Tutorial XX. Outliers in QSAR

Goal: Illustrate a detection of outlier protocol in a QSAR workflow.

Software: WEKA, ISIDA/ModelAnalyzerR, InstantJChem

Data: A sample of the IUPHAR [1] database dedicated to the serotonin receptor 5-HT_{2B} (IUPHAR_5HT2B.sdf), a set of publications (Publications.zip).

The dataset contains ligands of the human serotonin receptor 5-HT_{2B}, a member of the 5-HT₂ family involved in morphogenesis and anxiety [2]. But this receptor became an antitarget, since it was suspected to have a role in the development of valvular heart disease [3-5]. This discovery led to highly publicized drug withdrawal from the market, first in the Fenfluoramine/Phentermine case [6] and more recently in the benfluorex case [7].

The data for this tutorial were collected from the IUPHAR/BPS [8]. It is composed of 88 compounds together with their affinity to 5-HT_{2B} , their role either as agonist or antagonist, and the PubMed [9] references to the articles describing these information. If several values were reported for a given ligand, the retained value is the median of the collected values. All values are reported for human receptor, if available. Otherwise, the values for the rat receptor are provided (species is always mentioned together with the affinity value). All the articles used to collect this dataset are provided into the archive Publications.zip. The files of the articles are named according to their PubMed ID.

Compound structures and related data fields are in file IUPHAR_5HT2B.sdf. The SD fields affinity, type and pubmed_id contain the pKi value, the agonist or antagonist label and the list of bibliographic references respectively. The SD field CdId contains a unique integer to facilitate reference to a given compound and the ligand field contains the common name of the compound.

The regression problem consists in estimating ligand affinity (expressed as pK_i = negative log of the complex instability constant expressed in mol/l) to 5-HT_{2B}, as a function of the ligand structure.

Theoretical background.

The tutorial uses Gaussian Processes to build models [10, 11]. The general idea is to describe the dataset as a multivariate Gaussian probability distribution. The vector of target affinities, *Y*, is understood as a sample of a multivariate normal distribution resulting from the molecular descriptors, and a noise σ . The molecular descriptors are contributing to the definition of the distribution by estimating the covariance of the distribution. Technically, the covariance is identified to a kernel function of the training set instances: $\Sigma_{ij} = k(x_i, x_j)$.

The method depends of several choices: the level of noise and the kernel function that is also parameterized. The noise level is therefore related to the confidence in the values of affinities, while the kernel shall be chosen based on domain knowledge about the molecular descriptors in use.

Although it is not needed in the tutorial, the Gaussian Processes are often embedded into a global Bayesian reasoning. In this case, the parameters (noise, kernel parameters) are themselves sampled to maximize the marginal likelihood using dedicated algorithms such as Markov Chain Monte Carlo [12]. These approaches, combined to the numerical complexity of the linear algebra needed to solve the Gaussian Processes problems, can lead to rather costly calculations for datasets of thousands of instances.

Also, the tutorial adds some focus to the detection of outliers. The strategy proposed here consists into developing the best "non-over-fitted" model on the dataset. To avoid over-fitting, a rigorous external validation is recommended. Then, if the model cannot fit some

instances, they are potential outliers. Once the outliers are removed, model building and outlier analysis is iterated. The process is ended when no more outliers are found. This is a sequential inward approach [13-15]. An outward approach would consist in adding non-outlier data to the dataset in a sequential procedure. It is possible to search for ensembles of outliers, or to do the stepping by focusing on the single most outstanding outlier at every stage.

Irrespective of the algorithm, there are several important aspects to be kept in mind during outlier analysis.

- First, an outlier is not the result of a numerical procedure: an algorithm fails to fit a data point, but this may be called an outlier only if the anomalous value can be discarded for a reason: a potential measure problem, an unexpected event during data acquisition, sabotage... Without a cause, an anomalous point is not an outlier and cannot be discarded – on the contrary, it may simply be an indication of the failure of the modeling strategy.
- Second, a data point can be anomalous only relative to some *a priori* knowledge. In this tutorial, we propose for instance to use the Grubbs algorithm [16] which is very representative: an anomalous point is an extreme value compared to a normally distributed sample.

In this tutorial, we shall focus on the distribution of the residuals of a regression model. They are supposed to follow a Normal probability distribution. The *n* residues of values r_i are searched for the largest value r_n and the smallest value r_1 . It turns out that the quantity $G_n = \frac{r_n - \langle r \rangle}{s}$ and $G_1 = \frac{\langle r \rangle - r_1}{s}$, where *s* and $\langle r \rangle$ are the standard deviation and mean of the residuals, follow a studentized extreme deviation statistics [17]. The Grubbs test, consists

in comparing these decision variables, to the critical value $G_c = \frac{n-1}{\sqrt{n}} \sqrt{\frac{t^2}{n-2+t^2}}$, where *t* is the

 $\alpha/2n$ fractile of a student distribution of n-2 degrees of freedom, with a risk α . If a decision variable is larger than the critical value, the extreme value shall be considered as anomalous. By convention, if for a data value the test is positive up to a risk α of 1%, it is considered anomalous whereas with a risk of 5% it is only suspicious.

This setup is consistent as long as the hypothesis that the residuals are distributed according to the centered normal law is legitimate. This is not the always the case, as for instance for the residuals of a fit from of an ϵ -SVM model.

Step by step instructions

Exercise 1:

In this first part of the tutorial, the IUPHAR_5HT2B.sdf file is loaded into *InstantJChem*.

Instructions	Comments
 Start InstantJChem. Click on the main File->New Project or alternatively click on the icon indicated on Figure 1. 	It is essential to work with a database, in order to include or exclude easily some instances and to easily check all available information on a given instance.
•	Start the software <i>InstantJChem</i> (Figure 1). Start a new project to open the New Project wizard.
	$\checkmark \simeq \begin{bmatrix} a \\ z \end{bmatrix} \begin{bmatrix} a \\ a \end{bmatrix} \begin{bmatrix} a \\ z \end{bmatrix} \begin{bmatrix} a \\ z \end{bmatrix} \begin{bmatrix} a \\ z \end{bmatrix}$

Forms				Quick Actions
Form	Data Tree	Schema	Project	■ Connect to Database ▲ Import File
				Templates

Figure 1: The InstantJChem software starts with a Dashboard summarizing user's projects. To start a new project, click on the framed icon 1 in red in this picture. To import a dataset click the framed icon 2.

•	Select the IJC Poject (with local database) option, and then click on Next (Figure 2).	These operations create a new directory named <i>5-HT2B</i> that will
•	In the next step, name your project as 5-HT2B in the field Project Name .	contain the database. Yet, the database contains no data so far.
•	Location (Figure 3). A good choice can be the folder containing the tutorial files. The field Project folder is adequately set automatically.	The <i>5-HT2B</i> directory will contain everything that will be needed to manage a local database.
•	Click the Finish button.	

Steps	Choose Project	
Steps 1. Choose Project 2	Choose Project Categories:	Projects: UC Project (empty) UC Project (with local database) UC Project (local database with demo data
Instant JChem	Description: New Instant JChem project with local o	database
	Help < Back	Next > Finish Cancel

Figure 2: *Project creation wizard. Chose the second option that creates a project and setup a local database in one step.*

Steps	IJC Project Name		
 Choose Project IJC Project Name 	Project Name:	5-HT2B	
	Project Location:	/Users/marcou/Desktop/Tutorial	Browse
	Project Folder:	/Users/marcou/Desktop/Tutorial/5-HT2B	
		Close Already Opened Projects	
Instant JChem			
\bigotimes			
		Help < Back Next > Finish	Cancel

Figure 3 : Second step of the project creation wizard. Once a project name and location is set, the project folder is set automatically.

 Click on the main <i>File->Import File</i> or alternatively click on the is icon indicated (see Figure 1). Choose the file IUPHAR_5HT2B.sdf then click on the <i>Next</i> button (Figure 4). Then check the frame labeled <i>Field in file</i>. 	The wizard guides the user thru the process of importing the chemical structures with their additional information into the 5- <i>HT2B</i> database.
 Click on the item <i>CdId</i> and then on the button <i>Add</i>. Set the <i>Display name</i> of this field as <i>Original CdId</i> (Figure 5). Click the button <i>Next</i>. At the end of the loading process, click the button <i>Finish</i>. 	During the process, the original <i>Cdld</i> field from the file is not automatically imported because it seems to be a duplicate of the default <i>Cdld</i> field generated by <i>InstantJChem</i> during import.
	It is therefore necessary to force the import and to rename it as <i>Original CdId</i> . The name of the database field is automatically updated from the display name.

Ste	eps	File and new ta	able details		
1. 2. 3. 4.	Select schema File and new table details Field details Monitor import	Database: File to import:	Iocaldb /Users/marcou/Desktop/Tutorial/IUPH	AR_5HT2B.sdf	-5
		File type:	Structure file – SDF	\$	
		Table details:	III New structure entity (using JChemBa	se table) ᅌ	•••
		Summary:	IUPHAR_5HT2B [APP.IUPHAR_5HT2B] Typ	be: Molecules	
		10 fields found	:		
		Structure [Text,S CdId [Text,List (tructure,List (Text)] Text).Boolean.Decimal.Integer]	Records read:	88
		Cdid [Text,List (Text),Boolean,Decimal,Integer] Mol Weight [Text,List (Text),Decimal] Formula [Text,List (Text)] target_species [Text,List (Text)] ligand [Text,List (Text)] type [Text,List (Text)]			100 🗘
	Instant IChem	affinity_units [Text,List (Text)] affinity [Text,List (Text)]			
	8	pubmed_id [Tex	t,List (Text)]		
			Help < Back Next	> Finish	Cancel

Figure 4: File and table details of the import wizard.

Select schema			
File and new table	Fields in file	1	Fields in database
Field details Monitor import	Structure CdId Mol Weight Formula target_species ligand type affinity_units affinity pubmed_id	Add > < Merge > < Map > < Remove Move up	123₀ Cdld → O₀ Structure [Structure] 1,23₀ Mol Weight A₀ Formula + A target_species + A ligand + A type + A affinity_units +1,23 affinity
	New field type: 123 Display name: Original	Integer Field	+ 102 Original Cdld [Cdld]
Instant JChem	Required: FALSE		
8	DB Column Name: Orig	ginal_CdId	

Figure 5: Field details interface. Be sure to import the Cdld from the input file, after renaming it "Original Cdld".

•	In the grid view of the 5-HT2B database,	The automatically generated fields are disturbing for the management of
	corner and select the Open Column Manager option (Figure 6).	the database in the present case. Therefore, it is advised to hide them.
•	Select the items <i>Cdld</i> , <i>MolWeight</i> and <i>Formula</i> from the right hand frame then click the Remove button (Figure 7).	The database is now ready to use (Figure 8).



Figure 6: The widget configuration menu of the grid.

Available fields:		Selected fields:
a a a Search 123o Cdld 1,23o Mol Weight Ao Formula	Add -> Add All -> <- Remove <- Remove All	Structure A target_species A ligand A type A affinity_units 1,23 affinity A pubmed_id 123 Original CdId
	Move Up Move Down	
		Cancel OK

Figure 7: The column manager interface. Remove confusing columns: CdId, MolWeight and Formula.



Figure 8: State of the InstantJChem interface, once the 5-HT2B data has been loaded.

Exercise 2:

In the second part, the IUPHAR-5HT2B.sdf file is used to generate an external cross-validation framework.

• Start the <i>ExtCV</i> software (Figure 9).	The tool will create new folders called
• Use the interface (1, in the figure) to select	CVIteriFoldj, where i and j refer to a
the file IUPHAR-5HT2B.sdf. Choose a cross-validation experiment in the menu (2) indicated on the figure. Edit the boxes (3) to set the number of folds to $N=4$ and the number of repetition of the experiment to $k=1$	particular iteration and fold respectively. Inside each of these folders, are located the corresponding training and test sets. They are the files train.sdf and test.sdf.
• Click the button <i>Run!</i> .	During the procedure, the training set and test sets are isolated from each other. The calculation of the molecular descriptors being separated train and test sets, it is not possible to transfer information from the test set to the train inadvertently. Beside, it forces the modeler to produce a complete chain to apply his models to the test data.



Figure 9: Interface of the ExtCV software. The edit field and the button to its right (1) is used to select an SDF file, the combobox interface (2) is used to choose a validation protocole and the numeric text edit (3) are used to set the parameters of the validation (*N* iterations and *k* repetitions).

•	Start the <i>xFragmentor</i> software (Figure 10).	During this step, all the files train.sdf
•	Click the button Add SDF to add all the	are analyzed and ISIDA substructural
	train.sdf files located in the directories CVIter1Fold*.	molecular fragment (SMF) descriptors are computed
•	Add also the file IUPHAR_5HT2B.sdf.	

• • • • • • • • • • • • • • • • • • • •	Click the button <i>Add fragment</i> to add to the fragmentation algorithm, a simple atom count. Choose the <i>IIR</i> fragment topology from the <i>Select a topology</i> combo box. Set the <i>Min length</i> to 2 and the <i>Max length</i> to 4. In the combo box <i>Select a coloration scheme for atoms</i> , select the option <i>A</i> . In the combo box <i>Select a coloration scheme for bonds</i> , select the option <i>B</i> . Tick the box <i>Use formal charge</i> . Click the button <i>Add fragment</i> . Type the word "affinity" into the <i>SDF field of interest</i> text edit. Click the button <i>Run!</i> .	 More specifically, these descriptors are counting the number of atoms of each type (fragments of type <i>E</i>) and the number of atom centered fragments composed of paths of uniform length, the length varying between 2 and 4 atoms. The atom types and bond types are recorded into the fragment and the formal charges, if any, are annotated (fragments of type <i>IIAB_R(2-4)_FC</i>). In the folder containing a target SDF, the software creates several files: An XML file recording the state of the software when it generated the ISIDA SMF descriptors. A HDR file, containing the list of molecular fragment hashed from the chemical structures. An SVM file that is a sparse storage of the molecular descriptor values. An ARFF file that condense in one file the information of the HDR and SVM file, and can be used in the Weka data mining activation.
		During the procedure, the values found in the field named <i>affinity</i> into the SDF file are reported into the molecular descriptor files. It the value is missing it is replaced by a question mark ("?"). This field is considered as the target property of the QSAR.
•	Click the button <i>Save XML</i> . Save configuration as train_E_IIAB_R(2- 4)_FC.xml.	Save the current configuration of the software. It can be reloaded, so that exactly the same fragmentation can
•	Recursively select each line in the frame List of SDF files to process and click the	During this step, the ISIDA SMF descriptors are computed on the

button Remove SDF. test.sdf files. using the same Click the Add SDF button and select each ٠ algorithm as the train.sdf files, and test.sdf file in the folders CVIter1Fold*. the same molecular fragment Tick the box Use predefined fragments. dictionary (numbering). Tick the box Use only those fragments. • Type "train E IIAB R(2-4) FC" into the text If new fragments are discovered in ٠ edit of the frame Use predefined fragments. the process (and subsequently You can check that there are no misspelling ٠ ignored), a "!" will appear in the log of by clicking on the button Check. the interface, near the index of the Click the button Run!. • chemical structure. Therefore, the test.* files generated by the software are ready to be used as external test sets for models build and optimized in the train.* files only. List of SDF files to process Select a topology: IIR (Atom centered, homogeneous path s 💙 Add Fragment 🛛 E ktop/Tutorial/CVIter1Fold1/train.se Add SDF IIAB_R(2-4)_FC ktop/Tutorial/CVIter1Fold2/train.se ktop/Tutorial/CVIter1Fold3/train.s Min length Max length Delete Fragment ktop/Tutorial/CVIter1Fold4/train.se 2 🗘 4 🗘 ktop/Tutorial/IUPHAR_5HT2B.sdf SDF field of interest Select a coloration scheme for atoms: affinity| A Atom type Read XML Select a coloration scheme for bonds Save XML B Bond type Use predifined fragments Atom Pairs Base name of header files... Check Do All Ways Use only those fragments Use Formal Charges 3 0 Do not use marked atom Get atom contribution to fragment ***** ISIDA/xFragmento Université de Strasbourg, 2016 Quit Reset Run!

Figure 10: Interface of the xFragmentor software. The zone (1) is dedicated to define the list of SD files to compute molecular descriptors. The zone (2) is used to setup and combine fragmentation algorithms. The zone (3) is used to generate molecular descriptors using predefined fragments.

Exercise 3:

In the third part, SVM models are optimized on the training sets. Then, the model is applied to the corresponding test set.

•	Open the Weka software.	The Weka Experimenter
•	Choose the Experimenter tool (Figure 11).	software automatizes the
•	Click the button New.	process of running a set of
•	Type "ExtCVtrain.arff" into the <i>Results destination</i> text edit.	machine learning methods,
•	Into the Experiment Type frame: o Set the menu to Cross-validation	

• Set the Number of folds to 4

• Select the *Regression* radio button

- In Iteration Control set the Number of Iteration to 4.
- In the *Datasets* frame, click the button *Add* to add each of the files train_E_IIAB_R(2-4)_FC.arff located in the folders CVIter1Fold*.
- In the *Algorithms* folder, click the button *Add new* and accept the default method (*ZeroR*)
- Click the button Add new again.
- In the pop up window, click the button *Choose*.
- From the Weka hierarchy of machine learning methods, chose *GaussianProcesses* (Figure 13).
- Set the filter type to No normalization/standardization.
- Set the *noise* value to 1.
- Click the button OK.
- Repeat the process, add other Gaussian Processes, differing only by the *noise* value. Set the noise to integer values from 1 to 10.
- Click the button *Save…* and create an experiment file called ExtCVtrain.exp.
- Into the tab *Run* click the button *Start*.

on a series of training sets.

It is used on the training sets in order to estimate the optimal value of the noise to use later with the *GaussianProcesses* method in the current situation.

The methods needs to decide the shape of a covariance function, that is identified to a kernel. Using ISIDA molecular descriptors, a simple but efficient choice is a linear kernel (the default value for the method) and no transformation of the descriptors' values.

The configuration of the experiment is saved in a file named ExtCVtrain.exp. So, this setup can be reused at any time.

Setup Run Analyse	
experiment Configuration Mode Simple	
<u>O</u> pen	Save <u>N</u> ew
Results Destination	· · · · · · · · · · · · · · · · · · ·
ARFF file Filename: //Users/marcou/Desktop/Tutorial/ExtCVtrain.arff	Browse
xperiment Type	Iteration Control
Cross-validation	Number of repetitions: 4
Number of folds: 4	📿 💿 Data sets first
Classification Regression	Algorithms first
atasets	Algorithms
Add new Edit selected Delete selected User relative paths ////////////////////////////////////	Add new Edit selected Delete selected GaussianProcesses - L 10.0 - N 0 - K' weka.classifiers.functions.supportVector.PolyKernel - E 1.0 - C 250007' - S 1 GaussianProcesses - L 8.0 - N 0 - K' weka.classifiers.functions.supportVector.PolyKernel - E 1.0 - C 250007' - S 1 GaussianProcesses - L 9.0 - N 0 - K' weka.classifiers.functions.supportVector.PolyKernel - E 1.0 - C 250007' - S 1 GaussianProcesses - L 9.0 - N 0 - K' weka.classifiers.functions.supportVector.PolyKernel - E 1.0 - C 250007' - S 1 GaussianProcesses - L 0.0 - N 0 - K' weka.classifiers.functions.supportVector.PolyKernel - E 1.0 - C 250007' - S 1 GaussianProcesses - L 0.0 - N 0 - K' weka.classifiers.functions.supportVector.PolyKernel - E 1.0 - C 250007' - S 1 GaussianProcesses - L 0.0 - N 0 - K' weka.classifiers.functions.supportVector.PolyKernel - E 1.0 - C 250007' - S 1 GaussianProcesses - L 0.0 - N 0 - K' weka.classifiers.functions.supportVector.PolyKernel - E 1.0 - C 250007' - S 1 GaussianProcesses - L 3.0 - N 0 - K' weka.classifiers.functions.supportVector.PolyKernel - E 1.0 - C 250007' - S 1 GaussianProcesses - L 3.0 - N 0 - K' weka.classifiers.functions.supportVector.PolyKernel - E 1.0 - C 250007' - S 1 GaussianProcesses - L 3.0 - N 0 - K' weka.classifiers.functions.supportVector.PolyKernel - E 1.0 - C 250007' - S 1 GaussianProcesses - L 3.0 - N 0 - K' weka.classifiers.functions.supportVector.PolyKernel - E 1.0 - C 250007' - S 1
4. Up Down	5

Figure 11: Interface of Weka Experimenter. The top frame (1) is used to prepare the experiment: either prepare a new one, load or save an experiment. The frame below (2) is dedicated to saving the experiment report. The next frame (3) is the design of the experimental protocol: how model performances are evaluated and how many times the experiment is repeated. The bottom left frame (4) stores the datasets that are being processed and the bottom right frame (5) lists the machine learning methods to be experimented.



Figure 12: Location of the Gaussian Processes into the hierarchy of Weka machine learning methods.

weka.classifie	rs.functions.	GaussianProcesses							
About	About								
Impleme hyperpar	nts Gaussian ameter-tunir	n processes for regression without More ng. Capabilities							
	batchSize	100							
	debug	False							
doNotCheck	Capabilities	False							
	filterType	No normalization/standardization							
	kernel	Choose PolyKernel -E 1.0 -C 250007							
	noise	4.0							
numDeo	cimalPlaces	2							
	seed	1							
Open		Save OK Cancel							

Figure 13: Weka interface to configure a Gaussian Process method.

• Click the Analyse tab to display t	The analysis interface displays the 640
analysis interface of the experime	nt results into a table. By default, lines are
(Erreur ! Source du renvoi introuvable.).	datasets and columns are methods.
Click the button <i>Experiment</i> .	
Set the Comparison field	to The default method, ZeroR, should be
Relative_absolute_error.	the first one. This method consists in
Click the button <i>Perform test</i> .	using as a regression method, the
	average of the property. In other words,
	using this model, all compounds are
	predicted the same value which is the
	predicted the same value which is the
	average pKI value on 5-HI _{2B} as
	observed on the training set.
	This analysis tool does pair comparison
	hot was the first method (here ZereD)
	between the first method (here, ZeroR)

and other methods (the Gaussian Processes). The symbols (V/ /*) are used to annotate those results that have equivalent or lower value, larger, respectively, compared to a base line line classfier. Currently the base classifiere is the model ZeroR that consist into assigning the average pKi value to any instance. The bottom line summarizes the number of datasets for which the evaluated method got higher, equivalent or lower values compared to the reference. In the present case (Figure 15), the relative mean absolute error is used to check the models. Two conclusions emerge. First, the lowest errors are usually obtained for noise values about 3 or 4. However. these results are not annotated with an asterisk (*), meaning that they are no convicingly better than the base line model (ZeroR). These discrepancies are partly due to the small size of the dataset, to the low number of iterations in the experiment setup and to the presence of outliers. In the following the value of 4 is assumed to be the optimal noise level.

Setup Run Analyse																			
Source	-																		
Got 640 results															E	Eile	<u>D</u> atabase	Exper	iment
Actions																			
Perform test	Save output Open Explorer.																		
Configure test		T	est output																
Testing with	Paired T-Tester (corrected)	;[Tester: Analysing: Datasets:	weka.experimen Relative_absol 4	t.Paire ute_er	edCorrected ror	TTeste	er -G 4,5	6 —D	1 -R	2 -S 0.05	-result-m	atrix "we	ka.experi	ment.f	ResultMat	rixPlainT	ext —mean-	prec 2
Comparison field	Relative_absolute_error		Confidence: Sorted by: Date:	10 0.05 (two tail - 10/06/16 01:28	ed)														
Sorting (asc.) by	<default></default>		Dataset				(1)		(2)		(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
Test <u>b</u> ase	Select		/Users/marc /Users/marc /Users/marc	ou/Desktop/Tuto ou/Desktop/Tuto ou/Desktop/Tuto	rial/C\ rial/C\ rial/C\	/Iter1Fold1 /Iter1Fold2 /Iter1Fold3	(16) (16) (16)	86.58 89.82 90.35	85	5.92 9.31 9.75	85.26 88.77 89.16	84.60 88.24 88.62	83.99 87.90 88.11	83.5 87.6 87.6	3 0 5	83.22 87.33 87.28	83.02 87.15 87.50	83.41 87.16 88.88	85.0 87.5 91.0
Displayed Columns	Select		/Users/marc	ou/Desktop/Tuto	rial/C\	/Iter1Fold4	(16)	85.02	84	4.17 *	83.41	82.64	∗ 81.89	* 81.2	8 *	80.96 *	81.17	81.90	83.2
Show std. deviations								(v/ /*)	(6	9/3/1)	(0/3/1	(0/3/1) (0/3/	1) (0/3	/1)	(0/3/1)	(0/4/0)	(0/4/0)	(0/4
<u>O</u> utput Format	Select		Key:																
Result list			<pre>(1) functio (2) functio (3) functio</pre>	ns.GaussianProc ns.GaussianProc ns.GaussianProc	esses esses esses	'-L 10.0 -N '-L 9.0 -N '-L 8.0 -N	2K 2K \ 2K \	\"functio "functio" "functio"	ons.su ns.sup ns.sup	pport portV portV	Vector.Po ector.Pol ector.Pol	yKernel – Kernel –E Kernel –E	E 1.0 -C 1.0 -C 2 1.0 -C 2	250007\" 50007\" - 50007\" -	-S 1' S 1' · S 1' ·	-8620066 -86200669 -86200669	949967678 9499676785 9499676785	545 45 45	
01:23:59 - Root_mea	an_squared_error - functions.Gaussia		<pre>(4) functio (5) functio (6) functio</pre>	ns.GaussianProc ns.GaussianProc	esses esses esses	-L 7.0 -N -L 6.0 -N -L 5.0 -N	2 - K \ 2 - K \ 2 - K \	"function "function"	is.sup is.sup is.sup	oportV oportV	ector.Pol	Kernel -E Kernel -E	1.0 -C 2 1.0 -C 2 1.0 -C 2	50007\" - 50007\" - 50007\" -	5 1' - 5 1' - 5 1' -	-86200669 -86200669 -86200669	499676785 499676785 499676785	45 45 45	
01:24:55 - Root_mea	01:24:55 - Root mean squared error - /Users/marcol		(7) functio (8) functio	ns.GaussianProc ns.GaussianProc	esses esses	'-L 4.0 -N '-L 3.0 -N	2 – K N 2 – K N	"function" function	ns.sup ns.sup	oportV oportV	ector.Pol	Kernel −E Kernel −E	1.0 -C 2 1.0 -C 2	50007\" - 50007\" -	S 1' - S 1' -	-86200669 -86200669	499676785 499676785	45 45	
01:25:42 - Available	01:25:42 - Available resultsets		(9) functio	ns.GaussianProc	esses	-L 2.0 -N	2 -K \	"function	ns.sup	portV	ector.Pol	Kernel -E	1.0 -C 2	50007\" -	S 1' -	-86200669	499676785	45	
01:25:45 - Root_mean_squared_error - functions.Gaus			(10) TUNCTI	uns.uaussianPro	Lesses	-L 1.0 -N	2 -K	\ Tunction	ль.su	hholt	vector.Po	yvernet -	- 1.0 -0	220007/~	-5 1.	-0020000	94990/0/8	343	
01:25:35 - Relative_absolute_error - functions.Gaussia																			
01:26:54 - Relative_absolute_error - functions.Gaussia																			
01:28:14 - Relative_a	absolute_error - functions.Gaussia 🔻																		
	7.		•				_		_										7.0
	1																		

Figure 14: Analysis interface of Weka Experimenter

Dataset	(1) r	ules.Zer	I	(2) funct	(3) funct	(4) funct	(5) funct	(6) funct	(7) funct	(8) funct	(9) funct	(10) func	(11) func
CVIter1Fold1_affinity CVIter1Fold2_affinity CVIter1Fold3_affinity CVIter1Fold3_affinity CVIter1Fold4_affinity	(16) (16) (16) (16)	100.00 100.00 100.00 100.00	 	86.58 * 89.82 * 90.35 85.02 *	85.92 * 89.31 * 89.75 84.17 *	85.26 * 88.77 89.16 83.41 *	84.60 * 88.24 88.62 82.64 *	83.99 * 87.90 88.11 81.89 *	83.53 * 87.60 87.65 81.28 *	83.22 * 87.33 87.28 80.96 *	83.02 * 87.15 87.50 81.17 *	83.41 * 87.16 88.88 81.90 *	85.01 87.55 91.09 83.23 *
		(v/ /*)	I	(0/1/3)	(0/1/3)	(0/2/2)	(0/2/2)	(0/2/2)	(0/2/2)	(0/2/2)	(0/2/2)	(0/2/2)	(0/3/1)
<pre>Key: (1) rules.ZeroR '' 48055541465867954 (2) functions.GaussianProcesses '-L 10.0 -N 2 -K \"functions.supportVector.PolyKernel -E 1.0 -C 250007\" -S 1' -8620066949967678545 (3) functions.GaussianProcesses '-L 9.0 -N 2 -K \"functions.supportVector.PolyKernel -E 1.0 -C 250007\" -S 1' -8620066949967678545 (4) functions.GaussianProcesses '-L 8.0 -N 2 -K \"functions.supportVector.PolyKernel -E 1.0 -C 250007\" -S 1' -8620066949967678545 (5) functions.GaussianProcesses '-L 8.0 -N 2 -K \"functions.supportVector.PolyKernel -E 1.0 -C 250007\" -S 1' -8620066949967678545 (6) functions.GaussianProcesses '-L 6.0 -N 2 -K \"functions.supportVector.PolyKernel -E 1.0 -C 250007\" -S 1' -8620066949967678545 (7) functions.GaussianProcesses '-L 5.0 -N 2 -K \"functions.supportVector.PolyKernel -E 1.0 -C 250007\" -S 1' -8620066949967678545 (8) functions.GaussianProcesses '-L 5.0 -N 2 -K \"functions.supportVector.PolyKernel -E 1.0 -C 250007\" -S 1' -8620066949967678545 (9) functions.GaussianProcesses '-L 5.0 -N 2 -K \"functions.supportVector.PolyKernel -E 1.0 -C 250007\" -S 1' -8620066949967678545 (9) functions.GaussianProcesses '-L 3.0 -N 2 -K \"functions.supportVector.PolyKernel -E 1.0 -C 250007\" -S 1' -8620066949967678545 (10) functions.GaussianProcesses '-L 3.0 -N 2 -K \"functions.supportVector.PolyKernel -E 1.0 -C 250007\" -S 1' -8620066949967678545 (11) functions.GaussianProcesses '-L 3.0 -N 2 -K \"functions.supportVector.PolyKernel -E 1.0 -C 250007\" -S 1' -8620066949967678545 (12) functions.GaussianProcesses '-L 3.0 -N 2 -K \"functions.supportVector.PolyKernel -E 1.0 -C 250007\" -S 1' -8620066949967678545 (13) functions.GaussianProcesses '-L 3.0 -N 2 -K \"functions.supportVector.PolyKernel -E 1.0 -C 250007\" -S 1' -8620066949967678545 (14) functions.GaussianProcesses '-L 3.0 -N 2 -K \"functions.supportVector.PolyKernel -E 1.0 -C 250007\" -S 1' -8620066949967678545 (15) functions.GaussianProcesses '-L 3.0 -N 2 -K \"functions.supportVector.PolyKernel -E 1.0 -C 250007\" -S 1' -8620066949967678545 (16) functions.GaussianProc</pre>													

Figure 15: Example of results of Gaussian Processes with noise between 1 and 10, compared to a Zero rule model. The quantities tested are relative mean absolute errors.

•	Create a new directory named testsets. Copy each test_IIAB_R(2-4)_FC.arff file into the testsets directory and rename it so that it is prefixed by its original folder	To automatize the evaluation of the models on their corresponding external test set, some reorganization is needed.
	 For instance the file CVIter1Fold1/ test_E_IIAB_R(2-4)_FC.arff is copied as the file testsets/CVIter1Fold1_affinity_test .arff 	First, Weka expects that all test sets are located in the same directory. Second, the name of the test set file is derived from the name of the relation (the keyword @RELATION in an ARFF file).
•	Edit each ARFF training dataset file. Change the name of the relation so that it starts by the name of the folder in which it is located followed by _affinity.	Fragmentor, that follows a different logic. Therefore, some editing is needed.

be @RELATION "CVIter1Fold1_affinity".	
 In the Weka Experimenter software, click on the Setup tab. In the area 1 in Figure 11, set the Experiment Configuration Mode to Advanced. The interface changes and should look like in Figure 16. In the area 2, click on the button New. 	In this setup, the models are prepared on the training set files. For each training set, the performances of the model is evaluated on the correspondong test set.
 In the area 3, click on the default method InstancesResultListener to open the configuration interface, and change the name of the outputFile to ExtCVtest.arff inside the working directory. 	individual evaluation procedures on al datasets. For instance, in the frame of a cross-validation, one run would be one fold of cross-validation for each dataset
 Set the value of <i>Runs</i> values from 1 to 1. Click on the <i>testsetDir</i> text edit to set its value to the directory testsets containing the test datasets. Click the button <i>Choose</i> of the <i>Result</i>. 	In the present case, repeating the training test operations without any change on the training set nor the tes set is useless : the results would be
Generator frame and select ExplicitTestsetResutProducer.	exactly the same for each run. This is why, it is set to 1.
• Click the word <i>ExplicitTestsetResutProducer</i> to open the configuration interface (Figure 17).	Finally, it is a good idea to save the experimental setup.
Click the button Choose of the splitEvaluator frame and choose RegressionSplitEvaluator.	
• Click the <i>RegressionSplitEvaluator</i> to open a new configuration interface (Figure 18).	
 Click the button <i>Choose</i> of the <i>classifier</i> frame. Select the <i>GaussianProcesses</i> item 	
 (Figure 12). Click on the word <i>GaussianProcesses</i> to open the related configuration window. 	
 Configure the method as in Figure 13. Validate all the configuration interfaces by clicking on the buttons <i>OK</i>. 	
• Click the button <i>Save</i> and save your setup as ExtCVtest.exp.	

Setup Run Analyse			
xperiment Configuration Mode	Advanced 💌		
<u>O</u> pe	n	<u>S</u> ave	New
Destination			
Choose InstancesResultL	istener –0 ExpCVtest.arff	2	
lesult generator			
Choose ExplicitTestsetRe	suitProducer –R –O splitEvalutorOut.zip –dir /Users/mar	cou/Desktop/Tutorial/testsets –suffix	test.arff W weka.experiment.RegressionSplitEvaluator
		3	
luns	Distribute experiment	Generator properties	
From: 1 To: 1	Hosts	Disabled	Select property
	By data set O By run O By property		
teration control		_	
Data sets first	 Custom generator first 		
atasets		_	
Add new	Edit selected Delete selected		
Use relative paths	Euli sciecteu Delete sciecteu		Can't adit
/Users/marcou/Deskton/Tuto	rial/CV/Iter1Eold1/Itrain E IIAR R(2-4) EC arff		canteur
/Users/marcou/Desktop/Tuto	rial/CVIter1Fold2/train_E_IIAB_R(2-4)_FC.arff		
/Users/marcou/Desktop/Tuto	rial/CVIter1Fold3/train_E_IIAB_R(2-4)_FC.arff		
/Users/marcou/Desktop/Tuto	rial/CVIter1Fold4/train_E_IIAB_R(2-4)_FC.arff		
Up	Down		
		Notes	

Figure 16: Weka Experimenter Advanced mode configuration interface. The top frame (1) is used to prepare the experiment: either prepare a new one, load or save an experiment. The frame below (2) is dedicated to saving the experiment report. The next frame (3) is the design of the experimental protocol: it controls the runs of experiment, the type of model build on the training sets and how they are evaluated. The bottom left frame (4) stores the datasets that are being processed.

weka.experiment	.ExplicitTestsetResultProducer
About	
Loads the ex SplitEvaluator	ternal test set and calls the appropriate to generate some results.
outputFile	splitEvalutorOut.zip
randomizeData	False
rawOutput	False
relationFind	
relationReplace	
splitEvaluator	Choose RegressionSplitEvaluator – W weka.classifiers.
testsetDir	testsets
testsetPrefix	
testsetSuffix	_test.arff
Open	Save OK Cancel

Figure 17: Configuration interface of the ExplicitTestsetResultProducer.

weka.experiment.RegressionSplitEvaluator								
About								
A SplitEvaluator that produces results for a classification scheme on a numeric class attribute.								
classifier	Choose GaussianProcesses -L 4.0 -N 2 -K "weka.classifiers.functions.supportVector.PolyKernel -E 1.0 -C 250007" -S 1							
noSizeDetermination	False	-						
Open.	OK Cancel							

Figure 18: Configuration interface of the RegressionSplitEvaluator.

Datasets: 4			
Resultsets: 1			
Confidence: 0.05 (two tailed)			
			Date: 10/06/16 15:22
Dataset	(1) f	unctions.	
CVIter1Fold1 affinity	(1)	80.30	
CVIter1Fold2_affinity	(1)	74.91	
CVIter1Fold3_affinity	(1)	73.11	
CVIter1Fold4_affinity	(1)	84.02	

Key: (1) functions.GaussianProcesses '-L 4.0 −N 2 −K \"functions.supportVector.PolyKernel −E 1.0 −C 250007\" −S 1' -8620066949967678545

Figure 19: *Example of Gaussian Processes performances on the external test set with a noise to a value of 4.*

Click the <i>Run</i> tab. Click the button <i>Start</i> . Click the <i>Analyse</i> tab to display the analysis interface of the experiment (Erreur ! Source du renvoi introuvable.). Click the button <i>Experiment</i> . Set the <i>Comparison field</i> to <i>Relative_absolute_error</i> . Click the button <i>Perform test</i> .	Typical results provided in Figure 19. Generally the results are improved compared to the cross-validation experiments (Figure 15). This is an expected result since now the entire training set is used for model fitting, in view of external cross-validation on the test set. By contrast, in the previous experiment, the training set itself underwent internal cross-validation, never being used as a whole for model fitting Second, it is interesting to note that the datasets that performed best in internal cross-validation are those that generalized less in external cross-validation. This paradoxal result, the discrepancies during the optimization of the noise of level of the Gaussian Processes, are clues that the dataset might contain some outliers.

Exercise 4:

The last part of the tutorial will focus on the identification of outliers in the dataset.

٠	Start the Weka Explorer software (Figure	If some points are outliers, then they
	20).	shall be difficult to fit by a non-
٠	• Click the button <i>Open file</i> and load the file IUPHAR_5HT2B_E_IIAB_R(2-4)_FC.arff.	overfitted model. The external cross-
		validation procedure basically
		identified a procedure to build a non-
		overfitted model on the dataset.
		Therefore, the full dataset is now
		loaded into the Weka Experimenter

	and will be studied in detail.
Preprocess Classify Cluster Associate Select attributes	Visualize
Open file Open URL Open DB Gen	nerate Undo Edit Save
Choose None	Apply
urrent relation	Selected attribute
Relation: /Users/marcou/Deskt Attributes: 1436 Instances: 88 Sum of weights: 88	Name: class Missing: 0 (0%) Distinct: 49 Unique: 24 (27%)
ttributes	Statistic Value
All None Invert Pattern No. Name	Minimum5.2Maximum10.05Mean7.339StdDev1.156
$1428 \sqcup (O=C^*N), (O=C^*O), xO$ $1429 \sqcup (O=C^*N^*N), (O=C^*N-C), (O=C^*O^*C), xO$ $1430 \sqcup (O^*C), (O^*C), xO$	Class: class (Num)
1431(0^-C^-N),(0^+C-C),(0^+C=O),XO 1432(0^+C^+N-C),(0^+C-C^+C),(0^+C-C^+C),XO 1433(C-C-C),(C-C-C),(C-N^+C),(C-N^+N),XC 1434(C-C-C-N&FC+1&),(C-C-C-N&FC+1&),(C 1435(C-C-C-C),(C-C-N&FC+1&-C),(C-C-N&FC+ 1436(class	
Remove	3
tatus	5.2 7.63 10.0
OK	Log 💉 🕬

Figure 20: The Weka Explorer software preprocessing mode.

Click the C	Classify tab (Figu	re 21).		During	this	procedure,	the
 Click the GaussianF Click the C the configuration 	button <i>Choose</i> Processes item. GaussianProcess Iration interface	and select ses word and as in Figure 1	the set 3.	Gaussiar optimal s Experime	nProcess etup ider enter.	es is used with ntified with the b	n the <i>Veka</i>
 Click the organization set the Forget Click the body 	Cross-validation ds value to 4. utton <i>Start.</i>	radio button a	and	More de (Figure 2 consister the Weke	tailed sta 22). The nt with th a Experin	atistics are obta e performances nose obtained u nenter software.	ained are using

Preprocess Classify Cluster Asso	ciate Select attributes Visualize			
Choose GaussianProcesses -L 4.0 -N 2 -K "weka.classifiers.functions.supportVector.PolyKernel -E 1.0 -C 250007" -S 1				
Test options	Classifier output			
Use training set				
o ose training set	81,5.8,5.984,0.184			
Supplied test set Set	02,/.31,/.210,-0.234 92.6.2.6.221.0.021			
	84 5 9 5 958 8 858			
Cross-validation Folds 4	85 7 48 7 560 6 680			
Percentage split % 66	86.8.7 8.2740.426			
O Percentage spin % 00	87.6.05.6.064.0.014			
More ontions	88.7.3.7.598.0.298			
	00//10//100001200			
,	=== Evaluation on test set ===			
(Num) class	Time taken to test model on supplied test set: 0.02 seconds			
Start Stop	=== Summary ===			
esult list (right-click for options) 16:19:15 - functions.GaussianPro 16:20:38 - functions.GaussianPro 16:21:44 - functions.GaussianPro 16:22:06 - functions.GaussianPro	Correlation coefficient0.979Mean absolute error0.2359Root mean squared error0.2917Relative absolute error23.9013 %Root relative squared error25.3818 %Total Number of Instances88			
itatus				
ОК	Log 🛷 x C			

Figure 21: Interface of the classification mode of Weka Explorer.

```
Instances:
              88
Attributes:
              1436
              [list of attributes omitted]
              4-fold cross-validation
Test mode:
=== Classifier model (full training set) ===
Gaussian Processes
Kernel used:
  Linear Kernel: K(x,y) = \langle x,y \rangle
All values shown based on: No normalization/standardization
Average Target Value : 7.3390909090909044
Inverted Covariance Matrix:
    Lowest Value = -0.01993988339279234
    Highest Value = 0.04610399059836405
Inverted Covariance Matrix * Target-value Vector:
    Lowest Value = -0.06391563323426866
    Highest Value = 0.04158533375446235
Time taken to build model: 0 seconds
=== Cross-validation ===
=== Summary ===
Correlation coefficient
                                          0.5175
Mean absolute error
                                          0.7924
Root mean squared error
                                          0.9886
Relative absolute error
                                         79.8931 %
Root relative squared error
                                         85.4517 %
Total Number of Instances
                                         88
```

Figure 22: Cross-validation results of Gaussian Processes in the Weka Explorer software.

•	Click the Supplied test set radio button	During this procedure, the
	then, the Set button.	GaussianProcesses is used with the
•	Choose the file	optimal setup identified with the Weka
	IUPHAR_5HT2B_E_IIAB_R(2-4)_FC.arff as test set file.	Experimenter.
•	Click the button <i>More options</i> (Figure 23).	Some statistics are reported (Figure
•	In the configuration interface that opens,	24). The performances are consistent
	click the button Choose then select the	with those obtained using the Weka
	CSV item.	
•	Click the Start button.	

Right click on the line of the Result list corresponding to the fit results and select	Experimenter software.
the option Save result buffer.	The estimated affinity value of each
 Name the output file GP_Fit_all.out. 	compound, along with the
	experimental value are provided into
	the log of the calculations. This output
	will be analyzed with another software,
	and therefore it must be saved into the
	file GP Fit all.out.

☑ Output model		
✓ Output per-class stats		
Output entropy evaluation measures		
✓ Output confusion matrix		
✓ Store predictions for visualization		
Error plot point size proportional to margin		
Output predictions Choose CSV		
Cost-sensitive evaluation Set		
Random seed for XVal / % Split 1		
Preserve order for % Split		
Output source code WekaClassifier		
Evaluation metrics		
ОК		

Figure 23: The additional options of the classify tool of Weka Explorer.

75,8.05,8.003,-0.047 76,5.9,6.059,0.159 77,6.8,6.677,-0.123 78,8.9,8.504,-0.396 79,5.8,6.114,0.314 80,8.95,8.692,-0.258 81,5.8,5.984,0.184 82,7.51,7.216,-0.294 83,6.2,6.221,0.021 84,5.9,5.958,0.058 85,7.48,7.569,0.089 86,8.7,8.274,-0.426 87,6.05,6.064,0.014 88,7.3,7.598,0.298 === Evaluation on test set === Time taken to test model on supplied test set: 0.02 seconds === Summary === Correlation coefficient 0.979 Mean absolute error 0.2359 Root mean squared error 0.2917 Relative absolute error 23.9013 % Root relative squared error 25.3818 % Total Number of Instances 88

Figure 24: Weka Gaussian Processes model fit statistics.



Figure 25: To save the log of the calculation, right-click on the line of the Result list corresponding to the experiment to save, then select the Save result buffer option.

 Open the file GP_Fit_all.out using the software <i>ModelAnalyzerR</i> (Figure 26). Use the file IUPHAR_5HT2B.sdf as the source of information on chemical structures. Then click the <i>OK</i> button. Search for the worst predicted compound and identify this compound in you <i>InstantJChem</i> database. Eind the article referring to this molecule. 	The ISIDA/ModelAnalyzerR software computes performance statistics of the models, provide interactive experimental versus estimated values and REC plots. Each point on the plots is linked the chemical structure information in the provided SDF file.
and search for a reason to keep or exclude the compound.	It is easy to notice an outlier. The Grubbs test outputs a value of 3.4 located in between a risk of 5% and 1%. This is therefore a data point to consider.
	The point corresponds to the entry 19 of the database, melatonin (Figure 27). This value is determined in the reference [18], provided as the file 12750432.pdf.

melatonin was reported by displacement of radiolabeled melusergine. However, on page 957 the authors stated: "Melatonin (...) partially attenuated the action of 5-HT, although only about 50% of inhibition was acquired even at a concentration of 100 µM. It was not possible, for reasons of solubility, to evaluate higher concentrations of melatonin." In other words, the ligand is weak and measured at the limits of its solubility, in these conditions, the reported value is better understood as a qualitative, importance highlighting the of agomelatine. This is a good reason to exclude the entry from the study.



Figure 26: Interface of the software ISIDA/ModelAnalyzerR. The names of input files are provided in area 1. If multiple models are available, they can be selected in area 2. Molecules are depicted in area 3. Statistical parameters are reported in the text box 4. The experimental versus estimated values are plotted in area 5. The REC is plotted in area 6.



Figure 27: *Melatonin. The first outlier identified in the dataset. The database activity is 5.2.*

•	In the InstantJChem database, get the	In this step a new dataset is created
	Original Cala of the outlier.	without the outlier chemical structure.
•	Click the button Query.	
٠	In the Cell Original Cdld, type the	
	command "Not in list" followed by the	
	Original Cdld of the outlier.	
٠	Click the button Run Query.	
•	Click the button Export to file (\triangleq , Figure 8).	
	Save the file in SDF format as IUPHAR-	
	5HT2B-33 sdf in the directory of the	
	tutorial	
•	Click the button <i>Next</i> and be sure not to	
	include the fields CdId, Mol Weight and	
	Formula.	
•	Click the button Next then Finish.	
•	In the xFragmentor software, click the	During this step the ISIDA SMF
	button Read XML. Chose the file	molecular descriptors are computed

•	IUPHAR_5HT2B_E_IIAB_R(2-4)_FC.xml. Remove all SDF files from the left hand side window with the <i>Remove SDF</i> button.	for this new dataset.
•	Click the <i>Add SDF</i> button and add the file IUPHAR-5HT2B-33.sdf.	
•	Click the button Run!.	
•	In the <i>xFragmentor</i> software, click the button <i>Read XML</i> . Chose the file IUPHAR_5HT2B_E_IIAB_R(2-4)_FC.xml. Remove all SDF files from the left hand side window with the <i>Remove SDF</i> button	During this step the ISIDA SMF molecular descriptors are computed for this new dataset.
•	Click the <i>Add SDF</i> button and add the file IUPHAR-5HT2B-33.sdf. Click the button <i>Run!</i> .	
•	Start the <i>Weka Explorer</i> software (Figure 20).	A new <i>GaussianProcesses</i> is build using the <i>Weka</i> software. The model
•	Click the button <i>Open file</i> and load the file IUPHAR-5HT2B-33_IIAB_R(2-4)_FC.arff.	building uses the same optimal paramterization as discovered
•	Click the <i>Classify</i> tab.	
•	Click the button <i>Choose</i> and select the <i>GaussianProcesses</i> item.	During the cross-validation procedure,
•	Click the <i>GaussianProcesses</i> word and set the configuration interface as in Figure 13.	enhanced. At the fitting stage the
•	Click the <i>Cross-validation</i> radio button and set the <i>Folds</i> value to 4.	Same Observation is made.
•	Click the button Start.	
•	Click the <i>Supplied test set</i> radio button then, the <i>Set</i> button.	
•	Chose the file IUPHAR_5HT2B- 33_E_IIAB_R(2-4)_FC.arff as test set file.	
•	Click the button <i>More options…</i> (Figure 23).	
•	In the configuration interface that opens, click the button <i>Choose</i> then select the <i>CSV</i> item.	
•	Click the Start button.	
•	Right click on the line of the <i>Result list</i>	
•	the option Save result buffer.	
-		

=== Cross-validation ===	
=== Summary ===	
Correlation coefficient	0.5157
Mean absolute error	0.7881
Root mean squared error	0.9771
Relative absolute error	78.6143 %
Root relative squared error	83.9261 %
Total Number of Instances	87

Figure 28: Cross-validation results of Gaussian Processes, without the melatonin data.

=== Summary ===

0.9818
0.2299
0.2705
23.6221 %
23.881 %
87

Figure 29: Gaussian Process fit statistics, without melatonin.

•	Open the file GP_Fit_all-33.out using the	Now, there are no compounds that
	software ModelAnalyzerR. Use the file	can be identified as an outlier, or even
•	IUPHAR_5HT2B-33.sdf as the source of information on chemical structures. Then click the <i>OK</i> button.	be suspected to be an outlier.

Conclusion

During this tutorial, a very cautious workflow is illustrated, using external cross-validation to assess the predictive power of the QSAR model generated and internal cross-validation to estimate the optimal parameters of the machine learning method used. These optimal parameters are a guarantee that the models will be neither over-fitted nor under-fitted.

During the process, it was suspected that some point of the initial data could be pathological. The machine learning method with optimal parameters is used to fit the training set. The outlier data should be the data point most difficult to fit. Removing this point, the performances of the model are improved.

An important point, not illustrated by this tutorial, is that outliers may also depend on the molecular descriptors and of the machine learning method used. Therefore, an obvious solution to make outlier detection more robust is to repeat the procedure with different molecular descriptors and different machine learning algorithms.

Using this strategy, it is possible to prioritize several suspicious points. Yet, if the content of the dataset has changed because of the removing of one or more outliers, the whole procedure of internal cross-validation optimization of the machine learning parameters and of the external cross-validation estimate of generalization should be repeated.

References

- 1. Southan, C., et al., *The IUPHAR/BPS Guide to PHARMACOLOGY in 2016: towards curated quantitative interactions between 1300 protein targets and 6000 ligands.* Nucleic acids research, 2015: p. gkv1037.
- 2. Barnes, N.M. and J.F. Neumaier, *Neuronal 5-HT receptors and SERT.* Tocris bioscience scientific review series, 2011. **34**: p. 1-15.
- 3. Roth, B.L., *Drugs and valvular heart disease*. N Engl J Med, 2007. **356**(1): p. 6-9.
- Rothman, R.B., et al., Evidence for possible involvement of 5-HT2B receptors in the cardiac valvulopathy associated with fenfluramine and other serotonergic medications. Circulation, 2000. 102(23): p. 2836-2841.
- 5. Connolly, H.M., et al., *Valvular heart disease associated with fenfluramine–phentermine.* New England Journal of Medicine, 1997. **337**(9): p. 581-588.
- 6. *FDA Announces Withdrawal Fenfluramine and Dexfenfluramine (Fen-Phen)*, U.S.F.A.D. Administration, Editor. 1997.
- 7. Frachon, I., et al., *Benfluorex and unexplained valvular heart disease: a case-control study.* PloS one, 2010. **5**(4): p. e10128.
- Rodrigo Andrade, N.M.B., Gordon Baxter, Joel Bockaert, Theresa Branchek, Marlene L. Cohen, Aline Dumuis, Richard M. Eglen, Manfred Göthert, Mark Hamblin, Michel Hamon, Paul R. Hartig, René Hen, Katharine Herrick-Davis, Rebecca Hills, Daniel Hoyer, Patrick P. A. Humphrey, Klaus Peter Latté, Luc Maroteaux, Graeme R. Martin, Derek N. Middlemiss, Ewan Mylecharane, Stephen J. Peroutka, Pramod R. Saxena, Andrew Sleight, Carlos M. Villalon, Frank Yocca. 5-Hydroxytryptamine receptors: 5-HT2B receptor. IUPHAR/BPS Guide to PHARMACOLOGY 2016 12/08/2015 08/06/2016]; Available from: http://www.guidetopharmacology.org/GRAC/ObjectDisplayForward?objectId=7.
- 9. *PubMed*. U.S. National Library of Medecine 08/06/2016]; Available from: <u>http://www.ncbi.nlm.nih.gov/pubmed/?term=</u>.
- 10. MacKay, D.J., *Introduction to Gaussian processes*. NATO ASI Series F Computer and Systems Sciences, 1998. **168**: p. 133-166.
- 11. Rasmussen, C.E., *Gaussian processes for machine learning*. 2006.
- 12. Neal, R.M., *Monte Carlo implementation of Gaussian process models for Bayesian regression and classification.* arXiv preprint physics/9701026, 1997.
- 13. Maimon, O. and L. Rokach, *Data mining and knowledge discovery handbook*. Vol. 2. 2005: Springer.
- 14. Aggarwal, C.C. *Outlier analysis*. in *Data Mining*. 2015. Springer.
- 15. Ruggiu, F., et al., *Quantitative Structure-Property Relationship Modeling: A Valuable Support in High-Throughput Screening Quality Control.* Analytical Chemistry, 2014. **86**(5): p. 2510-2520.
- 16. Grubbs, F.E., *Sample criteria for testing outlying observations*. The Annals of Mathematical Statistics, 1950: p. 27-58.
- 17. Pearson, E.S. and C.C. Sekar, *The efficiency of statistical tools and a criterion for the rejection of outlying observations*. Biometrika, 1936. **28**(3/4): p. 308-320.
- 18. Millan, M.J., et al., *The novel melatonin agonist agomelatine (S20098) is an antagonist at 5-hydroxytryptamine2C receptors, blockade of which enhances the activity of frontocortical dopaminergic and adrenergic pathways.* Journal of Pharmacology and Experimental Therapeutics, 2003. **306**(3): p. 954-964.