -1.3970    0.0000 N   0  0  0  0  0  0  0  0  0  0  0  0
   3.5955   -0.2864    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   3.1379    1.1421    0.0000 N   0  0  0  0  0  0  0  0  0  0  0  0
   1.6719    1.4600    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   1.2143    2.8884    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
   5.0614   -0.6043    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
   0.5535   -1.8830    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
   0.1603    1.1950    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
  -1.1213    1.0791    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -2.5872    1.3970    0.0000 N   0  0  0  0  0  0  0  0  0  0  0  0
  -3.5955    0.2864    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -3.1379   -1.1421    0.0000 N   0  0  0  0  0  0  0  0  0  0  0  0
  -1.6719   -1.4600    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -1.2143   -2.8884    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
  -5.0614    0.6043    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
  -0.5535    1.8830    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
  1  2  1  0
  2  3  1  0
  3  4  1  0
  4  5  1  0
  5  6  1  0
  6  7  1  0
  7  8  1  0
  8  9  2  0
  6 10  2  0
  4 11  2  0
  3 12  1  0
  2 13  1  0
 13 14  1  0
 14 15  1  0
 15 16  1  0
 16 17  1  0
 17 18  2  0
 15 19  2  0
 13 20  2  0
 17  2  1  0
  8  3  1  0
M  END
```

Slika 11 – Ispis polja In[28]
Fig. 11 – Output result of In[28]

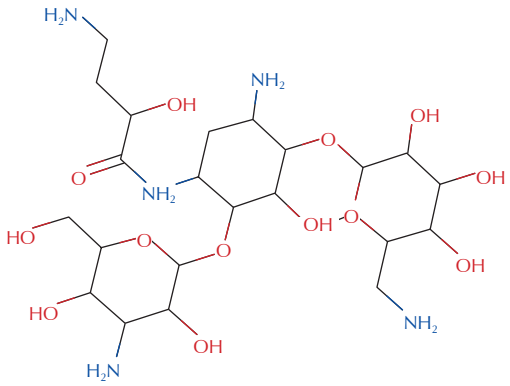
```
# In[29]: Draw.MolToImage(o1_mol)
AllChem.Compute2DCoords(o1_mol)
Draw.MolToImage(o1_mol)
# In[30]: o2_mol=Chem.MolFromSmiles(outlier2)
AllChem.Compute2DCoords(o2_mol)
Draw.MolToImage(o2_mol)
```



Slika 12 – Struktura o1_mol
Fig. 12 – Structure of o1_mol

Tablica 5 – Ispis polja In[26]
Table 5 – Output result of In[26]

	smiles	logP
105	NCCC(C(NC2CC(C(C(C2OC3OC(C(C(C3O)N)O)CO)O)OC1O...	−8,42
1056	OC1(C2(C(NC(NC2=O)=O)=O)O)C(NC(NC1=O)=O)=O	−4,82



Slika 13 – Struktura o2_mol
Fig. 13 – Structure of o2_mol

Na kemičaru je da iskoristi svoje poznavanje domene (spojevi i svojstva koja se ispituju) i protumači mogu li određeni spojevi biti u domeni modela ili ne (engl. *applicability domain*). Iz tog je primjera razvidno npr. da dvije odstupajuće molekule imaju iznimno niske log*P* vrijednosti. Treba uzeti u obzir da se log*P* računa na temelju doprinosa atoma i skupina atoma. Obje molekule iznimno su polarne i doprinos kisika i skupina −OH u strukturama potencijalno precjenjuje iznos log*P*. Korisno je konzultirati literaturu za tumačenja.

Zaključak

Python je vrlo čitak skriptni jezik pogodan za molekulsko modeliranje uz biblioteku RDKit. Uz poznavanje programskih jezika potrebno je dobro isplanirati svoj krajnji cilj i svrhu modeliranja, što se u ovom radu vidjelo na primjeru modeliranja topljivosti. U programiranju, kao i u svakoj vještini, bitno je isprobavati i učiti se na pogreškama. Upravo zato je Jupyter Notebook praktičan jer se kod izvršava liniju po liniju i pogreška se brzo pronađe (engl. *debugging*). U procesu modeliranja velik je fokus na izboru modela koji se trenira, na kvaliteti ulaznih podataka i ispitivanju kvalitete modela, pa se stoga korisniku preporučuje dodatno informiranje o teorijskim osnovama svih navedenih faktora. U idućim nastavcima serijala radova obradit će se modeli s više predikcijskih varijabli i detaljnije ući u problematiku računanja i modeliranja s deskriptorima.

ZAHVALA

Hvala dr. techn. Romanu Kernu (Know-Center) za podršku, Aniti Dogan, Valnei Sindičić i Hani Lovrić za testiranje koda te dr. sc. Boni Lučiću (IRB) za savjete i sugestije.

Literatura
References

1. *I. V. Tetko, O. Engkvist, U. Koch, J.-L. Reymond, H. Chen*, BIGCHEM: Challenges and Opportunities for Big Data Analysis in Chemistry, Mol. Inform. **35** (2016) 615–621, doi: https://doi.org/10.1002/minf.201600073.
2. *I. V Tetko, O. Engkvist, H. Chen*, Does "Big Data" exist in medicinal chemistry, and if so, how can it be harnessed?, Futur. Med. Chem **8** (2016) 1801–1806, doi: https://doi.org/10.4155/fmc-2016-0163.
3. *S. Kim, P. A. Thiessen, E. E. Bolton, J. Chen, G. Fu, A. Gindulyte, L. Han, J. He, S. He, B. A. Shoemaker, J. Wang, B. Yu, J. Zhang, S. H. Bryant*, PubChem Substance and Compound databases, Nucleic Acids Res. **44** (D1) (2016) D1202–D1213, doi: https://doi.org/10.1093/nar/gkv951.
4. URL: https://pubchem.ncbi.nlm.nih.gov/. (1. 3. 2018).
5. *G. Van Rossum, F. L. Drake*, Python Tutorial, History **42** (2010) 1–122, doi: https://doi.org/10.1111/j.1094-348X.2008.00203_7.x.
6. *M. Kaštelan-Macan, M. Petrović* (ur.), Analitika okoliša, HINUS, Fakultet kemijskog inženjerstva i tehnologije, Zagreb, 2013.
7. *B. Petz, V. Kolesarić, D. Ivanec*, Petzova statistika, 7. izd., Naklada Slap, Zagreb, 2012.
8. *J. Votano, M. Parham, L. Hall*, Applied Chemometrics for Scientists, John Wiley & Sons, 2004.
9. *A. Varnek* (Ed.), Tutorials in Chemoinformatics, 1st Ed., Wiley Online Library, 2017., str. 1–462, doi: https://doi.org/10.1002/9781119161110.
10. *K. Roy, S. Kar, R. N. Das*, A Primer in QSAR/QSPR Modeling, Springer, 2015., doi: https://doi.org/10.1007/978-3-319-17281-1.
11. *C. Hansch, J. E. Quinlan, G. L. Lawrence*, Linear free-energy relationship between partition coefficients and the aqueous solubility of organic liquids, J. Org. Chem. **33** (1968) 347–350, doi: https://doi.org/10.1021/jo01265a071.
12. *K. Tanabe, B. Lučić, D. Amić, T. Kurita, M. Kaihara, N. Onodera, T. Suzuki*, Prediction of carcinogenicity for diverse chemicals based on substructure grouping and SVM modeling, Mol. Divers. **14** (2010) 789–802, doi: https://doi.org/10.1007/s11030-010-9232-y.
13. *D. Weininger, A. Weininger, J. L. Weininger*, SMILES. 2. Algorithm for generation of unique SMILES notation, J. Chem. Inf. Model. **29** (1989) 97–101 , doi: https://doi.org/10.1021/ci00062a008.
14. *A. Dalby, J. G. Nourse, W. D. Hounshell, A. K. I. Gushurst, D. L. Grier, B. A. Leland, J. Laufer*, Description of Several Chemical Structure File Formats Used by Computer Programs Developed at Molecular Design Limited, J. Chem. Inf. Comput. Sci. **3213** (1992) 244–255, doi: https://doi.org/10.1021/ci00007a012.
15. *I. V. Tetko, J. Gasteiger, R. Todeschini, A. Mauri, D. Livingstone, P. Ertl, V. A. Palyulin, E. V. Radchenko, N. S. Zefirov, A. S. Makarenko, V. Y. Tanchuk, V. V. Prokopenko*, Virtual Computational Chemistry Laboratory – Design and Description, J. Comput. Aided. Mol. Des. **19** (2005) 453–463, doi: https://

doi.org/10.1007/s10822-005-8694-y.

16. *J. Huuskonen*, Estimation of Aqueous Solubility for a Diverse Set of Organic Compounds Based on Molecular Topology, J. Chem. Inf. Comput. Sci. **40** (2000) 773–777, doi: https://doi.org/10.1021/ci9901338.
17. *W. McKinney*, Data Structures for Statistical Computing in Python, Proc. of the 9th Python in Science Conference **1697900**, 2010., str. 51–56.
18. *T. E. Oliphant*, Python for scientific computing, Comput. Sci. Eng. **9** (2007) 10–20 , doi: https://doi.org/10.1109/MCSE.2007.58.
19. *S. Seabold, J. Perktold*, Statsmodels: Econometric and Statistical Modeling with Python, Proc. of the 9th Python in Science Conference, 2010., str. 57–61.
20. *F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, É. Duchesnay*, Scikit-learn: Machine Learning in Python, J. Mach. Learn. Res. **12** (2012) 2825–2830, doi: https://doi.org/10.1007/s13398-014-0173-7.2.
21. *J. D. Hunter*, Matplotlib: A 2D Graphics Environment, Comput. Sci. Eng. **9** (2007) 90–95 , doi: https://doi.org/10.1109/MCSE.2007.55.
22. *M. Waskom, O. Botvinnik, D. O'Kane, P. Hobson, S. Lukauskas, D. C. Gemperline, T. Augspurger, Y. Halchenko, J. B. Cole, J. Warmenhoven, J. de Ruiter, C. Pye, S. Hoyer, J. Vanderplas, S. Villalba, G. Kunter, E. Quintero, P. Bachant, M. Martin, K. Meyer, A. Miles, Y. Ram, T. Yarkoni, M. L. Williams, C. Evans, C. Fitzgerald, Brian, C. Fonnesbeck, A. Lee, A. Qalieh*, Mwaskom/Seaborn: V0.8.1 (September 2017), doi: https://doi.org/10.5281/zenodo.883859.
23. *G. Landrum*, URL: http://www.rdkit.org/ (1. 3. 2018.).
24. *S. Džalto, I. Gusić*, Simulacija jednostavne linearne regresije, Kem. Ind. **66** (2017) 59–68, doi: https://doi.org/10.15255/KUI.2016.004.

SUMMARY

Molecular Modelling of the Quantitative Structure Activity Relationship in Python (Part I)

Mario Lovrić

Nowadays, the amount of data is increasing considerably, as is their value and knowledge of how to manipulate and extract valuable information. A well-known example of information exploitation is the search of known and design of new chemical compounds based on modelling for the purpose of researching new potential drugs. Therefore, a chemistry student must be well prepared for the current digital era, where it is no longer enough to be skilled in the laboratory, but also unavoidable to be proficient in modelling and analysing data. This handbook covers the basics of molecular modelling and QSAR and the basics of data handling using Python, a free programming language and its molecular modelling library RDKit. Other Python libraries which will be used throughout the manual are: Pandas, for handling and processing all kinds of data; statsmodels, Numpy, Scipy, and SKLearn for mathematical and statistical operations, and linear algebra and Matplotlib and Seaborn for visualisation. The Python programming language is integrated with its mentioned libraries into the Anaconda software. Anaconda enables the user to easily use and manage libraries, as well as use the Jupyter Notebook interface for programming, plotting and data analysis. In this first part of the manual series, the problem of water solubility prediction of a set of organic compounds will be analysed using univariate linear regression. The aim of this series of manuals is to familiarise chemists with the Python programming language, its libraries and practical approaches for solving molecular modelling problems.

Keywords
QSAR, Python, Jupyter Notebook, molecular modeling, RDKit

[a] *Know-Center, Inffeldgasse 13/6, 8010 Graz, Austria*
[b] *NMR Centre, Ruđer Bošković Institute, Bijenička cesta 54, 10 000 Zagreb, Croatia*

Professional paper
Received December 21, 2017
Accepted May 1, 2018