User manual of

FS-Studio

for Weka Explorer

Index

1 INTRODUCTION

The application is an information system that allows attribute selection studies to be carried out on data sets designed for supervised learning. For this, optimization algorithms based on metaheuristics are provided as search algorithms to find the most relevant subsets of attributes, for which the free Weka analysis tool is used. Statistical tests have been included to verify the veracity of the results obtained with the different algorithms, specifically, Bayesian tests, which is in a Python file (since the library is of this programming language) and with which it has been carried out. a connection from Java.

In addition, the integration of the CVOA algorithm developed by the Department of Data Science & Big Data of the Pablo de Olavide University has been carried out to have one more optimization algorithm.

When performing an experiment, datasets can be added and deleted, as well as modifying their classIndex and positive class, and loading them again in the Preprocess tab. On the other hand, the management of search and integration algorithms, as well as classification/regression algorithms, is also allowed. It is also possible to choose the validation method (hold-out split, cross validation and leave-one out) and configure them (percentage allocated to training in hold-out Split and number of bags in cross validation). It also allows us to execute the experimentation with the number of threads that we want, thus doing the execution in parallel. This progress is shown by a bar at the bottom.

Once the experimentation has been carried out and the data obtained, we can see these results grouped into 4 components: Predictions (where the predictions are shown with their real and predicted value together with the combination to which they belong), Metrics (where we can choose the metrics to display next to each combination, choosing between 12 classification and regression metrics), Graph (where it allows us to view a metric to choose based on the dataset, evaluator, search engine or classifier) and Statistical tests (where information related to the tests appears). statistics as well as their results). Also, different ways of exporting the results are included, such as CSV, ARFF, etc., even to a relational DB from which the previously exported data can also be loaded.

Each and every one of the functionalities will be explained and detailed in the following points.

2 INTERFACE DESCRIPTION

The developed interface consists of two tabs: Experiment and Results.

In the Experiment tab we have 4 components plus the one belonging to the execution of the experimentation, these components are:

- datasets
- Feature Selection
- Classifier
- validation

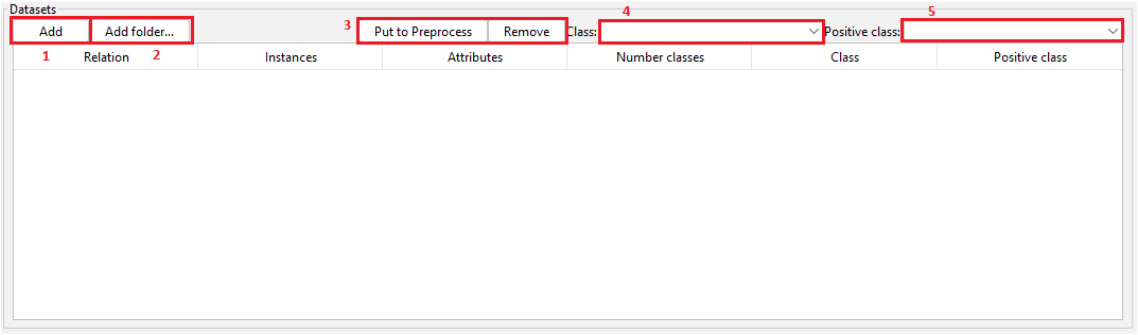
2.1 Experiment tab

2.1.1 Datasets

The Datasets section is made up of the following elements:

1. **Add button:**adds the dataset previously loaded in the Weka Preprocess tab to the Datasets table and as such, also to the experimentation to be executed.
2. **Add folder button:**adds all the datasets that are present in the selected folder in arff format to the Datasets table and as such, also to the experimentation to be executed.
3. **Action Buttons:**the Put to Preprocess button loads the dataset previously selected in the table in the Weka Preprocess tab. The Remove button removes the previously selected dataset from the table, as well as from experimentation.
4. **ComboBoxClass:**displays the possible values of classIndex of the dataset selected in the table, thus being able to set its current value. When selecting a dataset in the table, the value of said comboBox is set to the value of the selected dataset.
5. **ComboBox Positive Class:**displays the possible positive class values of the selected dataset in the table, thus being able to set its current value. When selecting a dataset in the table, the value of said comboBox is set to the value of the selected dataset.
6. **Table:**table where the datasets added to the experimentation with the Add button will appear. Its fields are:

    - **Relationship:**dataset name.
    - **Instances:**number of rows in the dataset.
    - **Features:**number of dataset attributes.
    - **Numberclasses:**number of classes in the dataset.
    - **Class:**indexClass currently selected for the dataset.
    - **Positiveclass:**positive class currently selected for the dataset.

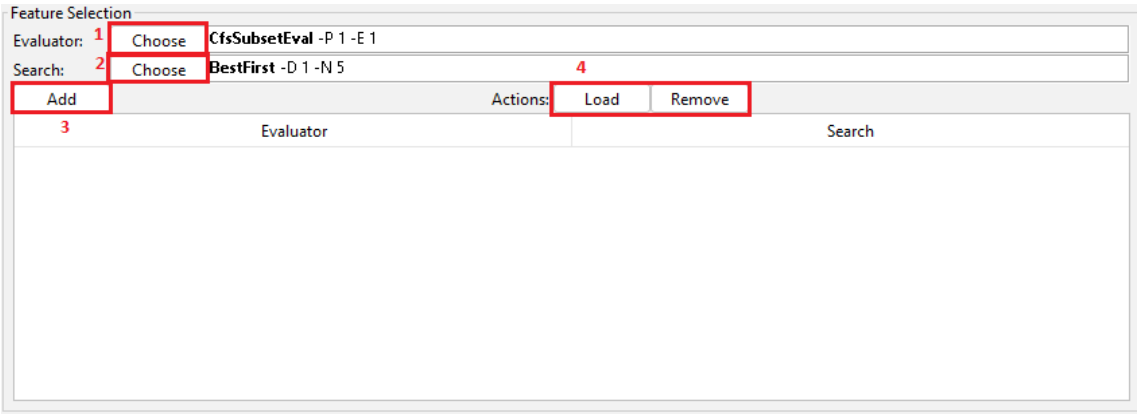


2.1.2 Feature Selection

The Feature Selection section consists of the following elements:

1. **Choose Evaluator button:**When clicking, a menu is displayed where we can choose the evaluation algorithm that we want among those available. If we want to change the options of said algorithm, we must click on its textField and a menu will pop up with its options where we can change them.
2. **Choose Search button:**When clicking, it displays a menu where we can choose the search algorithm that we want among those available. If we want to change the options of said algorithm, we must click on its textField and a menu will pop up with its options where we can change them.
3. **Add button:**adds the algorithms previously chosen, configured and loaded in their respective textFields to the table, and therefore to experimentation.
4. **Action Buttons:**the Load button loads the previously selected row of the table (ie a combination of Evaluator with Search) into their respective textFields. The Remove button removes the previously selected row from the table, as well as from experimentation.
5. **Table:**table where the evaluation and search algorithms added to the experimentation with the Add button will appear. Its fields are:

    - **Assessor:**added evaluator name and options.
    - **Search:**added search algorithm name and options.



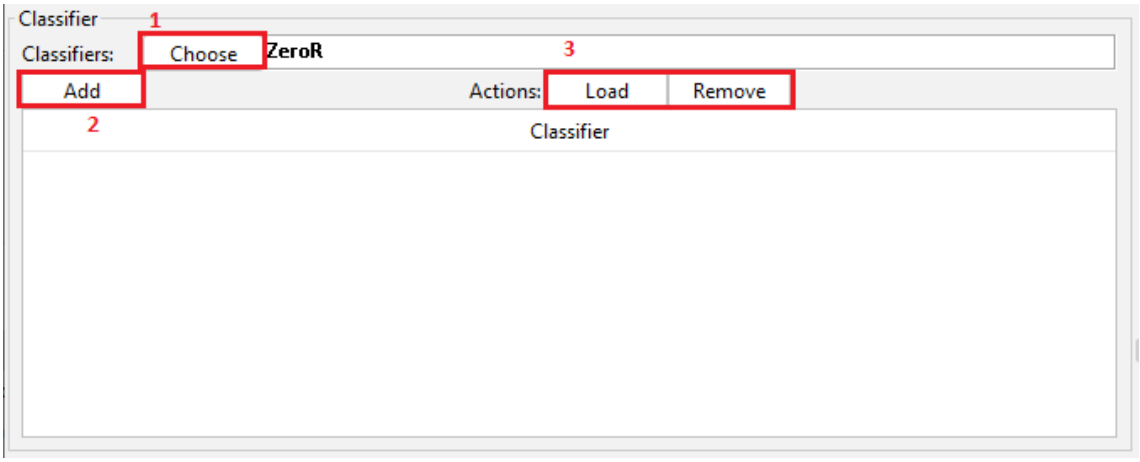
2.1.3 Classifier

The Classifier section is made up of the following elements:

1. **Choose button:**When clicking, it displays a menu where we can choose the classification/regression algorithm that we want among those available. If we want to change the options of said algorithm, we must click on its textField and a menu will pop up with its options where we can change them.
2. **Add button:**adds the algorithm previously chosen, configured and loaded in its textField to the table, and therefore to experimentation.

3. **Action Buttons:**the Load button loads the previously selected row of the table into its textField. The Remove button removes the previously selected row from the table, as well as from experimentation.
4. **Table:**table where the search/regression algorithms added with the Add button previously will appear. Its fields are:

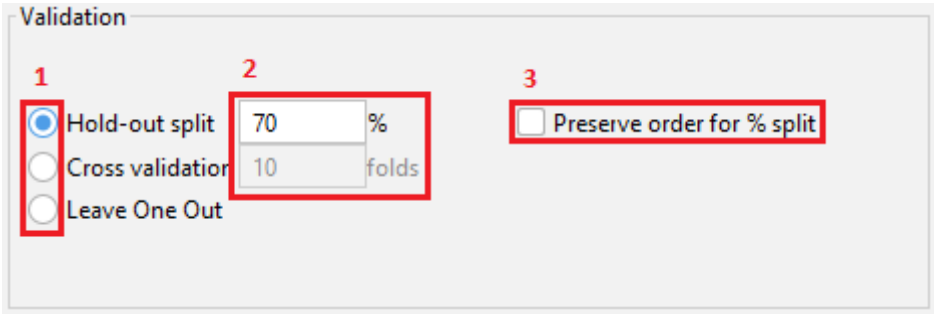
    - **Classifier:**added classification/regression algorithm name and options.



2.1.4 Validation

The Validation section consists of the following elements:

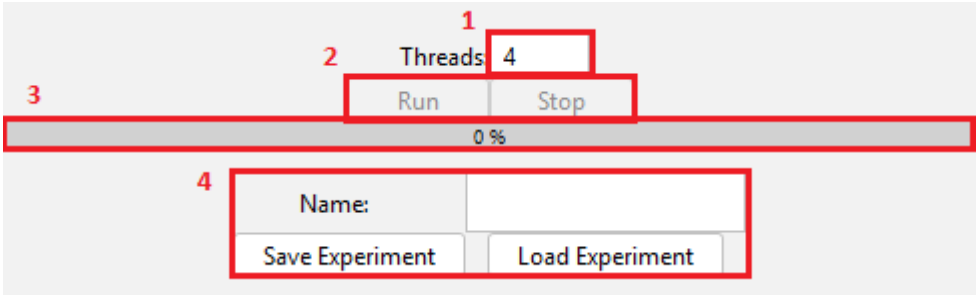
1. **Radio buttons:**allows us to choose the validation method that we want to apply to the experimentation. Only one can be used, and when this method is chosen, if it has a possible configuration, its corresponding textField will be enabled.
2. **Configuration TextFields:**allows us to configure (if selected) the available options for the Hold-out Split method (percentage intended for training), Cross validation (number of bags) and Preserve order Split (maintain the order of the instances in the datasets when doing hold-out split).
3. **Check box:**allows us to maintain the order of the dataset instances when performing the separation in the hold-out split validation.



2.1.5 Execution

The Execution section consists of the following elements:

1. **TextFields threads:**It allows us to configure the number of threads with which the experimentation will be executed in parallel.
2. **Execution buttons:**the Run button starts executing the experimentation, which can be stopped with the Stop button.
3. **Progress bar:**it shows us the current progress of the execution, in case of stopping it, said value is set to zero.
4. **Save and load experiment:**It shows us the corresponding configuration to save the data of an experiment (everything that is seen in the Experiment tab). We can enter the name of the experiment and then load it by selecting the place where we have previously saved it.



2.2 Results tab

2.2.1 Predictions

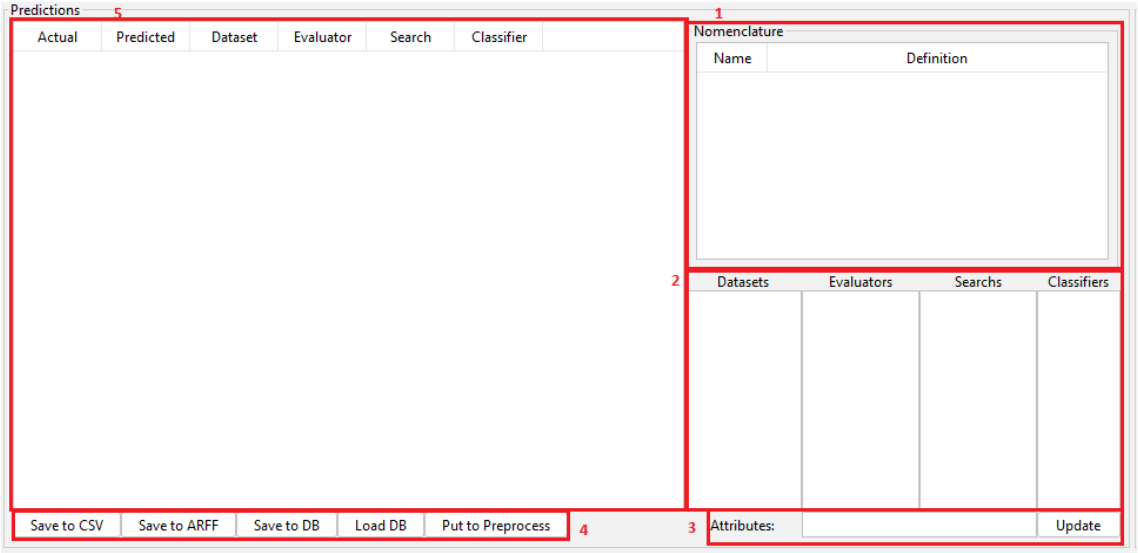
The Predictions section consists of the following elements:

1. **Nomenclature:**shows the legend of the different algorithms used (both evaluation and search as well as regression/classification).
2. **Lists:**lists where the datasets, evaluation and search algorithms and classification/regression algorithms, respectively, with which the experimentation has been carried out, appear in each of them. In addition, the "original" value will also appear in the Evaluators and Search lists referring to the combination that does not use search or evaluation algorithms. Allows you to select and deselect values which act as a filter.
3.
4. **TextField Attributes:**allows us to enter the attributes that we want to be displayed in the table as one more column of the different datasets that appear (its value can be empty).
5. **Action Buttons:**buttons that allow us to export and import data and update the table, these are:
    - **Save to CSV:**allows you to export the data contained in the table in CSV format to the chosen directory.
    - **Save to ARFF:**allows you to export the data contained in the table in ARFF format to the chosen directory.
    - **Save to DB:**allows you to export the data contained in the Predictions table and in the Metrics table (it exports the values of all the metrics for the combinations, whether or not they are chosen) to a database with the parameters indicated in

the DatabaseUtils.props file (its configuration will be seen later). If the database or any table is not previously created, it will be created automatically by using sqls scripts.

- **Put to Preprocess:**allows the loading of the table data in the Preprocess tab of Weka.
- **LoadDB:**allows you to load the data previously saved in the database to the Predictions and Metrics tables.
- **Update:**updates the table with the previously selected datasets, evaluation and search algorithms, and classification/regression algorithms in their respective lists. In addition, it adds the attributes indicated in the textField Attributes to the table in the form of columns.

6. **Table:**table where the results obtained from the experimentation referring to the predictions will appear. Its fields are:
   - **Actual value:**returns the original value of that instance in the dataset.
   - **Predictedvalue:**displays the predicted value for that instance.
   - **Data set:**name of the dataset to which it belongs.
   - **Assessor:**name of the evaluator together with its options with which that value has been obtained.
   - **Search:**name of the search algorithm together with its options with which that value has been obtained.
   - **Classifier:**name of the classification/regression algorithm together with its options with which that value has been obtained.
   - **Attributes i (optional):**each column corresponding to the attributes entered in the textField Attributes. Each row indicates the value of that attribute for the instance of that row.



2.2.2 Metrics

The Metrics section consists of the following elements:

1. **Classification:**ranking metrics. Each metric has an associated checkBox which, when pressed, inserts the column corresponding to said metric in the table.
2. **Regression:**metrics related to regression. Each metric has an associated checkBox which, when pressed, inserts the column corresponding to said metric in the table.
3. **Action Buttons:**buttons that allow us to export and load table data, these are:

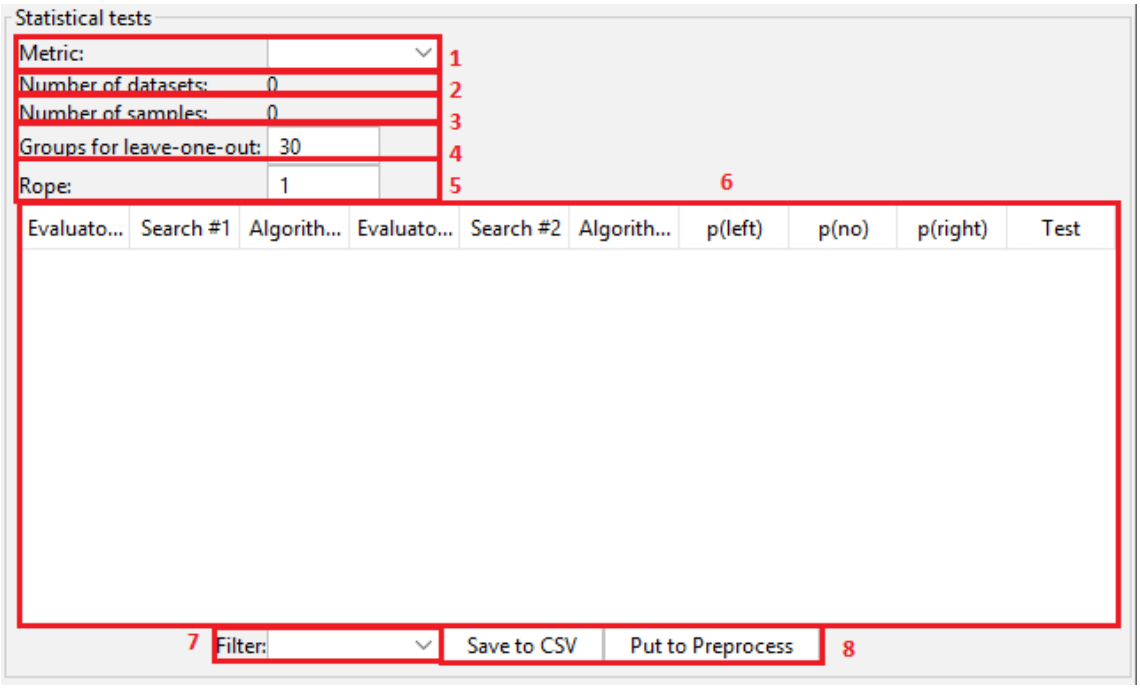
    - **Save to CSV:**allows you to export the data contained in the table in CSV format to the chosen directory.
    - **Save to ARFF:**allows you to export the data contained in the table in ARFF format to the chosen directory.
    - **Save to Latex:**allows you to export the data contained in the table in LATEX format to the chosen directory.
    - **Put to Preprocess:**allows the loading of the table data in the Preprocess tab of Weka.
4. **Table:**table where the results obtained from the experimentation regarding the metrics for each combination (experiment) will appear. Its fields are:

    - **Data set:**name of the dataset to which it belongs.
    - **Assessor:**name of the evaluator together with its options with which that value has been obtained.
    - **Search:**name of the search algorithm together with its options with which that value has been obtained.
    - **Classifier:**name of the classification/regression algorithm together with its options with which that value has been obtained.

2.2.3 Statistical Tests

The Statistical Tests section is made up of the following elements:

1. **Metric:**comoboBox to choose the metric with which the statistical tests will be calculated. The metrics to show will depend on the datasets chosen when doing the experimentation (classification or regression). By default, the statistical tests will be calculated with the metric F-measure for classification and MAE for regression. As soon as the metric is changed, the tests will be recalculated with the chosen metric.
2. **Number of datasets:**number of datasets with which the experimentation is carried out.
3. **Number of samples:**number of examples contained in the statistical tests.
4. **Groups for leave-one-out:**number of groups that will be made to obtain the metrics and which will be used to carry out the statistical tests. By default 30. It only applies in the case of LOO.
5. **Rope:**the region of practical equivalence (rope) is specified by the caller and should correspond to what is "equivalent" in practice. By default it will be 1, as soon as its value changes and "Enter" is pressed, the tests will be calculated with that new value.
6. **Table:**table where the results of the statistical tests appear. The results are compared with other combinations, hence the duplication of the Evaluator, Search and Algorithm columns. Its fields are:
    - **Assessor:**appraiser corresponding to that combination.
    - **Search:**search algorithm used in that combination
    - **Algorithm:**classification/regression algorithm used in that combination.
    - **p(left):**probability of the first set (evaluator #1, search #1, algorihtm #1).
    - **p(not):**probability of none better.
    - **p(right):**probability of the second set (evaluator #2, search #2, algorihtm #2).
    - **Test:**indicates which combination of algorithms has been better, 1 in case of the first combination (indicated with #1), 2 in case of the second (indicated with #2) or 0 in case neither of the two is significantly better than the other.
7. **filter:**allows us to filter the results obtained in the table by the values of the Test column, being able to show all (All, is the default value), only the positive ones (Positives, values 1 and 2) or only the negative ones (Negatives, value 0).
8. **Action Buttons:**buttons that allow us to export and load table data, these are:
    - **Save to CSV:**allows you to export the data contained in the table in CSV format to the chosen directory.
    - **Put to Preprocess:**allows the loading of the table data in the Preprocess tab of Weka.



2.2.4 Chart

The Graph section consists of the following elements:

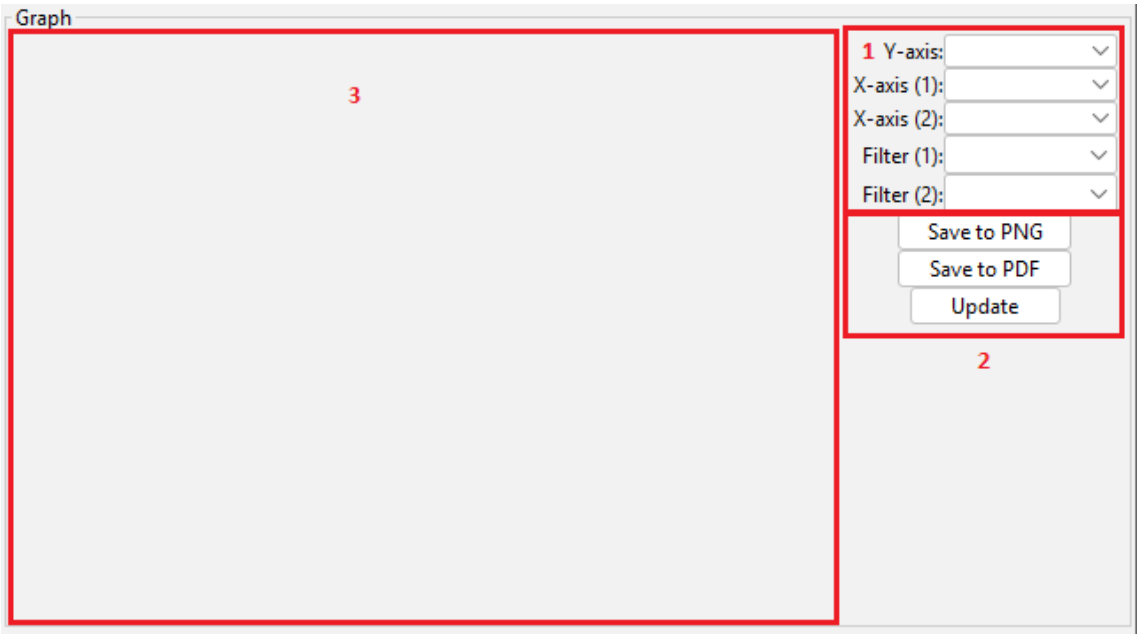
1. **Combo Boxes:**allow you to configure the grouped bar graph with the desired values. They are divided into:

    - **Metric:**metric that we want to appear on the y-axis of the graph. The available values are those of the 12 metrics in the Metrics section.
    - **X axis (1):**value that we want to appear on the X axis. The available values are: dataset, search, evaluator and classifier.
    - **X axis (2):**value that we want to appear that the bars mean. Available values are: dataset, search, evaluator, and classifier.
    - **Filter(1):**Both the label of this field and its values will depend on the values placed in the X-axis comboBoxes. For example, if its values are Dataset and Search, the label of this field will be Evaluator (which will contain the nomenclature of the evaluators with which the experimentation has been carried out) and of the Filter (2) will be Classifier (which will contain the nomenclature of the classifiers/regressors with which the experimentation has been carried out) and will be the values of the combinations shown in the graph.
    - **Filter(2):**Both the label of this field and its values will depend on the values placed in the X-axis comboBoxes. For example, if its values are Dataset and Search, the Filter label (1) will be Evaluator (which will contain the nomenclature of the evaluators with which the experimentation has been carried out) and this field will be Classifier (which will contain the nomenclature of the classifiers/regressors with which the experimentation has been carried out) and will be the values of the combinations shown in the graph.

2. **Action Buttons:**buttons that allow us to export and create the graph, these are:

    - **Save to PNG:**exports the graph to the chosen directory in PNG format.
    - **Save to PDF:**exports the graph to the chosen directory in PDF format.
    - **Update:**creates a new graph replacing the previous one, configured with the values indicated in the Metric, X-axis (1), X-axis (2), Filter (1) and Filter (2) comboBoxes.

3. **Graph:**where the graph will appear once generated.



3 MAIN USE CASE

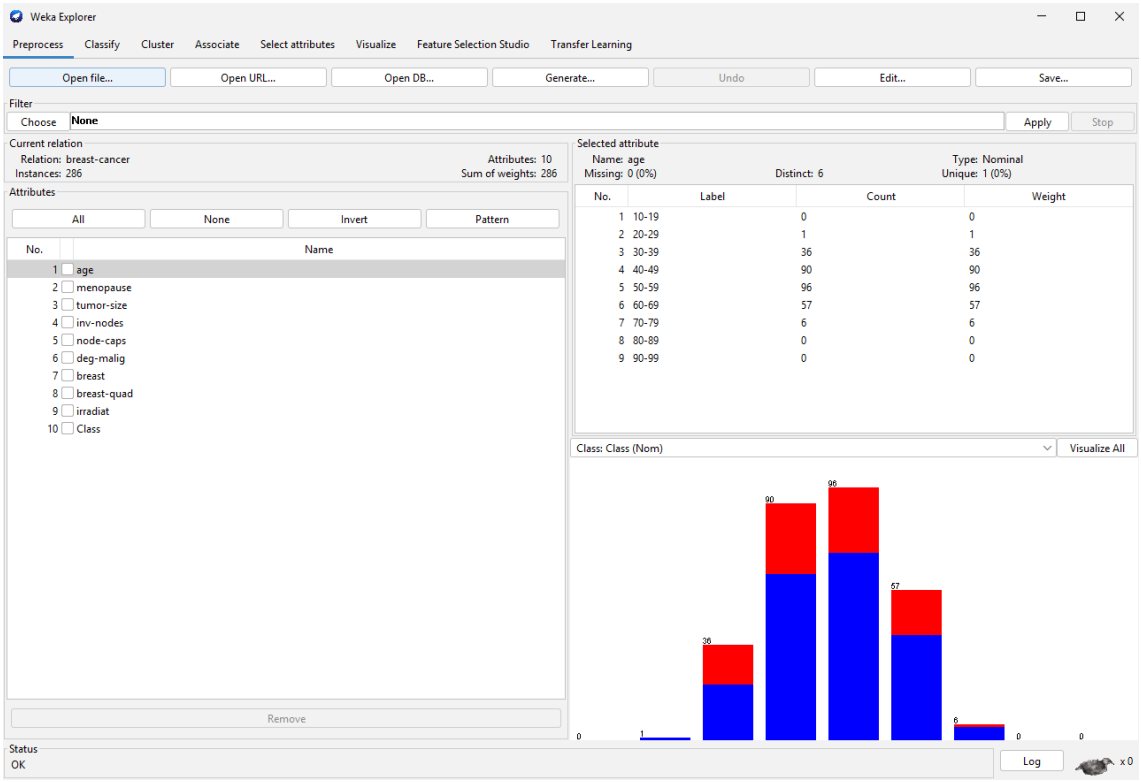
Once each of the different parts of the interfaces have been explained and detailed, and having previously installed the FeatureSelectionStudio package in Weka, an example of the normal flow of a user when using the information system is shown, adding several sets of data, along with several search algorithms, evaluation, classification, as well as choosing an evaluation method and the number of threads with which the experimentation will be executed. Also, we will see the results, we will filter them, we will add new columns, we will export and load data and we will generate graphs:

3.1 Experimentation

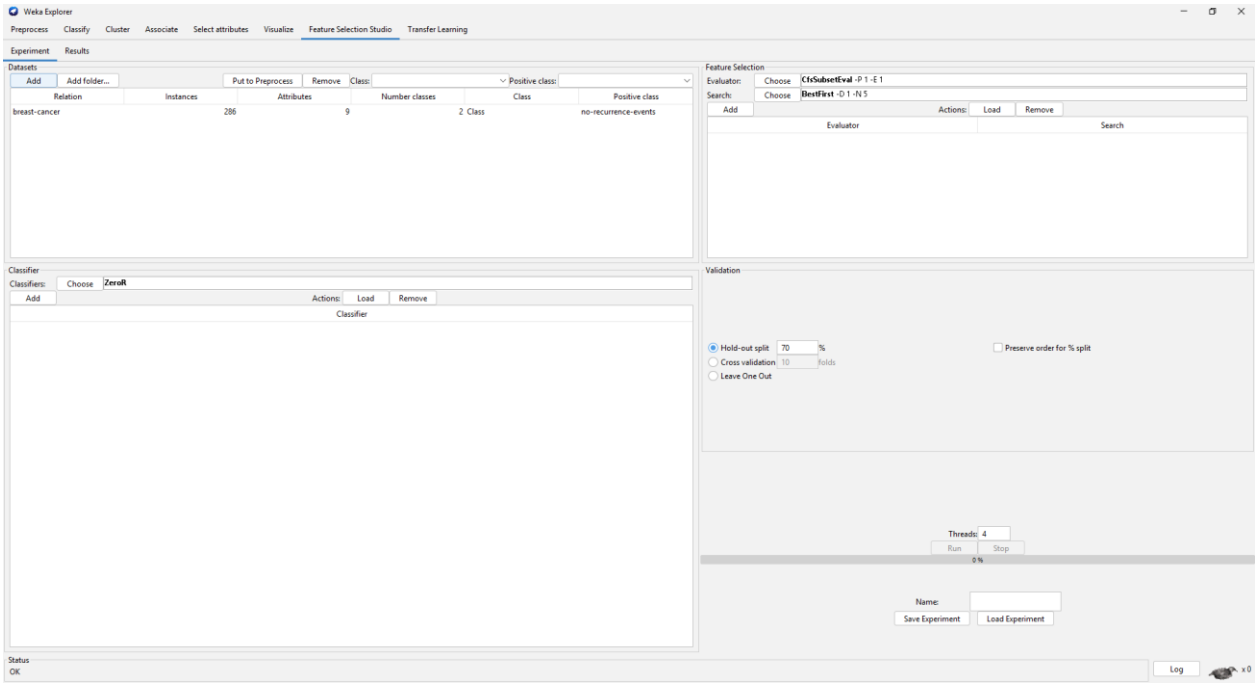
STEP 1: Datasets

The first thing we must do is in the Preprocess tab, load the dataset that we want to add to the experiment. I will show the steps to add 1 since with the rest it is exactly the same.

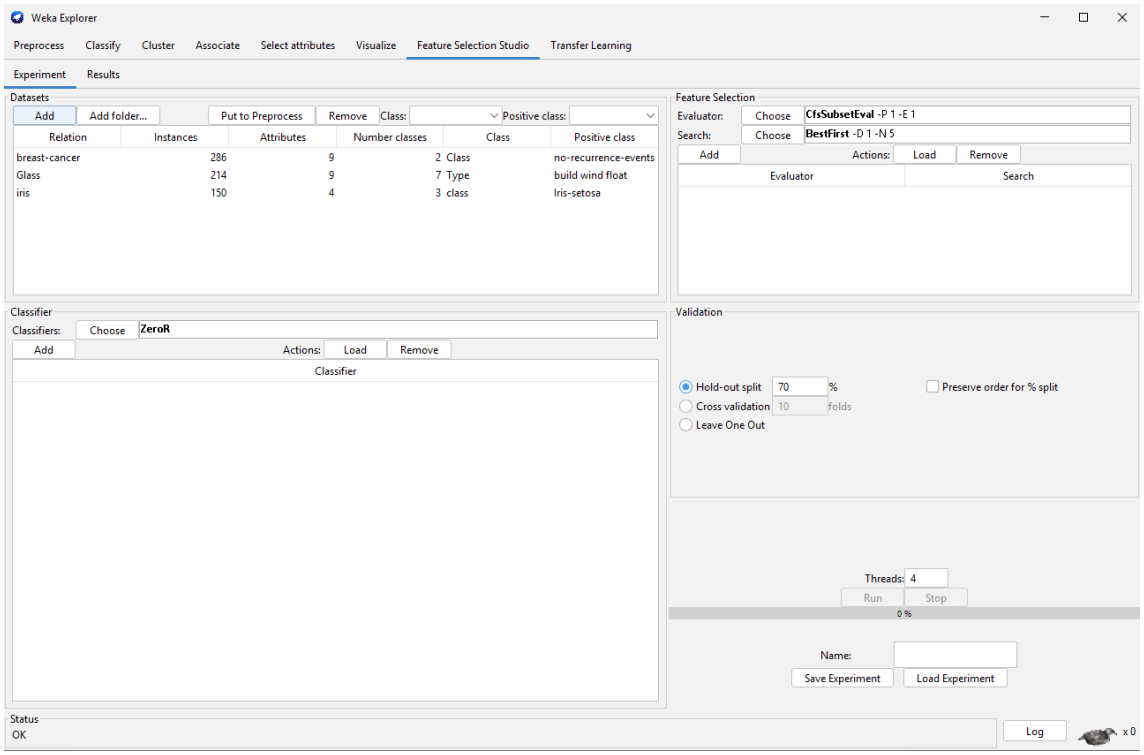
In Preprocess we click on OpenFile, the directory menu will open to select the dataset, in my case I will select breast-cancer and we will open it, thus proceeding to load it in the Preprocess tab:



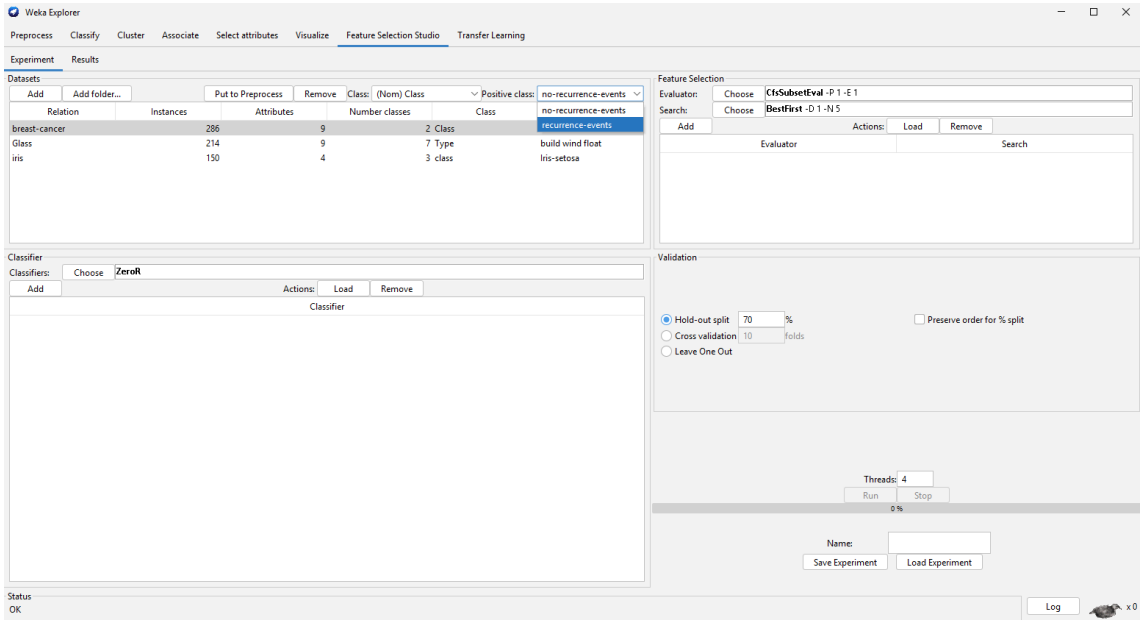
Now we go to the FeatureSelectionStudio tab, what we want to do is add the dataset that we just loaded to the experimentation, so we hit Add in the Datasets section and see how it is added to the table:



I'm going to add two more, Glass and Iris (all three add-ons come with Weka):



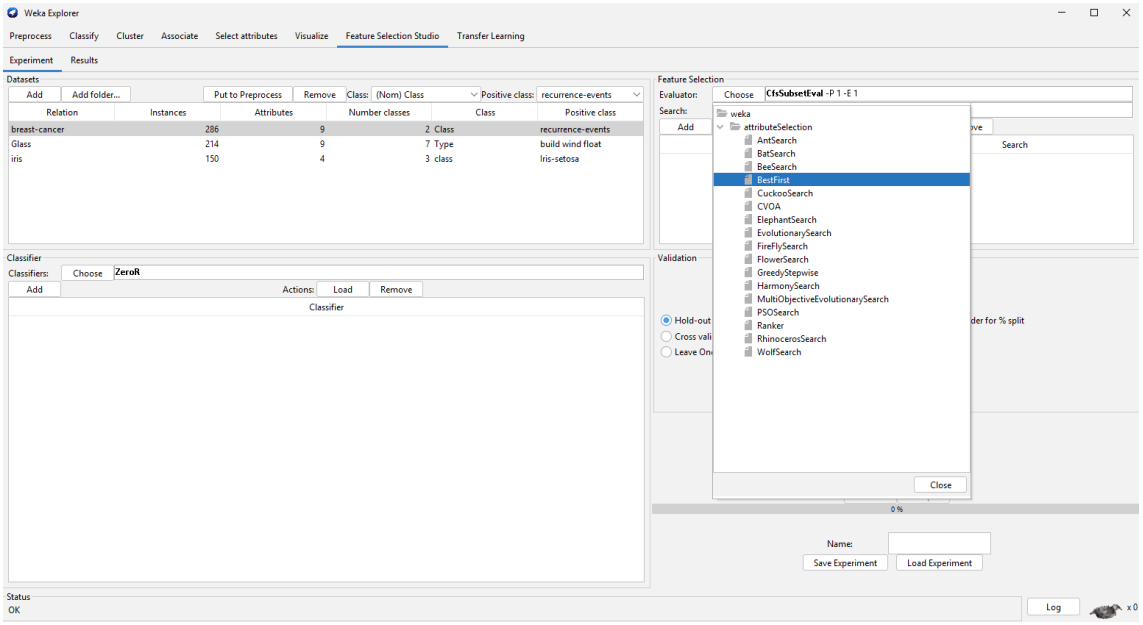
As we can see, all the added datasets appear along with their data for each line, so we are going to modify the Positive class of breast-cancer, for this we select the row and in the Positive class comboBox we select recurrence-events:



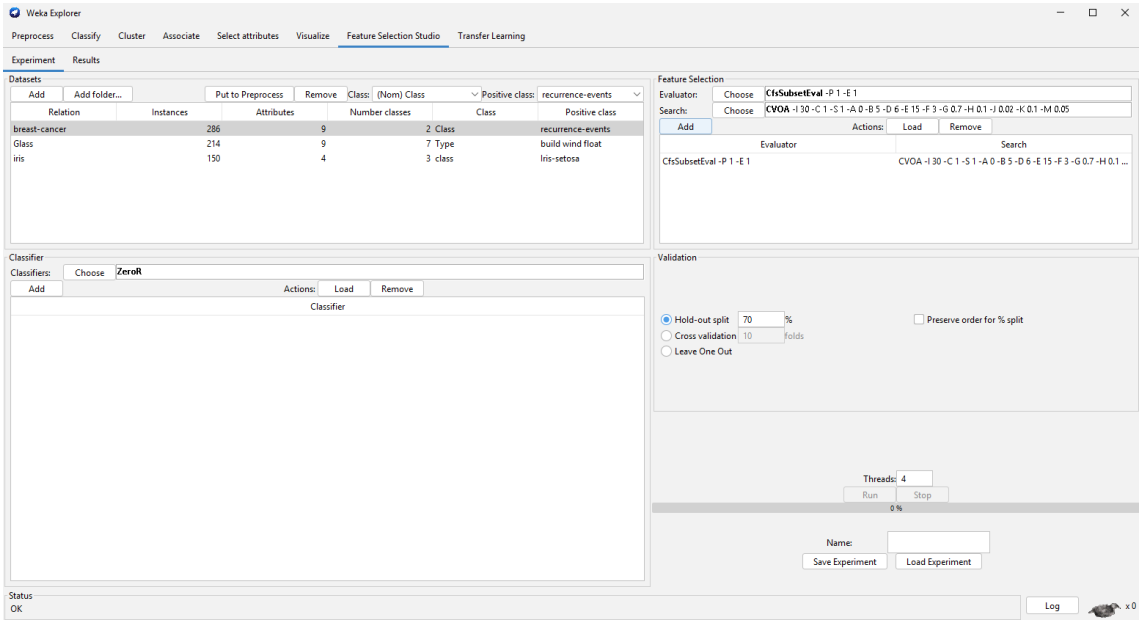
Once selected, we can see that its value is set in the Positive class column.

STEP 2: Search and evaluation algorithms

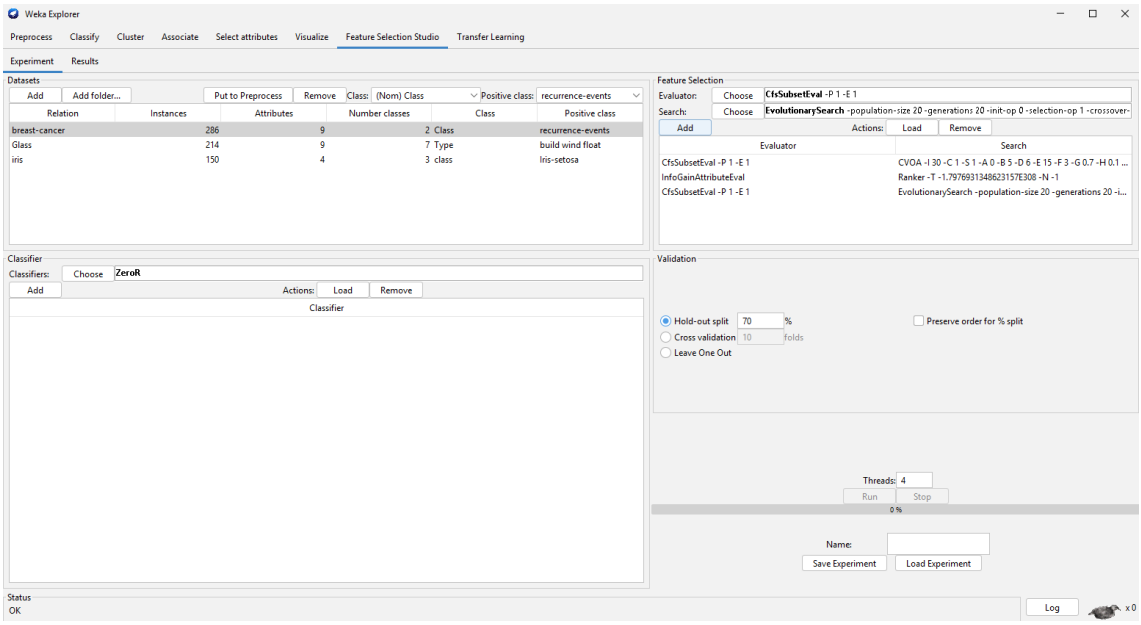
We will do the same as before, we will show how to add one, but we will add 3. To do this, we give the Choose buttons of Evaluator and Search respectively. We are going to add CVOA (which is the one we have integrated and adapted to Weka). We click on Choose in the Search section:



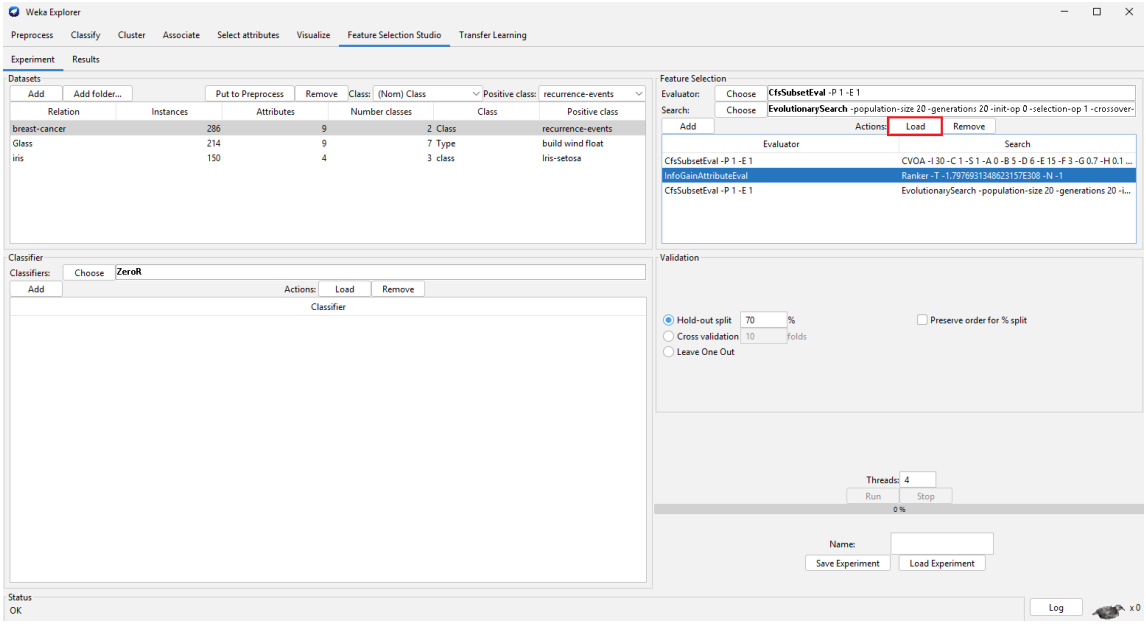
In that menu we choose CVOA, and since we want that evaluator, we click on Add:



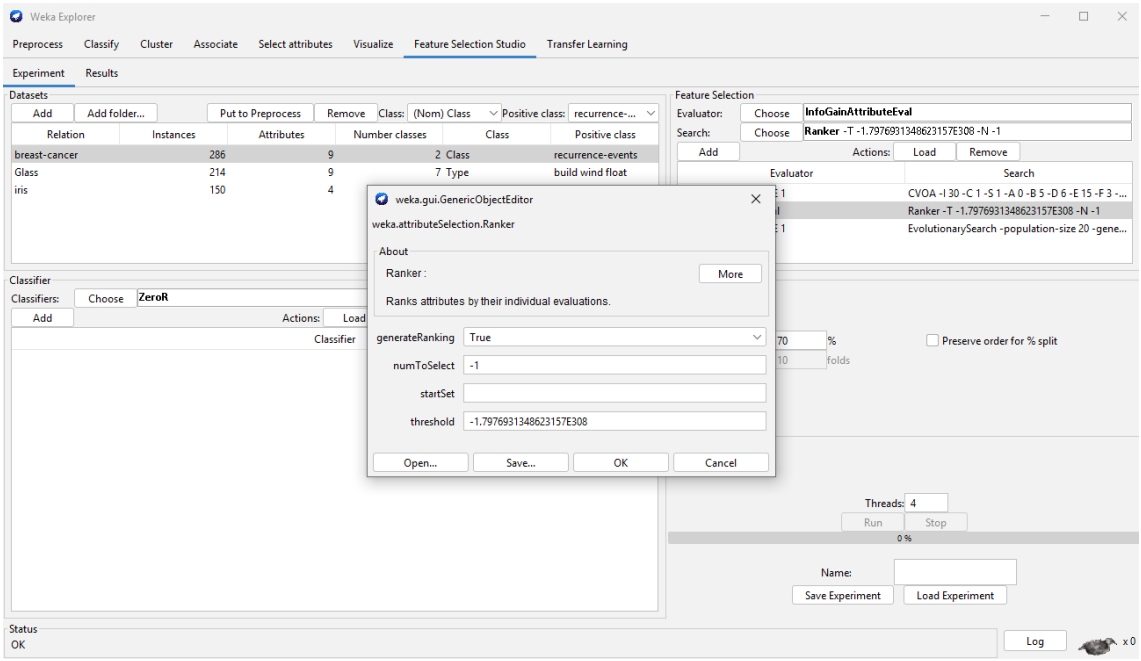
We will also add Ranker and EvolutionarySearch:



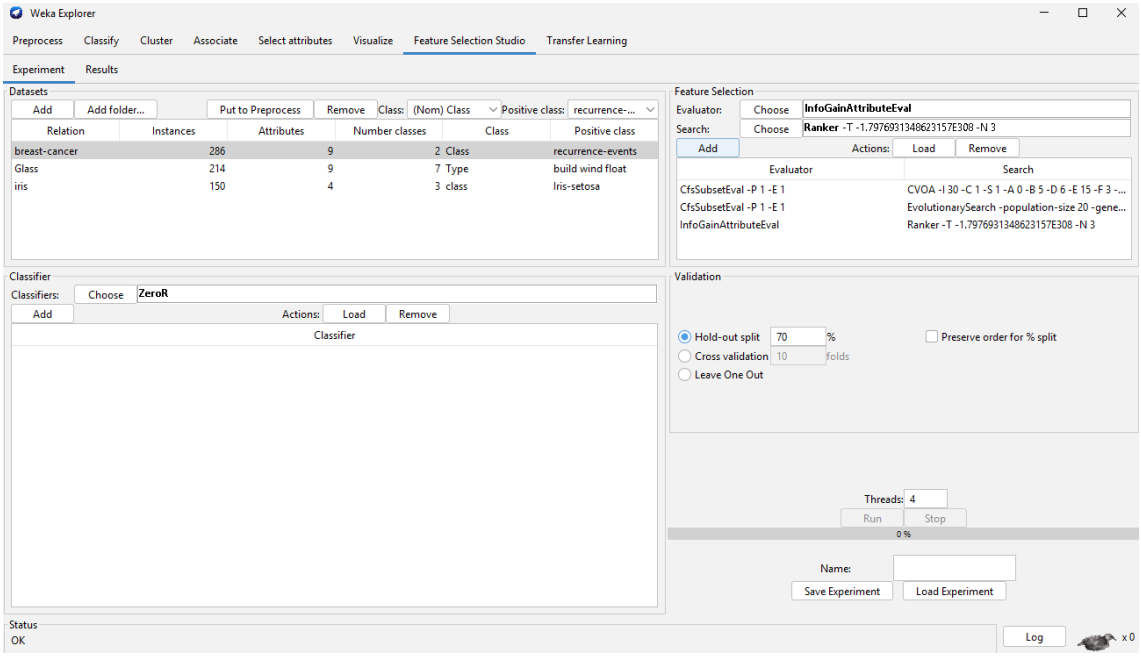
As can be seen, the last element added is loaded in the textFields, we are going to load the Ranker together with its evaluator and modify its options (the same process is followed to load any algorithm or dataset). First we select its row and give Load:



As we can see, these algorithms have been loaded in their respective textFields. Now to modify its options, click on the textField in which the Ranker appears and a menu will appear with its options:



In the numToSelect option we delete the -1 and put 3, and we click Ok. To add this modified algorithm, we delete the previous one by clicking on Remove, since the row is already selected, and then we click on Add again:

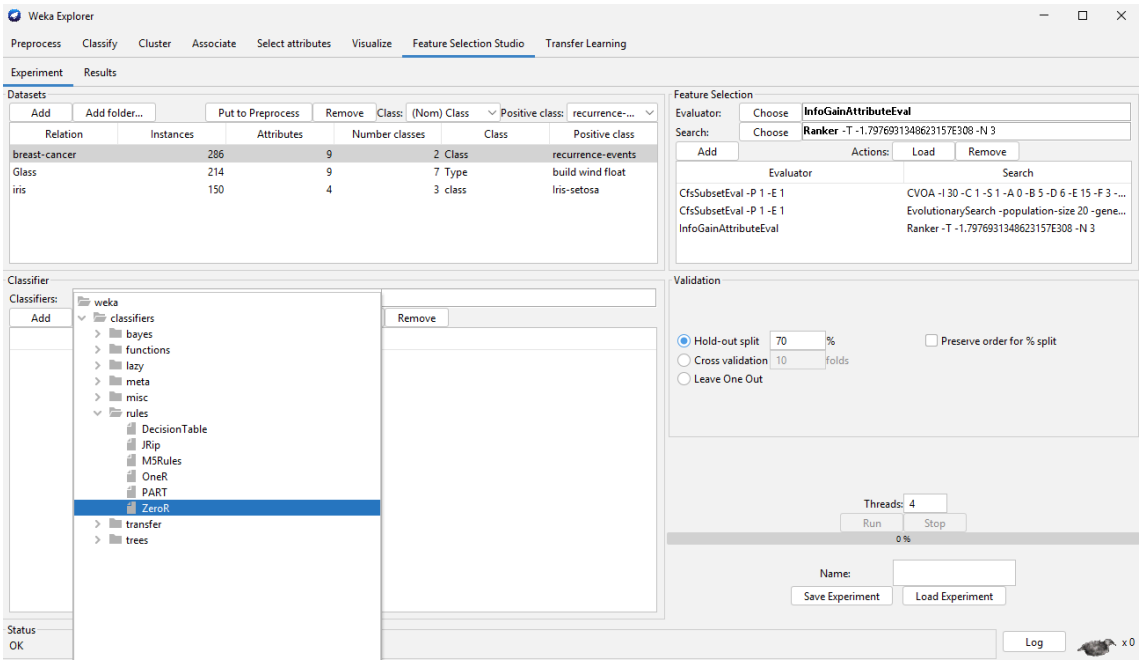


When an algorithm is loaded in your textField before adding it, we can modify its options following the process described above without the need to delete or select it (since it would not be added yet) and add it directly. We have followed this process to show how an algorithm would be loaded, modified and deleted, as for the rest (evaluation and classification/regression algorithm would be the same).

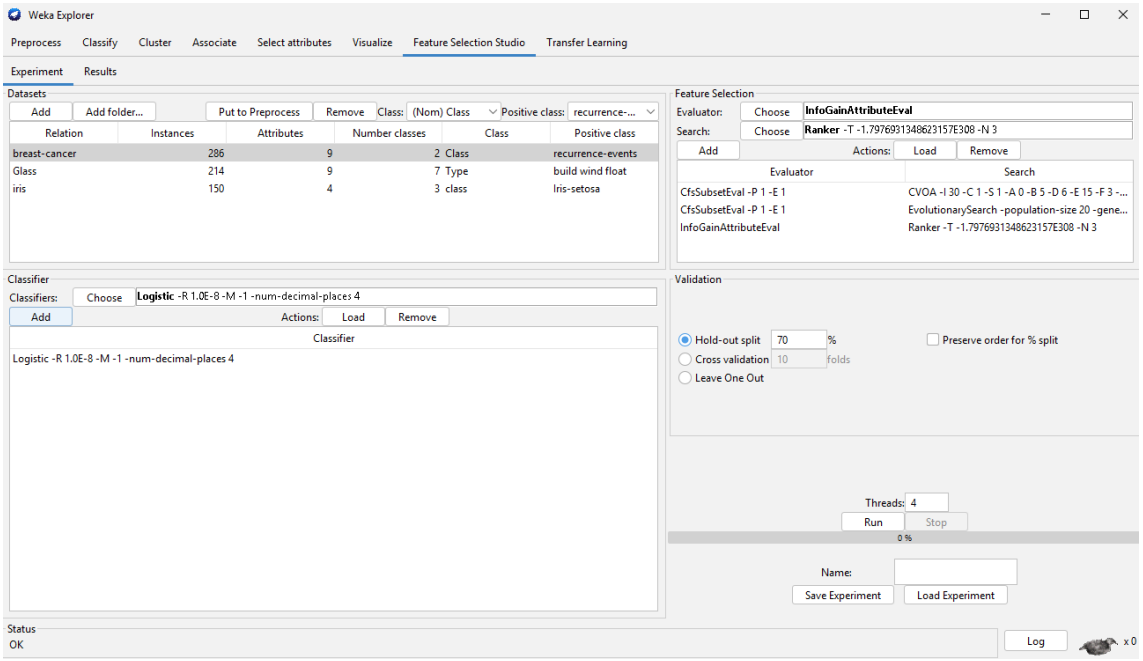
STEP 3: Classification/Regression Algorithm

We are going to show the process to add one, although we will add two to this experimentation.

We click on the Choose button in the Classifier section, a menu will appear where we will go to functions and click on Logistic:



Now to add it, click Add:



I will also add the J48 classifier, which is in the Trees section.

STEP 4: Validation method

We leave the validation method and its default settings (hold-out Split and 70%). If we want to change the validation method, we simply choose it by clicking on its radio button. And if we want to change its options, we modify its value in its respective textField.

STEP 5: Execution

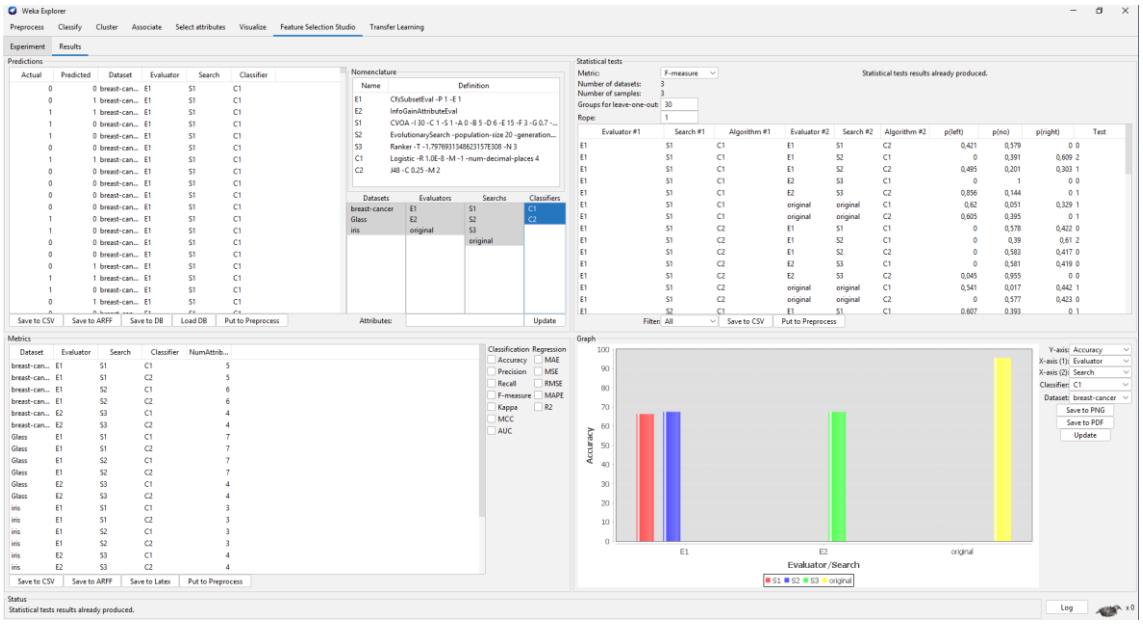
On this occasion, we will also leave the default number of threads (4) that we want to use so that the experimentation is carried out in parallel. If we want to modify this number, we simply modify its value in the corresponding textField.

Now to run the experimentation, we click on Run. The current execution percentage will be shown in the progress bar, when it reaches 100% all its results will be shown in the Results tab.

Just when reaching 100%, a warning is displayed that, if you have chosen the CV or LOO validation method, the number of attributes that will appear is the one applied directly to the dataset and not that of a bag.

3.2 Results

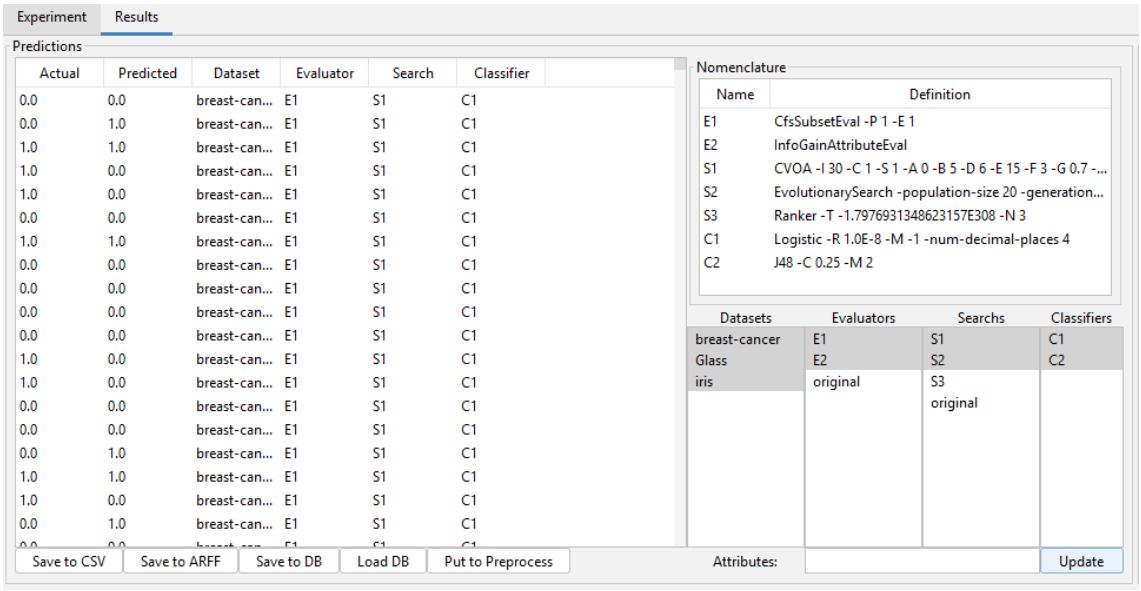
Once the experimentation is executed, in the Results tab we will have something like this:



STEP 6: Predictions

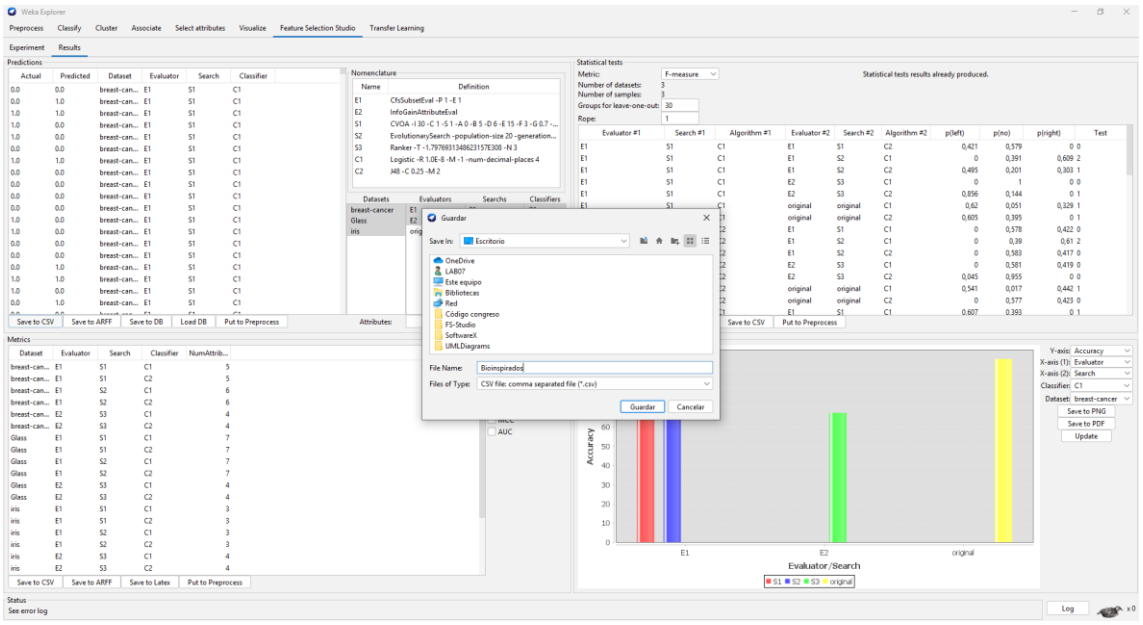
Let's make the case that we want only bio-inspired algorithms (ie CVOA and EvolutionarySearch) to be displayed in that table. To do this, in the Evaluators lists we select the two evaluators (E1, E2, we could also leave it as it is since they go together with their search algorithms, and the filtering would do the same, but it is more correct to do it this way) and in Searchs CVOA and EvolutionarySearch (S1, S2).

Now for these changes to be applied to the table, we click on the Update button:

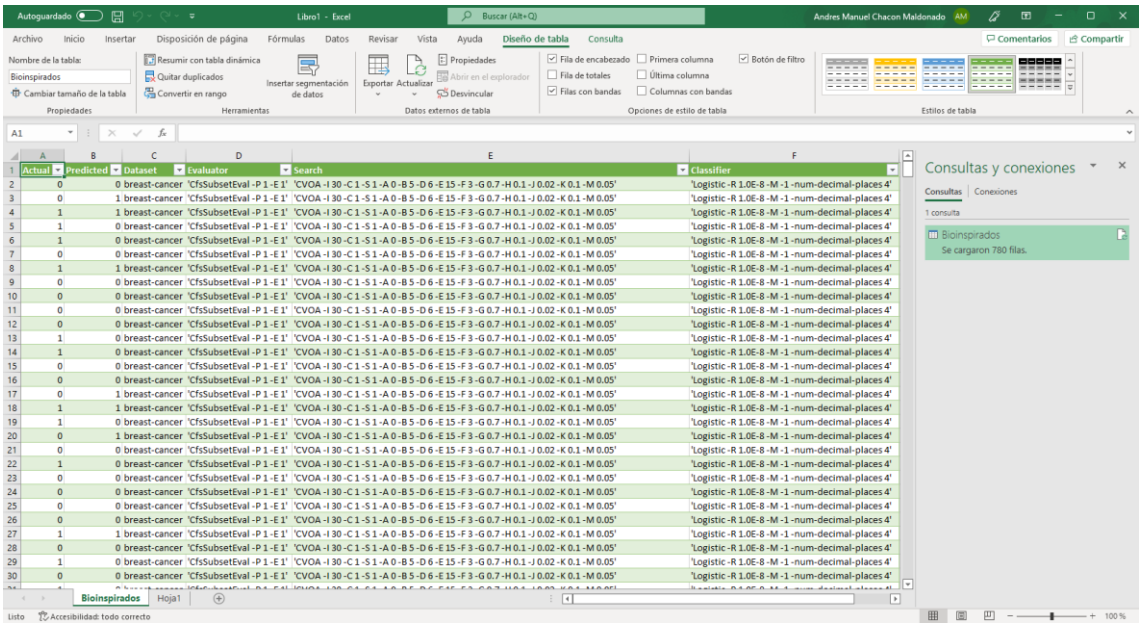


The prediction table has been filtered. As to load said data in the Preprocess tab of Weka you only have to click on the Load dataset button, we are going to show how the data is saved in CSV and in a database (for this second action you must have configured the DatabaseUtils.props file as explained in point 4 DATABASEUTILS CONFIGURATION of this document).

To save the data in CSV, click on the Save to CSV button in said section, appearing the menu where we will choose the name of the file and its location:

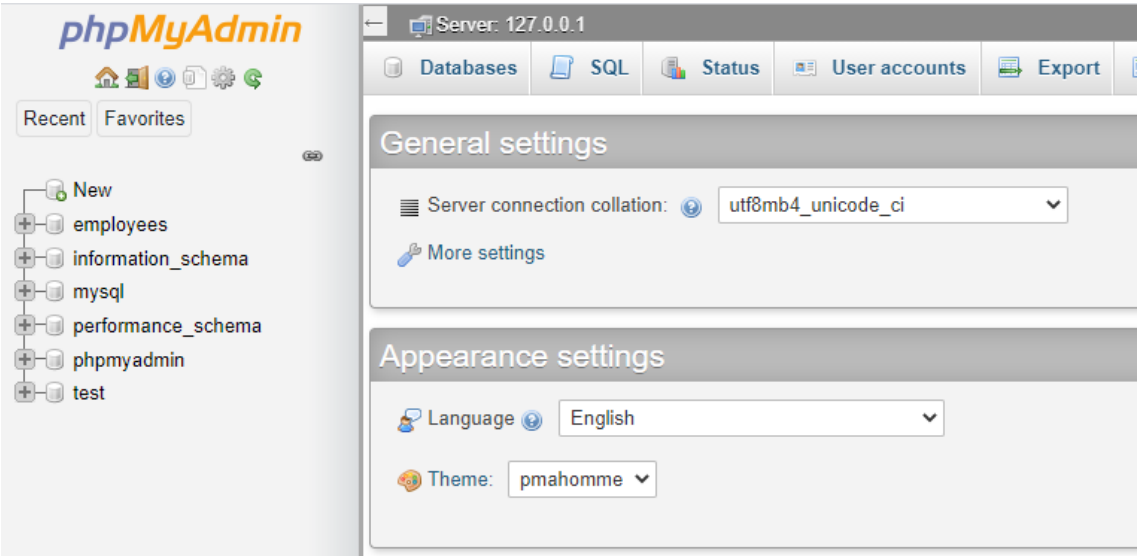


And we click Save. To check that it has been saved correctly, we go to that folder and open the CSV:

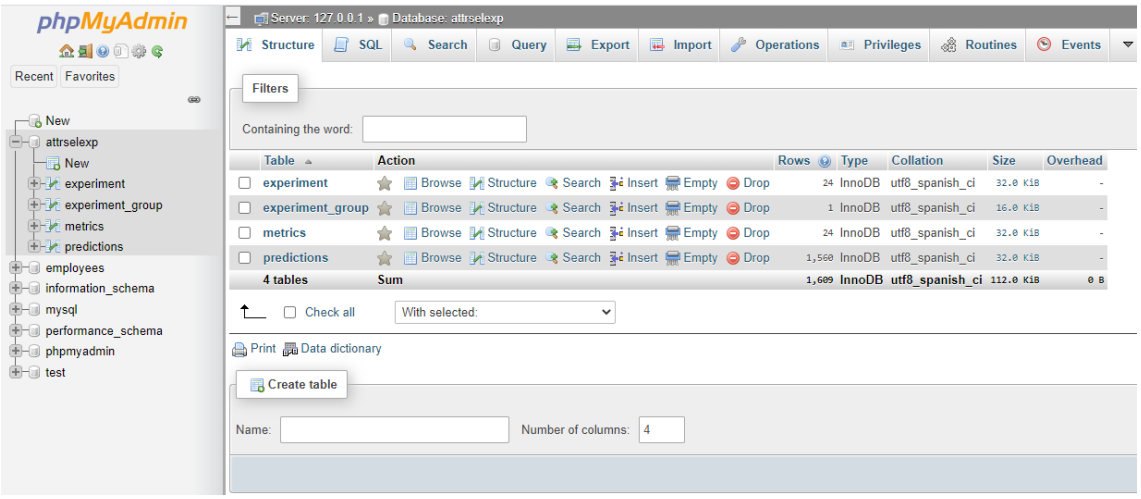


Now we will save the data in a database, for this and once the DatabaseUtils.props file is configured, we click on Save to DB (if the database does not exist or there is no table, they will be created automatically from sqls scripts). This action saves both the predictions and the metrics (in the latter, it saves all the metrics whether they have been chosen or not).

First and to show that it actually creates the database if it does not exist, with the xampp database manager and through PhpMyAdmin we are going to see the databases that we currently have:



Now we click on the Save to DB button, and we see that the database has been created with the saved data:

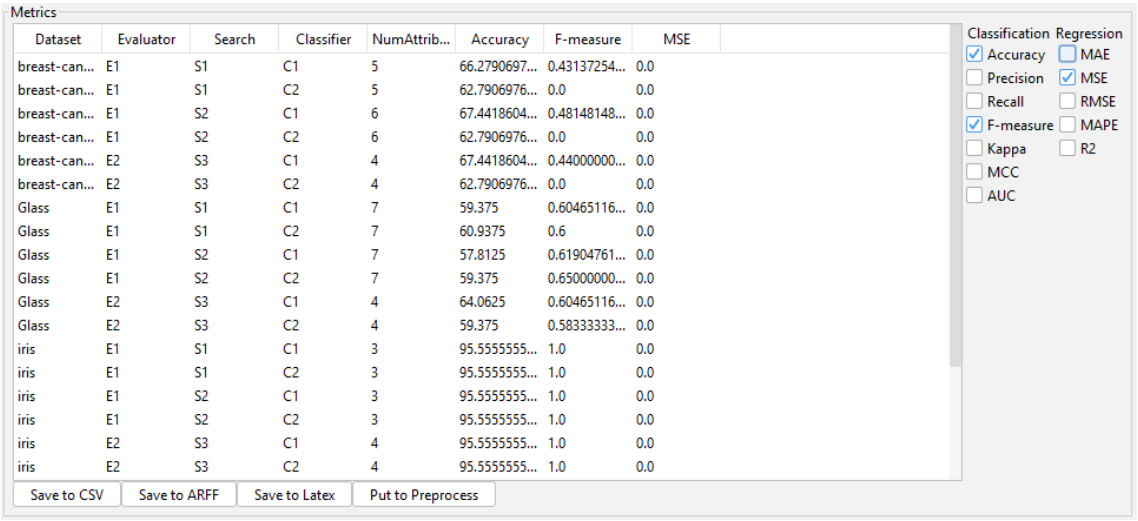


Later we will load the currently saved data.

STEP 7: Metrics

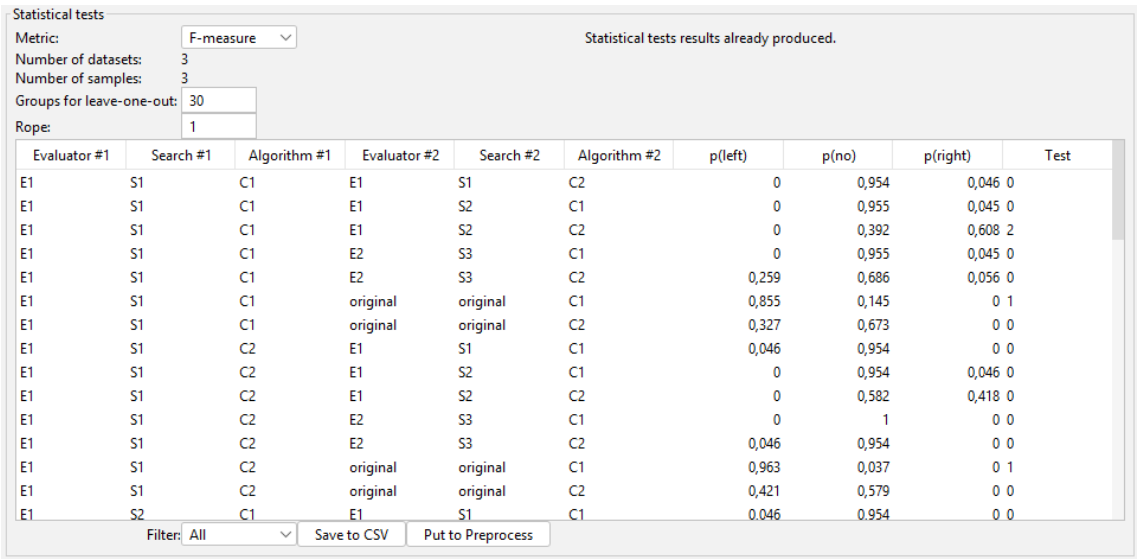
To save the data in CSV or ARFF, we must follow the same steps carried out previously to save the predictions in CSV, and to load them in the Preprocess tab, since the Load dataset button would also be pressed, but this time from this section. Therefore, we are going to show how to add metrics to the table.

This process is as easy as clicking on the checkBoxes of the metrics that we want to add. Since these are classification datasets, I will add the Accuracy and F-measure metrics. I will also add a regression metric to see that if the datasets are of the opposite type, that column shows all its values to zero:



STEP 8: Statistical Tests

By default, the statistical tests are calculated with the F-Measure metric for classification (and MAE for regression), as soon as this metric changes in the comboBox, the statistical tests are recalculated. In our current example, the statistical tests part would look like this:



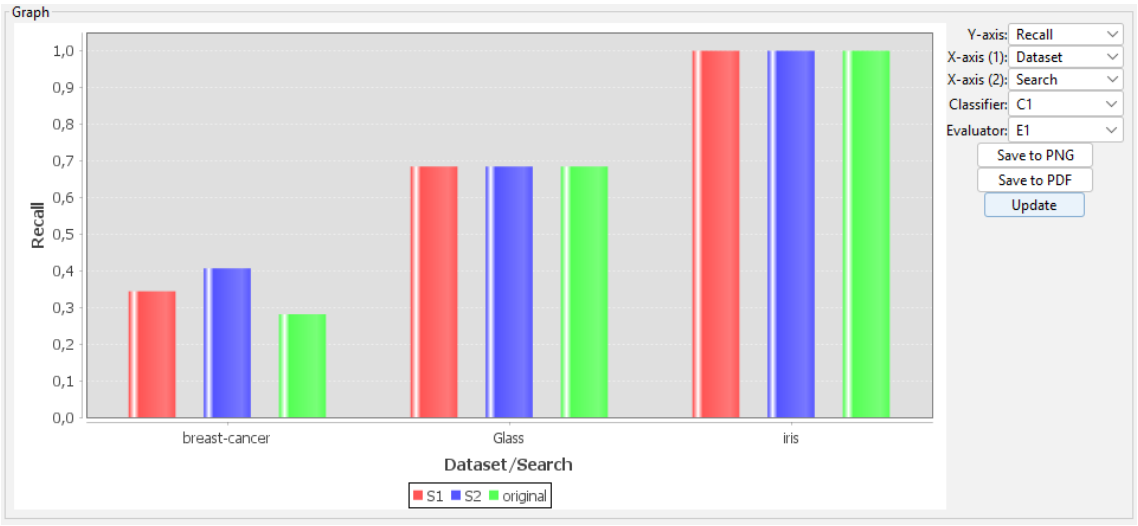
In order for the statistical tests to be calculated, the user must have **python, numpy (pip install numpy)** and the package used to perform these tests **baycomp (pip install baycomp,** https://github.com/janezd/baycomp**)** installed on the computer running Weka.

STEP 9: Charts

By default, a graph is generated with the Accuracy or MAE metric (depending on whether the experimentation carried out is on regression or classification datasets), having the evaluator as the first value of the X axis and the search algorithm as the second. We are going to generate a new

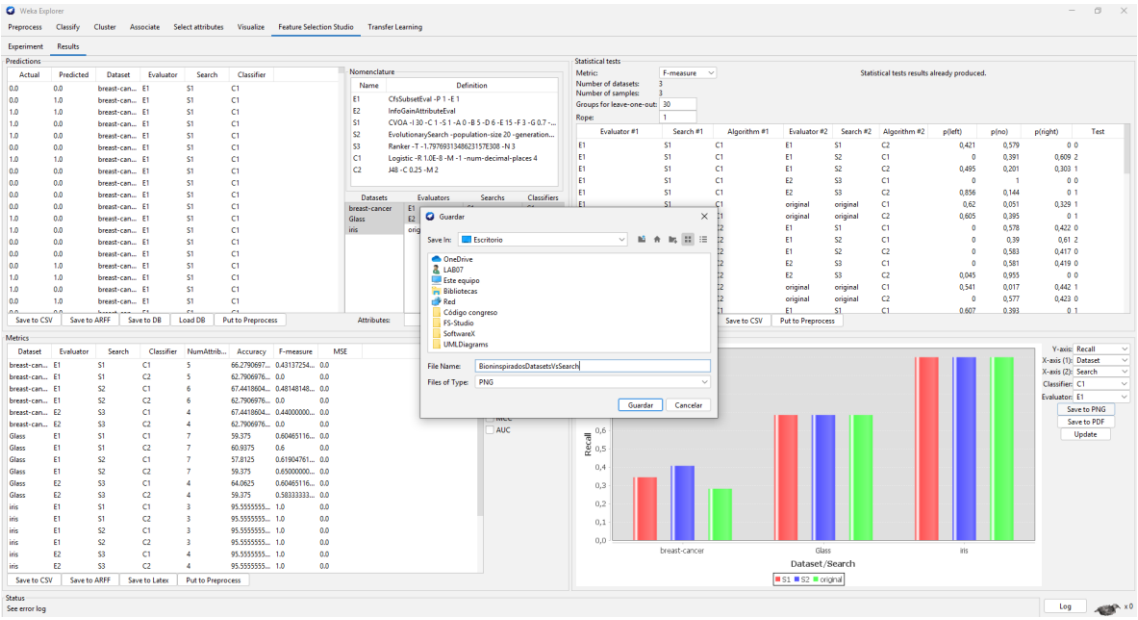
graph with other values and export it in PNG for example (the process for PDF would be exactly the same).

To show that the metrics inserted in the metrics table are independent of whether or not they appear in the graph (that is, you can have some metrics added to the table, and here select any other), we are going to choose the Recall metric, the value on the X 1 axis we will put Dataset and on the X 2 Search axis, on the Classifier we will put C1 and on the Evaluator we will put E1. To do this, we simply click on their respective checkBoxes where the possible values will be displayed. Once this is done, to generate the new graph we click on the Update button:

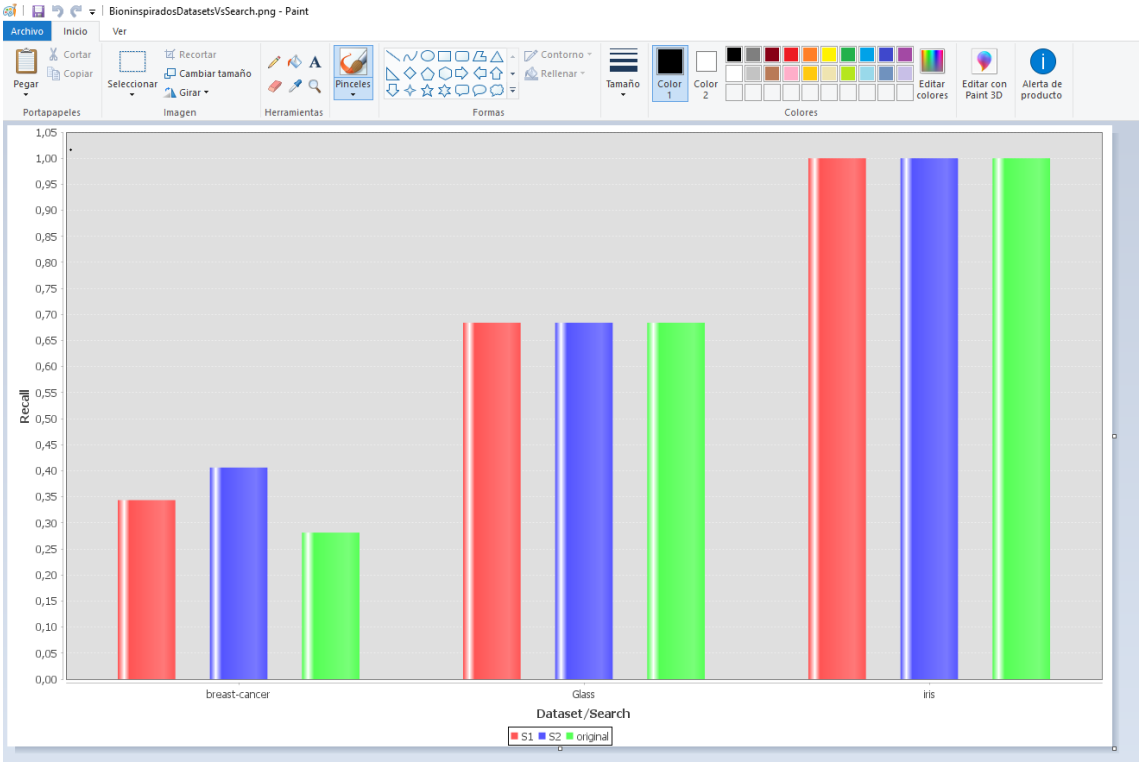


In this case, the search algorithms used can be easily identified by their colors.

To export the graph in PNG format, click the Save to PNG button. A dialog will appear where we will choose where to save it, and its name:



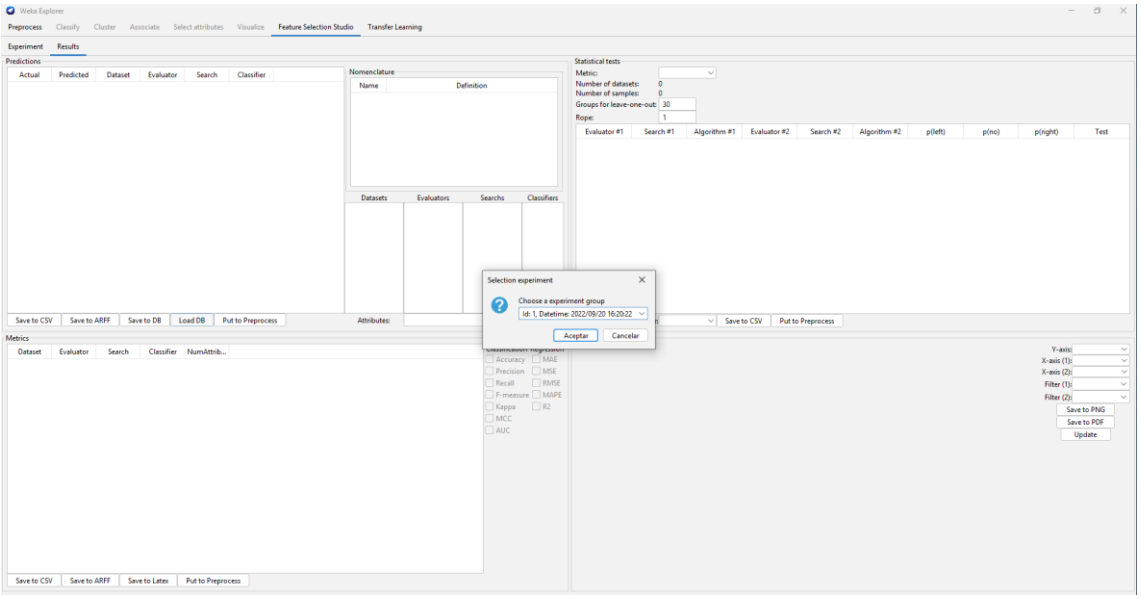
And we give it to save. To check that it has been saved correctly, we go to that folder and open the image:



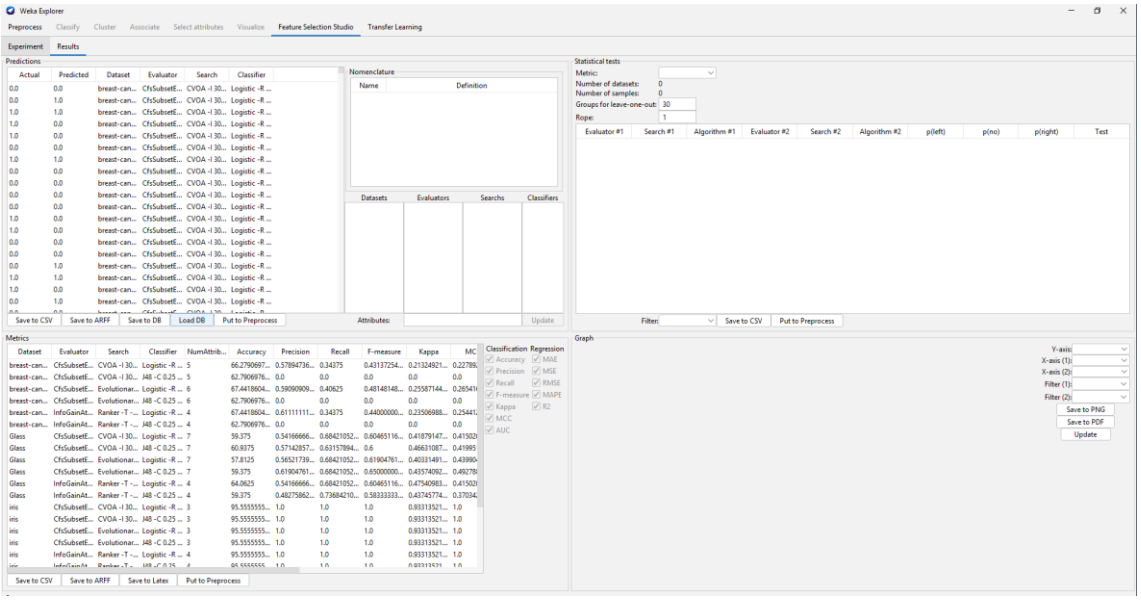
EXTRA STEP: Load data from the database

Data can be loaded from the database without having to run an experiment. That is why we will use this path to better see the differences that are caused in the interface (they can also be loaded after executing an experiment and obtaining the results without any problem).

To do this, we go to the FeatureSelectionStudio tab in Weka, and here to the Results tab. Now we click on Load (we need to have the databaseUtils.prop file correctly configured, see section 4 DATABASEUTILS CONFIGURATION of this document) and the following dialog will appear:



When selecting the checkBox that appears, all the saved experiments will be displayed, represented by their Id and the date and time in which they were saved. We select the one that appears to us since it is the only one we have and we click OK:



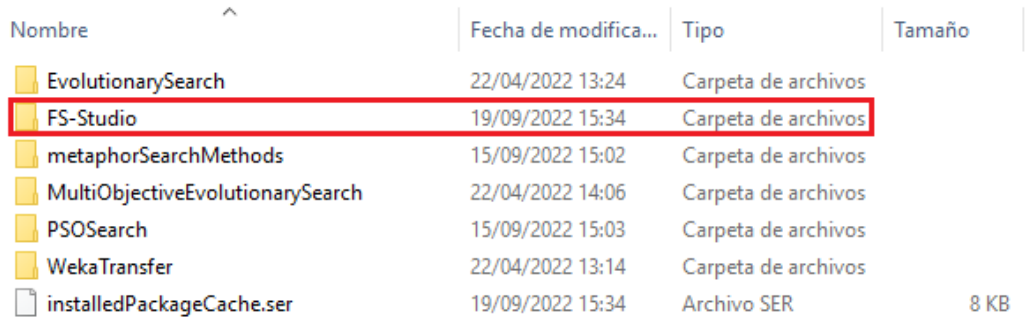
And we would have loaded the data from the database. It can be seen that there are options that are not available, it is because what is saved is data and not an object. The available options are: save to CSV, to ARFF, back to a database and load data again from the database.

4 DATABASEUTILS CONFIGURATION

To be able to use the functions associated with the database, it is necessary to correctly enter the data requested below and in the way that is exposed in the databaseUtils.props file, in addition to having the server belonging to the url that we will introduce more ahead. The steps to follow to carry out the correct configuration of the necessary parameters to access the database are:

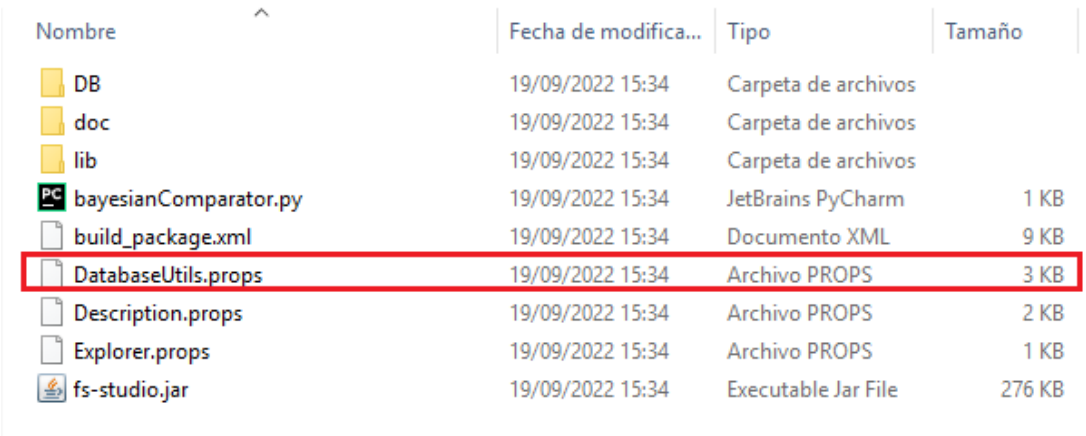
STEP 1

Once we have the FeatureSelectionStudio package installed in Weka, like the rest of the packages that we install, it will be unzipped to the address C:\Users\Usuario\wekafiles\packages on our computer. So we go to that address and enter the FS-Studio folder:



STEP 2

Once there, we open the DatabaseUtils.props file with any text editor, in our case, we will open it with notepad:



When opening it, we must focus on the commented lines of "#The url to the experiment database", "#Your user" and "#Your password":

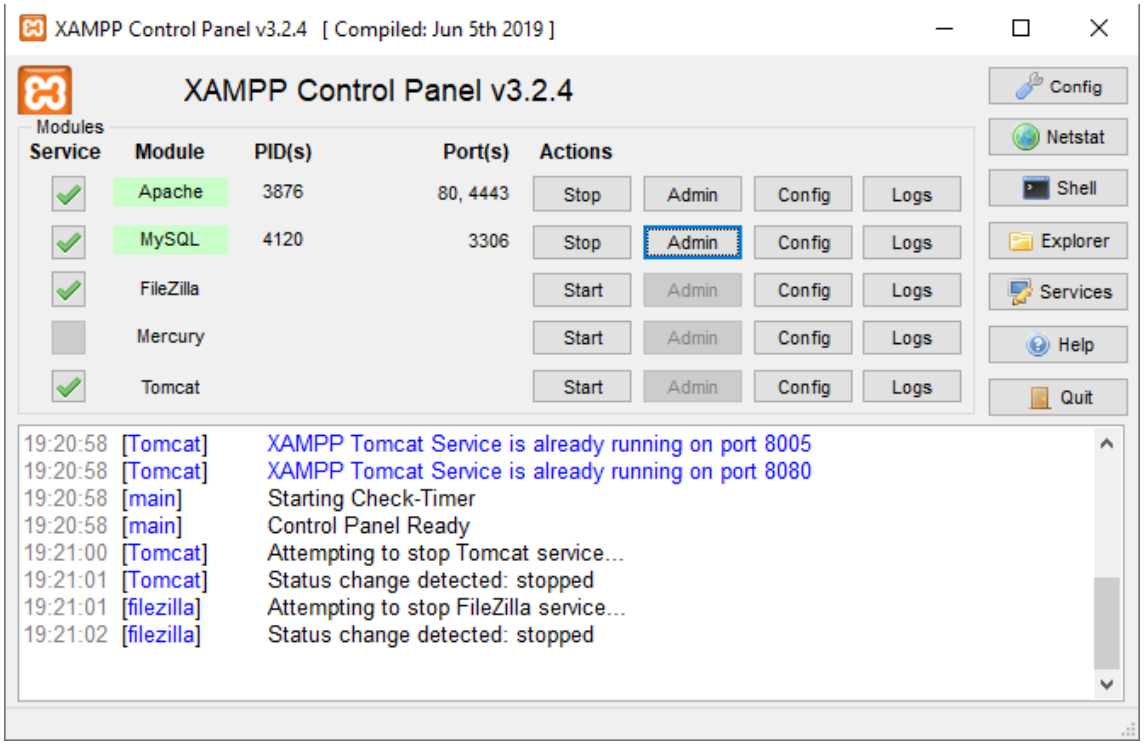
DatabaseUtils.props: Bloc de notas

Archivo Edición Formato Ver Ayuda

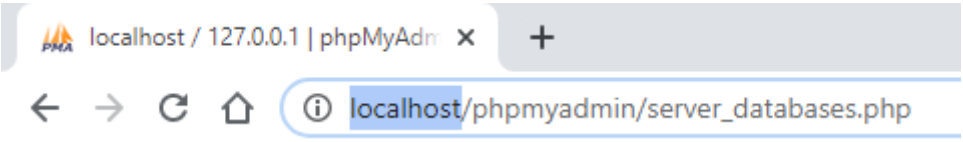
```
# General information on database access can be found here:
# https://waikato.github.io/weka-wiki/databases/
#
# Version: $Revision: 15257 $

# The comma-separated list of jdbc drivers to use
#jdbcDriver=RmiJdbc.RJDriver,jdbc.idbDriver
#jdbcDriver=jdbc.idbDriver
#jdbcDriver=RmiJdbc.RJDriver,jdbc.idbDriver,org.gjt.mm.mysql.Driver,com.mckoi.JDBCDriver,org.hsqldb.jdbcDriver
jdbcDriver=com.mysql.jdbc.Driver

# The url to the experiment database
#jdbcURL=jdbc:rmi://expserver/jdbc:idb=experiments.prp
#jdbcURL=jdbc:idb=experiments.prp
#jdbcURL=jdbc:mysql://localhost:3306/

#Your user
#user=root

#Your password
#password=

# the method that is used to retrieve values from the db
# (java datatype + RecordSet.<method>)
# string, getString() = 0;    --> nominal
# boolean, getBoolean() = 1;  --> nominal
```

We uncomment the last line of each of them by deleting the "#" symbol. In case of having such data (url = localhost:3306/, user = root and empty password), we save and it would be ready.

In case of having other data, we delete the ones that come by default and put ours. To know the url of our server, if we have a database manager like Xampp and make sure it is active, in the MySql section we give Admin:



Now it will open in our browser that we have the PhpMyAdmin manager set by default, we take its url (only what appears before the first "/") and we put it in its corresponding place in the file:




```
# The url to the experiment database
#jdbcURL=jdbc:rmi://expserver/jdbc:idb=experiments.prp
#jdbcURL=jdbc:idb=experiments.prp
#jdbcURL=jdbc:mysql://localhost
```

Now just after we put ":" and the port which previously appeared in Xampp (in our case 3306), followed by "/":

```
# The url to the experiment database
#jdbcURL=jdbc:rmi://expserver/jdbc:idb=experiments.prp
#jdbcURL=jdbc:idb=experiments.prp
#jdbcURL=jdbc:mysql://localhost:3306/
```

And that's it, we can now access the functions that interact with the database.