

## table of contents

Using MSClassifier.....	2
Overwiev.....	2
Data input format.....	3
Creating a new profile.....	3
Creating a Score plot of a profile.....	5
Creating a loading plot of a profile.....	8
carrying out a cross validation.....	8
Classify new items.....	11
live classification.....	13
Profile Info Dialog.....	14
Developing MSClassifier.....	15
Import the Project into NetBeans.....	15
What Libraries are used and how to import them.....	15
Where is the Javadoc.....	16
How to build the Jar file.....	16

# Using MSClassifier

## Overview

To start the program simply double-click on the .jar file or under Linux right-click on it and choose *open with Oracle Java 8*. The graphical user interface (GUI) the the program consist of a single window containing six tabs (Illustration 1).

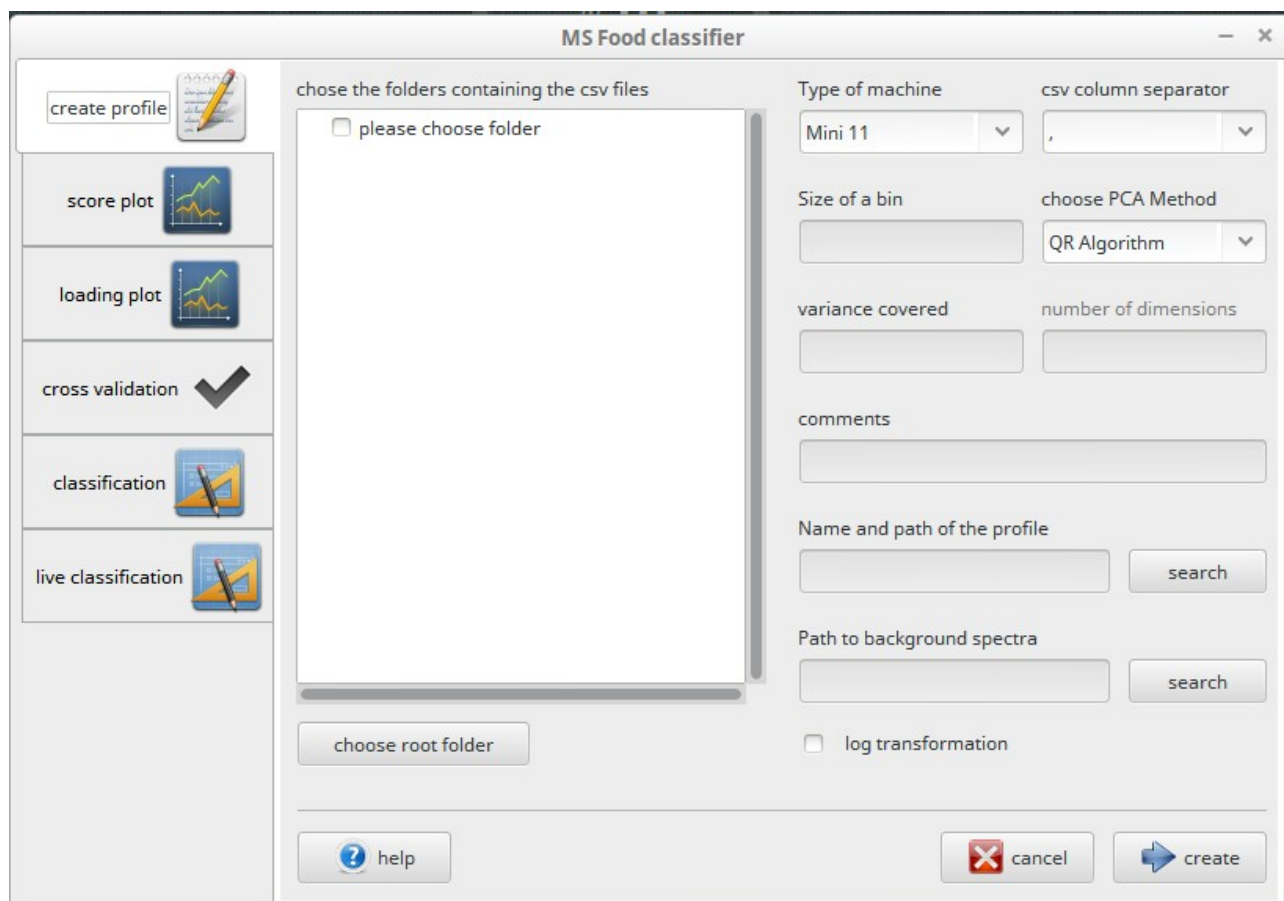


Abbildung 1: The GUI presented at startup.

**create profile tab:** In this tab you can create a new profile for your measured food items to later use to classify unknown items.

**score plot tab:** In this tab you can create a 2-dimensional or 3-dimensional score plot of a profile.

**loading plot tab:** n this tab you can create a 2-dimensional or 3-dimensional loading plot of a profile.

**cross validation tab:** In this tab you can carry out a leave-10%-out cross validation using choosen parameters to find the right parameters to create a profile.

**classification tab:** In this tab you can load a profile and use it to classify new items.

**live classification tab:** In this tab you can load a profile and use it to classify new items during a MS experiment.

## Data input format

The program reads csv files. Each csv file must only contain one spectrum. The m/z ratios are read from the first column and the intensity values from the second. Additional column after these two will be ignored. The header can contain any information but the last line of the header must either read exactly „M/Z,Voltage“ or must start with the word „Masse“ with a capital M. The first one is the header of the Mini 11 output, the second one the header of the RawBrowser used to export csv files from the exactive RAW files. The ability to read other headers has not yet been implemented.

## Creating a new profile

In the Tab *create Profile* you can create a profile of several food items that have been measured via MS. The data is transformed into PC space via PCA. These profiles can be used later to classify unknown spectra of food items. If you start working with the program you will most likely start from here as the first step in classifying food items is to create a profile using reference data. Illustration 2 shows the tab.

Abbildung 2: The tab to create a new profile

### Elements:

**folder containing csv files:** This is where you choose what items to put into the profile. Choose a root folder. In the subfolders should lie the spectra to each group of food. If you have for ex. cow milk, goat milk and soy milk as food items then there should only be three

subfolders in your chosen root folder: *cow milk*, *goat milk*, *soy milk* (or whatever you would like to name them) each group is named by the subfolder it lies in. You can also have multiple layers of subfolders for ex. *Milk* → *cow* etc. then the group name will be *milk\_cow* for this group as the names of the subfolders for each group will be concatenated using an underscore. The panel above the *choose root folder* button displays a tree view after selection of a root folder. In the tree view you can see all the subfolders of that root folder. By clicking the checkboxes in front of the desired subfolders you can choose what food items should go into the profile. Warning: spaces in the names of the folders can cause problems on some systems. It is recommended that you don't use spaces in folder names.

**type of machine:** The MS device used for measuring. You can currently choose between *Mini 11* and *Exactive*

**csv column separator:** The character that separates the column in the csv files containing the spectra. Each spectrum must be in its own csv file with the mz-value in the first column and the intensity value in the second column. Additional columns will be ignored.

**size of bin:** The desired size of the m/z bins.

**PCA Algorithm:** The algorithm used to find the eigenvectors which are used to transform our data matrix (the mz-bins of all spectra) into PC space. Currently there are two algorithms available: the NIPALS algorithm and the QR algorithm.

**variance covered:** This is the amount of variance of the original dataset that should be covered by PCA. This must be a number between 0 and 1 where 0 is no variance at all and 1 is all the variance from the original dataset (all dimensions (mz-bins) will get transformed to principal components). The amount of covered variance is only available for QR algorithm. Although you can choose to keep all of the variance the program will only transform 60 principal components at maximum since a bigger number has shown not to be more effective and can lead to errors when using the mahalanobis distance measure.

**number of dimensions:** When using the NIPALS algorithm for transformation into PC space you cannot choose how much variance should be covered. You can only choose the number of dimensions from the original data matrix to transform into PC space. If you are not sure to how much variance this number of dimensions amounts you can create a test profile using the QR algorithm first and then open the profile info in one of the plot tabs or classification tabs (load a profile and click on the *Info* button to the right of the profile textfield) and look up the number of dimensions in that profile. Doing that you can get a rough glimpse what number of dimensions makes up what percentage of the total variance.

**Comments:** In this text field you can write anything you want. This can be helpful to store information that does not fit into the profiles filename. If you need to check this information later on you can access it via the *Profile info dialog* from the *score plot*, *loading plot*, *classification* and *live classification* tabs.

**name and path to profile:** Where and under what name should the profile be saved. The program cannot produce additional folders. So you should always save the profile to an existing folder.

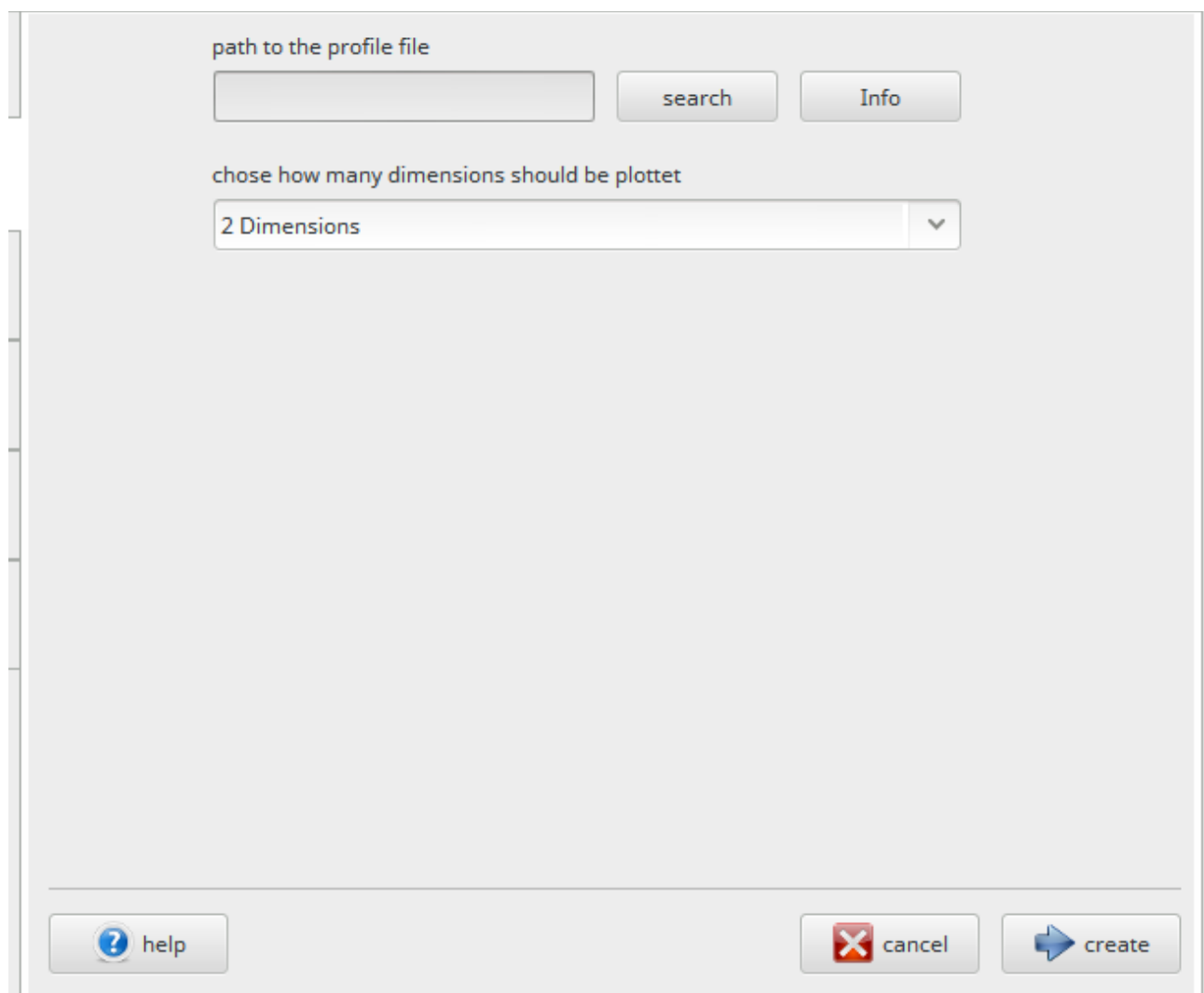
**Path to background spectra:** If you want to subtract background data from all of your spectra then you can give the path to a folder with csv files containing background data. The csv files must be formatted the same way as the other csv files (two columns (mz, intensity) and same separator). The program calculates the means of every mz-bin of the background over all spectra in the folder and subtracts them from the mz-bins of the original data. Warning: spaces in the names of the folders can cause problems on some systems. It is recommended that you don't use spaces in folder names.

**log transformation:** Should the data input be log transformed.

Once you filled out all the information you can click on the *create* button to create the profile. When the program has finished calculating a dialog message will pop up and confirm that the profile has been created.

## Creating a Score plot of a profile

The information in every profile can be displayed in a score plot if the pca transformed data in the profile has at least two dimensions. The tab *score Plot* lets you create a score plot of a profile.



The image shows a software interface for creating a score plot. It features a light gray background with a white border. At the top, the text "path to the profile file" is displayed in a small, dark font. Below this is a white text input field. To the right of the input field are two buttons: "search" and "Info". Below the input field, the text "chose how many dimensions should be plottet" is displayed. Below this text is a dropdown menu showing "2 Dimensions" with a small downward arrow on the right. At the bottom of the interface, there are three buttons: "help" (with a question mark icon), "cancel" (with a red X icon), and "create" (with a blue right-pointing arrow icon).

Abbildung 3: The design of the score plot tab and the loading plot tab.

## **Elements:**

**Path to the profile file:** This is the complete path including name to the profile file including the transformed data to plot.

**search:** When you press the search button on the right side of the textarea a search dialog opens. If you have found your desired file you can either double click on it or mark it and click the *ok* button.

**Info:** Once you selected a profile file you can load basic info about the profile and display it. The info includes things as number of dimensions, MS device used, comments etc.

**dimensions:** The dropdown menu for the number of dimensions to be plotted.

Once you filled everything out you can click on the *create* button and the plot will be rendered for you. The plot appears in a separate dialog window shown in Illustration 4. Above the plot there are several menu items with useful functions. In the image the buttons are numbered for easier distinction.

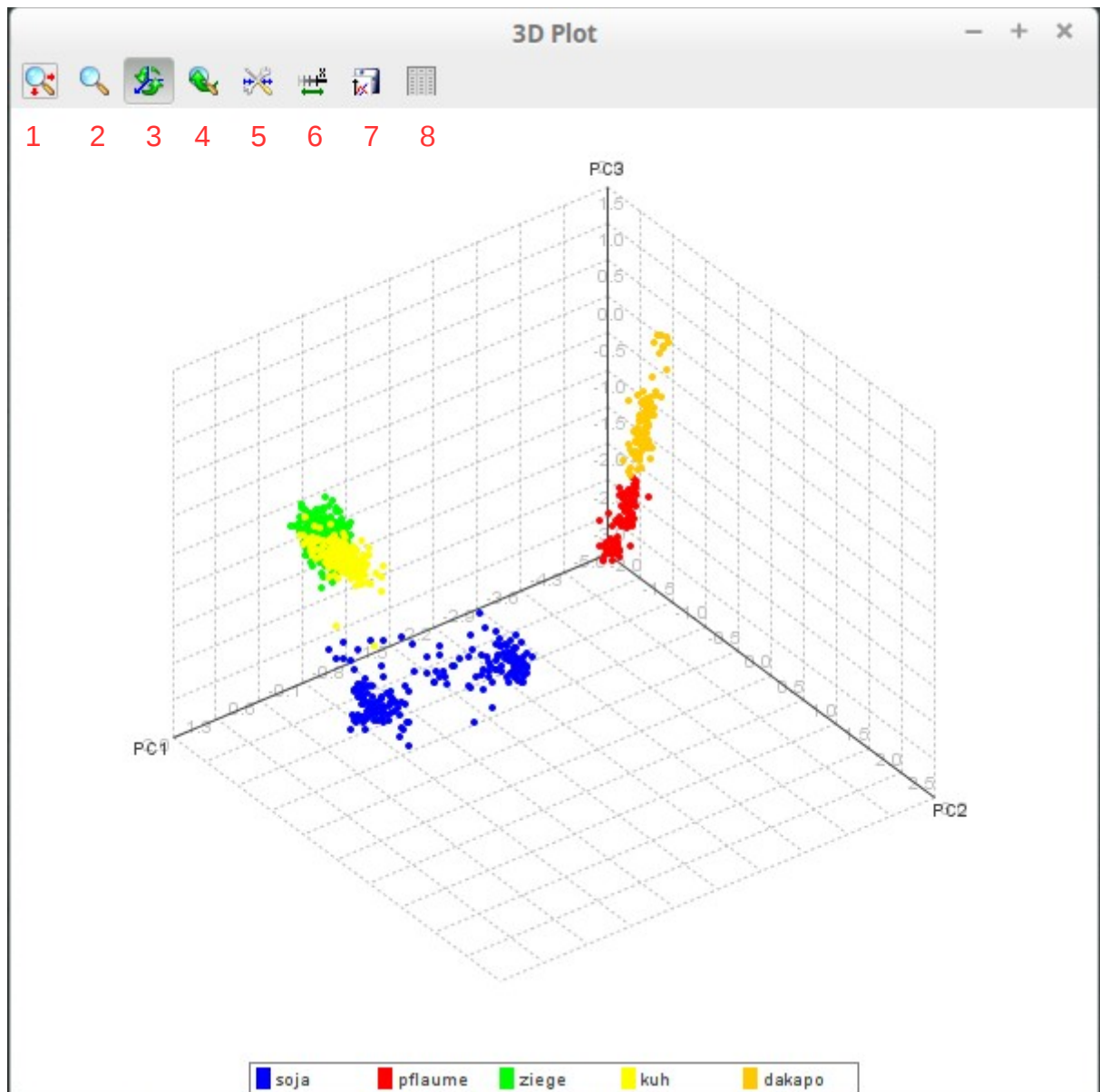


Abbildung 4: The plot dialog window with a score plot showing.

### **Elements:**

**Plot:** The main plot of a profile.

### **Menu:**

1. Move the plot in the plane.
2. Zoom into a marked area. Click this button then click and drag in the plot area until you marked everything you want to zoom bigger.
3. Rotate the plot to view different angles.
4. Reset zoom and axis rotation.

5. Edit the axes. Here you can change the names of the axes, their ranges and even if they should be log scaled or not.
6. Fix bounds.
7. Save the plot to a .png image file. The image dimensions are the same as the plot window, so if you want a bigger image you have to adjust the window size.
8. Table view of the plotted data. You can view the data of each group separately and save them or copy them to the clipboard if you want. You can also adjust the color for a group here.

**Legend:** The legend showing all plotted groups and their respective colors is at the bottom of the plot. If you click on one group in the legend all the points from that group in the plot appear in a dark grey. If you hover over the points its exact values on the axes are shown in a small tooltip.

## Creating a loading plot of a profile

The information in every profile can be displayed in a loading plot. The tab loading *Plot* lets you create a loading plot of a profile. The loading plot tab looks exactly as the score plot tab, also the dialog showing the rendered plot has the same functions with an additional feature to display the original mz-bin for each data point of the loadings. To do that look at the point *Legend* below. Each data point in the plot is a variable across several loadings (principal components). For consultation of how to use the elements in the tab go to the last chapter *Creating a score plot of a profile* as the information can be directly applied to the loading plot tab.

**Legend:** As all points of the loading plot belong to just one group the legend of the loading plot window only shows the group *data points*. By clicking on it all points on the plot appear in a dark grey color. If you hover over the points the original mz-bin a point of the loadings is shown in a small tooltip. This value shows which bin of the original dataset amounts how much to the variance of the pca transformed data.

## carrying out a cross validation

In the Tab *cross validation* you can carry out a leave-10%-out cross validation to test certain profile parameters. This comes in handy to find the right parameters to create a profile for productive use. The Tab looks very similar to the *create profile* tab which makes its use easier.



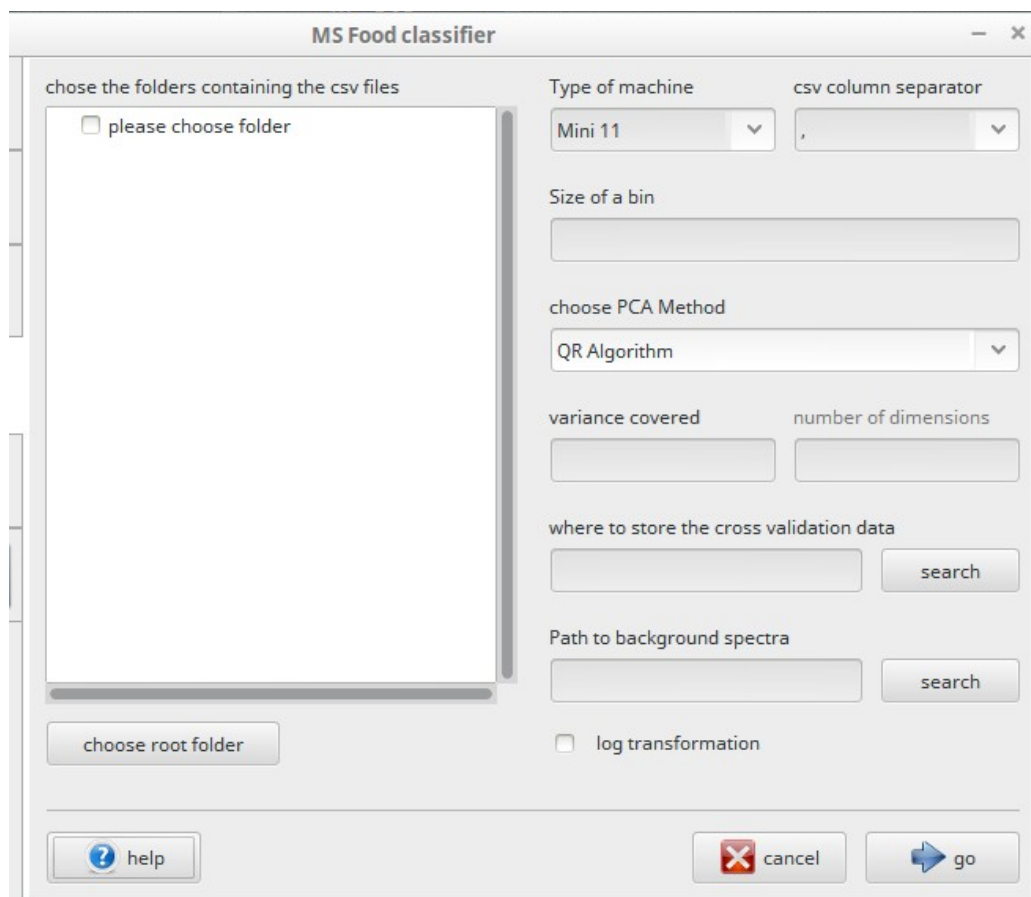


Abbildung 5: cross validation tab

## Elements:

**folder containing csv files:** This is where you choose what items to put into the profiles. Choose a root folder. In the subfolders should lie the spectra to each group of food. If you have for ex. cow milk, goat milk and soy milk as food items then there should only be three subfolders in your chosen root folder: *cow milk*, *goat milk*, *soy milk* (or whatever you would like to name them) each group is named by the subfolder it lies in. You can also have multiple layers of subfolders for ex. *Milk* → *cow* etc. then the group name will be *milk\_cow* for this group as the names of the subfolders for each group will be concatenated using an underscore. The panel above the *choose root folder* button displays a tree view after selection of a root folder. In the tree view you can see all the subfolders of that root folder. By clicking the checkboxes in front of the desired subfolders you can choose what food items should go into the profile. Warning: spaces in the names of the folders can cause problems on some systems. It is recommended that you don't use spaces in folder names.

**type of machine:** The MS device used for measuring. You can currently choose between *Mini 11* and *Exactive*

**csv column separator:** The character that separates the column in the csv files containing the spectra. Each spectrum must be in its own csv file with the *m/z*-value in the first column and the intensity value in the second column. Additional columns will be ignored.

**size of bin:** The desired size of the *m/z* bins.

**PCA Algorithm:** The algorithm used to find the eigenvectors which are used to transform our data matrix (the mz-bins of all spectra) into PC space. Currently there are two algorithms available: the NIPALS algorithm and the QR algorithm.

**variance covered:** This is the amount of variance of the original dataset that should be covered by PCA. This must be a number between 0 and 1 where 0 is no variance at all and 1 is all the variance from the original dataset (all dimensions (mz-bins) will get transformed to principal components). The amount of covered variance is only available for QR algorithm. Although you can choose to keep all of the variance the program will only transform 60 principal components at maximum since a bigger number has shown not to be more effective and can lead to errors when using the mahalanobis distance measure.

**number of dimensions:** When using the NIPALS algorithm for transformation into PC space you cannot choose how much variance should be covered. You can only choose the number of dimensions from the original data matrix to transform into PC space. If you are not sure to how much variance this number of dimensions amounts you can create a test profile using the QR algorithm first and then open the profile info in one of the plot tabs or classification tabs (load a profile and click on the *Info* button to the right of the profile textfield) and look up the number of dimensions in that profile. Doing that you can get a rough glimpse what number of dimensions makes up what percentage of the total variance.

**where to store the cross valid.:** Give a folder where to store all the data that is produced during cross validation. The program cannot produce additional folders. So you should always save the cross validation data to an existing folder.

**Path to background spectra:** If you want to subtract background data from all of your spectra then you can give the path to a folder containing csv files containing background data. The csv files must be formatted the same way as the other csv files (two columns (mz, intensity) and same separator). The program calculates the means of every mz-bin of the background over all spectra in the folder and subtracts them from the mz-bins of the original data. Warning: spaces in the names of the folders can cause problems on some systems. It is recommended that you don't use spaces in folder names.

**log transformation:** Should the data input be log transformed.

Once you filled out all the information you can click on the *go* button to start the cross validation. When finished a dialog message will be displayed showing the percentage and number of correctly assigned samples during cross validation for each of the classification methods (euclidean distance, mahalanobis distance and linear discriminant analysis). The dialog is shown in Illustration 6.

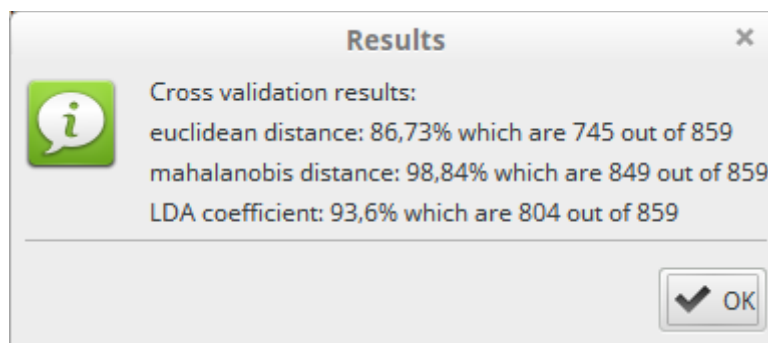


Abbildung 6: cross validation results dialog showing percentage and number of correctly classified samples

## Classify new items

In the *classification* tab you can classify new samples of food using a created profile. As for the profile creation the input files must be of csv format with the mz-value in the first column and the intensity in the second column. Each spectrum must be in its own csv file. The program does not recognize multiple spectrum files. The GUI of the *classification* tab is shown in Illustration 7.

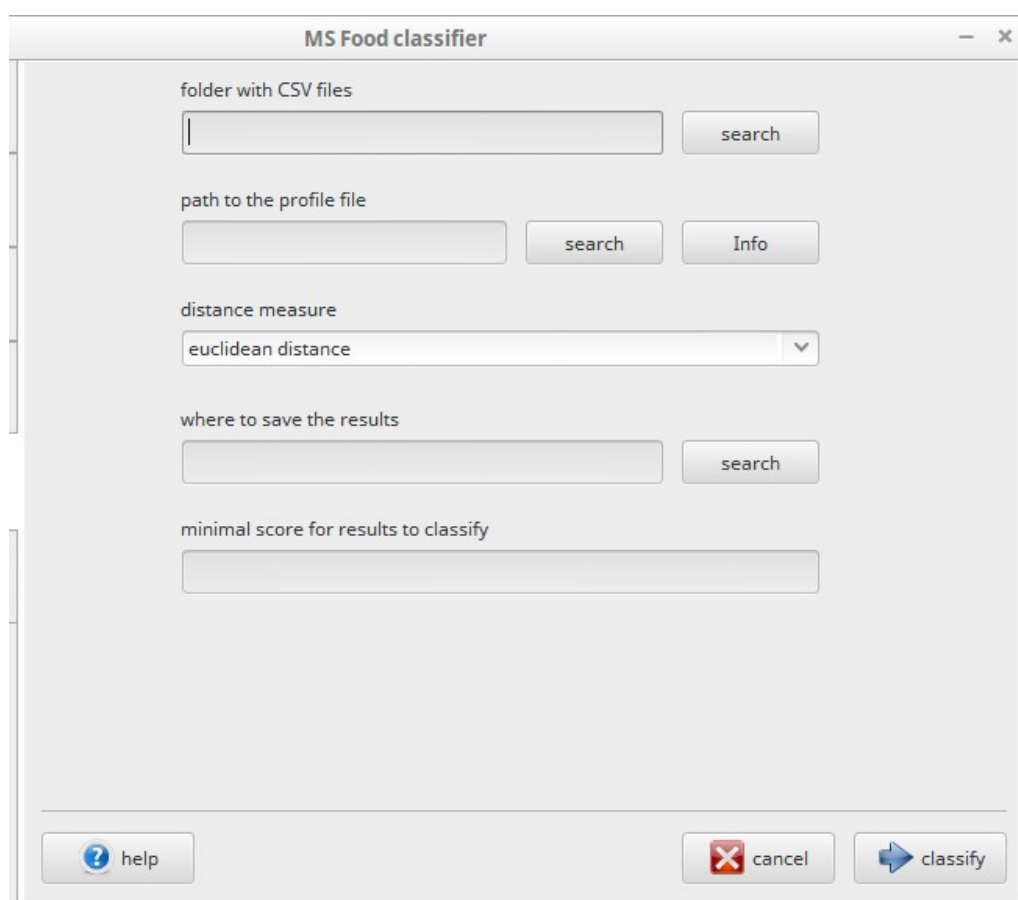


Abbildung 7: The design of the classification and live classification tab.

## Elements:

**folder with CSV files:** The folder where the MS device saves the csv files containing the spectra. You can search for the folder using the *search* button to the right of the field.

**Path to the profile file:** The complete path to the profile file to be used for classification of new samples. The *Info* button to the right of the *search* button loads the chosen profile and gives a short overview of the profile properties such as MS device, number of dimensions etc.

**distance measure:** This dropdown menu lets you choose a classification method for your samples. You can choose between three distance measures, namely the euclidean distance, the mahalanobis distance (which accounts for the different variance in the individual dimensions and is therefore the most robust method for classification) and the linear discriminant analysis.

**Where to save the results:** Choose a path and filename to a log file containing the results.

**minimal score:** Every distance measure additionally calculates a score to verify the quality of the classification. A low score usually indicates an insignificant classification (how confident are we that the made classification is correct). This score works quite well with the euclidean and mahalanobis distance but has shown to be often non significant when using LDA. If you want all spectra classified just type 0 in the field.

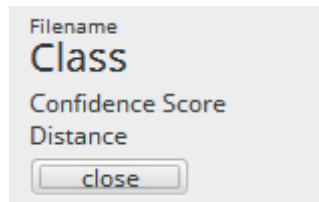
When you have everything filled out just click the *classify* button. All spectra will be classified and the results will pop up on screen in a table and will be additionally saved to your log file. The results view and log file both contain the filename of the read csv file, the assigned group, the calculated distance of the spectrum to nearest group and the score (confidence) for the classification. Illustration 8 shows the results view.

Results			
Filename	assigned class	distance	score
Soyamilch_149.csv	soja	2.142099063944843	0.9760100456931261
Soyamilch_65.csv	soja	1.7733496897473775	0.9590000305368648
Soyamilch_72.csv	soja	2.447597484944143	0.9821262031679853
Soyamilch_161.csv	soja	2.338342697227733	0.9880442996079905
Soyamilch_89.csv	soja	1.9805661561149708	0.9474579403412838
Soyamilch_135.csv	soja	2.9625607696134413	0.957535934280905
Soyamilch_214.csv	soja	3.7962879424827385	0.9528121979187567
Soyamilch_54.csv	soja	3.191871089572885	0.9441072605087283
Soyamilch_160.csv	soja	3.094401220480294	0.9655414056184238
Soyamilch_218.csv	soja	1.8843559796933664	0.9790321736607701
Soyamilch_83.csv	soja	1.5569132871886995	0.9613542141384603
Soyamilch_33.csv	soja	3.9479827849599585	0.9363197659003227
Soyamilch_41.csv	soja	1.7268531041152448	0.9870979456732195
Soyamilch_66.csv	soja	3.7851817638715315	0.8066239328764626
Soyamilch_49.csv	soja	2.5283226036688102	0.8969505032257848
Soyamilch_200.csv	soja	2.972738238747357	0.9807095780640137
Soyamilch_31.csv	soja	3.1659871114793283	0.9853097868404589
Soyamilch_202.csv	soja	2.1916963192935235	0.9772570198970584
Soyamilch_85.csv	soja	1.3068320839343524	0.9908785424289243
Soyamilch_61.csv	soja	2.4191321109284587	0.9305869896347793
Soyamilch_37.csv	soja	3.731218512748055	0.8734049795222706
Soyamilch_77.csv	soja	2.3582259622035946	0.9892069796880351
Soyamilch_87.csv	soja	2.9573053312750455	0.9170152250327642
Soyamilch_201.csv	soja	1.6407756356145822	0.989868483583737

Abbildung 8: The results of a classification run displayed in a table view.

## live classification

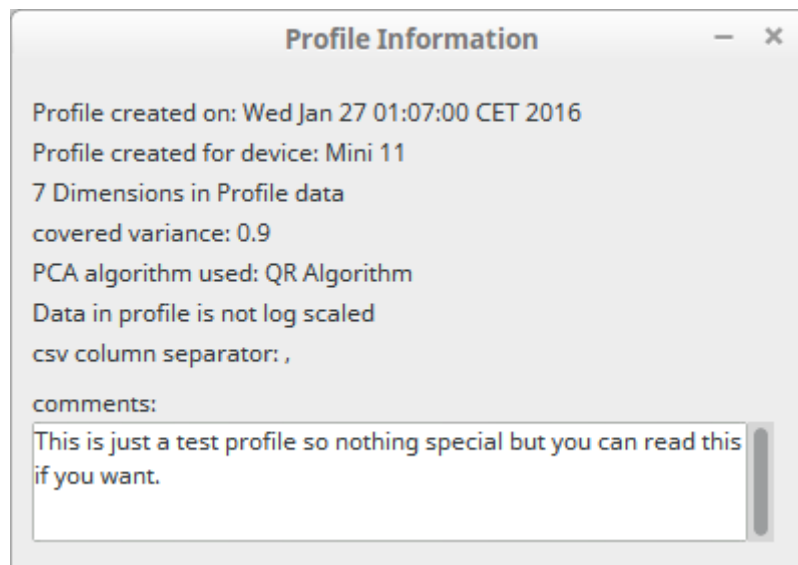
The *live classification* tab looks exactly like the *classification* tab discussed prior. Except that when you press classify a small dialog window opens at the lower right corner of the screen just above the main system bar. This dialog shows the information about the classification of a sample during an MS experiment. The chosen folder in the *live classification* tab is being monitored while the small dialog is open. When the MS devices software writes a csv file into a chosen folder the program is notified and reads the csv file and classifies it using the chosen profile. The assigned group, the measured distance and the calculated score are shown in the dialog window along with the filename of the read file. Once your MS experiment is done you can close the program by clicking on the *close* button. Illustration 9 shows the dialog window.



*Abbildung 9: the live window before anything has been classified.*

## Profile Info Dialog

The *profile info dialog* is a small dialog window which shows some information about a chosen profile. The window can be opened in the tabs *score plot*, *loading plot*, *classification* and *live classification* by pressing the *info* button after choosing a profile file. This window can be useful if you want to compare values of different profiles like number of dimensions etc. or you need to look up the comments for a profile or which device a profile was created for. Illustration 10 shows the dialog.



*Abbildung 10: profile info dialog*

# Starting from a Terminal and more Memory

## Starting from a Terminal

You can also start MSClassifier from a Terminal. This can be necessary if the program behaves unexpected without displaying an error message. Then you may want to see the error output on the terminal. To start from a Terminal the first thing you need to do is open a Terminal window:

**Windows XP/Vista/7:** click on start → type *cmd* into the search box at the bottom of the start menu → in the search results click on *cmd.exe*. A Terminal Window (called Console in Windows) should now open. Now navigate to the Folder containing your MSClassifier.jar file by using the command *cd*. If the .jar file is in the directory *C:\Users\YourName\MS*, then type *cd C:\Users\YourName\MS* into the Terminal. *Cd* stands for “change directory”.

**Linux:** Either start your Terminal from the menu (place depending on which shell you use e.g. Gnome shell or unity etc.) or use the keyboard shortcut *Ctrl+Alt+T* to open a Terminal directly. Now navigate to the Folder containing your MSClassifier.jar file by using the command *cd*. If the .jar file is in the directory */home/yourName/MS*, then type *cd /home/yourName/MS* into the Terminal. *Cd* stands for “change directory”.

**Mac:** Go to the menu and click on Applications → Utilities → Terminal. A Terminal Window should now open. Now navigate to the Folder containing your MSClassifier.jar file by using the command *cd*. If the .jar file is in the directory */home/yourName/MS*, then type *cd /home/yourName/MS* into the Terminal. *Cd* stands for “change directory”.

Once you have your Terminal open and navigated to the folder containing your .jar file you can start the program using the following command on all OS:

```
java -jar MSClassifier.jar
```

This command will start a Java Virtual machine and launch the Program MSClassifier.

## Giving the program more Memory

For very large datasets and small bin sizes it can be necessary to give MSClassifier more RAM memory to not run out of memory and crash. To give MSClassifier more memory open a Terminal and navigate to the folder containing your .jar file as discussed in the prior chapter. Then you can start the program using the command from the prior chapter including the extra parameter *-Xmx* in front of or directly after *-jar*.

The *-Xmx* parameter is paired with a number of megabytes you want your java program to get at a maximum. If you want your program to be able to use 2 gigabytes at maximum the command to do this is *-Xmx2048m* where 2048 is the number of megabytes in 2 gigabytes and *m* stands for megabytes. You can hand over any number but be aware that you shouldn't use all of your computers memory as the operating system and other currently running programs will run out of memory eventually. This can result in a complete system crash or slowing down of the system to a point where it is not usable anymore. If that

happens you need to restart your computer and use a smaller amount of memory for your program. Usually you are on the safe side if you leave at least 1.5 gigabytes to the system. Having 4 gigabytes of RAM memory that would give MSClassifier 2.5 gigabytes to use.

Example: We start MSClassifier using 2 gigabytes as maximum memory size:

```
java -Xmx2048m -jar MSClassifier.jar
```

## Developing MSClassifier

This chapter will shortly discuss some points of developing MSClassifier. Mainly we will focus on setting up the project so you can start develop fast. The project comes in a zip file. The zip file contains the main project folder. In its subfolders lie all the source code assets and libraries for the project.

1. IDE: MSClassifier was developed using NetBeans IDE 8.1
2. Java Version: Oracle Java 8 was used. Do not use OpenJDK of any version since the libraries and the program itself were only compiled and tested using Oracle JDK so stability cannot be ensured when using OpenJDK.
3. Versioning: Git was used as Versioning system. The Git project files are still in the project folder in the subfolder *.git*. Although you can delete them and use whatever versioning system you want for development. But this deletes old versions of the source code as well.

## Import the Project into NetBeans

Importing the Project into NetBeans is easy. Just unzip the zip file into any location you want. Then open NetBeans IDE and go to the menu → *File* → *Open Project* → navigate to your project folder and click *open project*. The project should now be loaded into your IDE under the name MassSpec.

If your JDK version is not recognized then NetBeans will give you an error at the beginning. This could be because the name of the configured JDK version in the project has an underscore instead of a space in it so NetBeans does not recognize it as the default Java 8. You can fix this by going to the menu → *Tools* → *Java Platforms* and then click on *Add Platform*. A Dialog opens. Select *Java Standard Edition* and click *next*. Then navigate to the folder containing your Oracle Java 8 JDK. If you don't know where that folder is just look it up by clicking on your default JDK 8. It is the exact same folder. Choose the folder and click *next*. Then type the name *JDK\_1.8* as *Platform name* and click *finish*.

## What Libraries are used and how to import them

**Apache Commons IO 2.4:** The program uses the Apache Commons IO library in the version 2.4 for some of its operations. The .jar file of the library is stored in the *lib* folder within the project folder. Project side: <https://commons.apache.org/proper/commons-io/>



**Jide-oss:** The program uses the JIDE OSS library to generate the folder tree structure in the *create profile* and *cross validation* tabs. The libraries .jar file is stored in the *lib* folder of the project. Project site: <http://www.jidesoft.com/products/oss.htm>

**JmathPlot:** To create the 2D and 3D plots the program uses the jMathPlot library. The libraries .jar file is stored in the *lib* folder of the project. Project site: <https://github.com/yannrichet/jmathplot>

**WEKA:** The program uses the WEKA library for all linear algebra calculations. The libraries .jar file is stored in the *lib* folder of the project. Project site: <http://www.cs.waikato.ac.nz/ml/weka/>

Although you can download and use newer versions of the libraries it is highly recommended that you use the given libraries in the *lib* folder of the project for compatibility reasons. Newer versions might have removed or altered features and stability can therefore not be guaranteed.

Click on the + symbol in front of the project name in the *Projects* view (*Ctrl + 1* to open view) on the left side of the screen and navigate to *Libraries*. Again click on the + symbol in front of *Libraries*. Check if you see the following entries (Illustration 11):

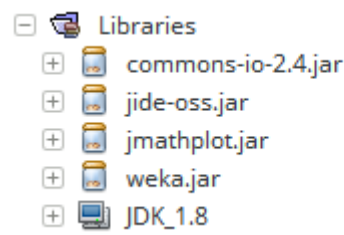


Abbildung 11: The libraries of the project.

If you cannot see the libraries as in Illustration 11 then right-click onto *Libraries* and click on *Add/JAR Folder*. A Dialog will open. Navigate to the *lib* folder of the project and import the .jar files in it.

## Where is the Javadoc

To ease development of the program Javadoc files have been created. The javadoc files are in the folder *dist\javadoc* in the project folder. By opening the file *index.html* in the javadoc folder with a web-browser you can view the documentation of the code.

## How to build the Jar file

To build the JAR file for the project containing all the libraries in it navigate to the Files view on the left side of the screen (the second tab). If the view is not present you can open it either via *Ctrl+2* or by navigating to the menu → *Window* → *Files*. In the file view right click on the file *build.xml* of the project and navigate to *Run Target* → *Other Targets* → *package-for-store*. The process is shown in Illustration 12. This runs the package-for-store target defined in the build.xml file. The compiled JAR will show up in the *store* folder in the

project folder. Warning: do not change anything in the file except the name of the JAR file to create:

```
<property name="store.jar.name" value="MSClassifier"/>
```

The following line of code shows the property for defining the name for the output jar. The value after *value=* is the name of the jar. This property is the only one that can be changed. Every other change can damage the build script.

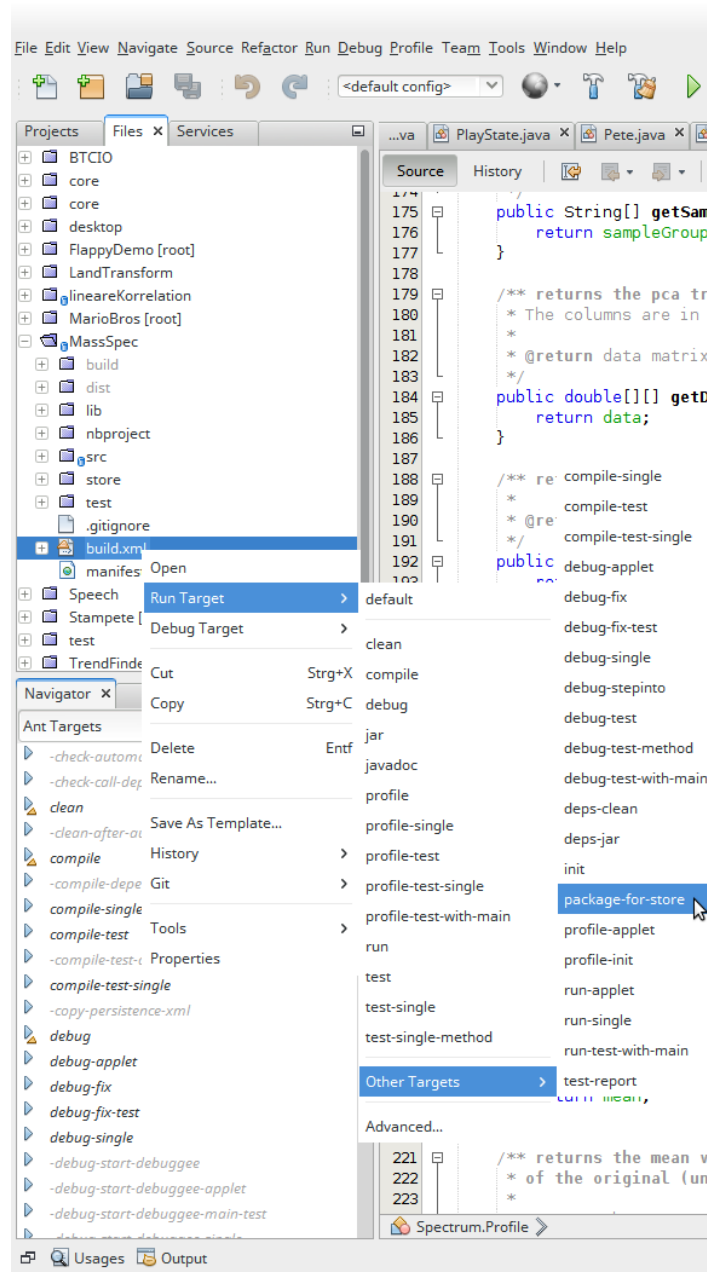


Abbildung 12: The menu item to build the JAR file for the project