# Cortana Light Manual

## 1 What is Cortana, and what does it do?

Cortana is a tool which, given a dataset, performs Subgroup Discovery. This is a form of Supervised Descriptive Local Pattern Mining:

**Local Pattern Mining:** a run of Cortana strives to find *subsets* of the dataset where *something interesting* is going on. We try to pinpoint local exceptionalities in your dataset.

**Descriptive:** subsets delivered by Cortana are not just any subsets of the data, but *coherent* subsets, defined on attributes. Such subsets are called *subgroups*.

**Supervised:** the interestingness is measured with respect to a user-defined *target concept*: a subset of attributes that we are particularly interested in. Cortana finds subgroups for which the target concept is substantially different than the target concept over the entire dataset.

As an example, suppose a dataset about people, and let the target concept be the distribution of just one binary attribute: whether the person develops lung cancer or not. Cortana can find subgroups like *smoking = true* (the smokers, who have a substantially higher incidence of lung cancer), and *weekly walking kilometers* $\geq 50$ (athletes, who have a substantially lower incidence of lung cancer).

## 2 Launching Cortana

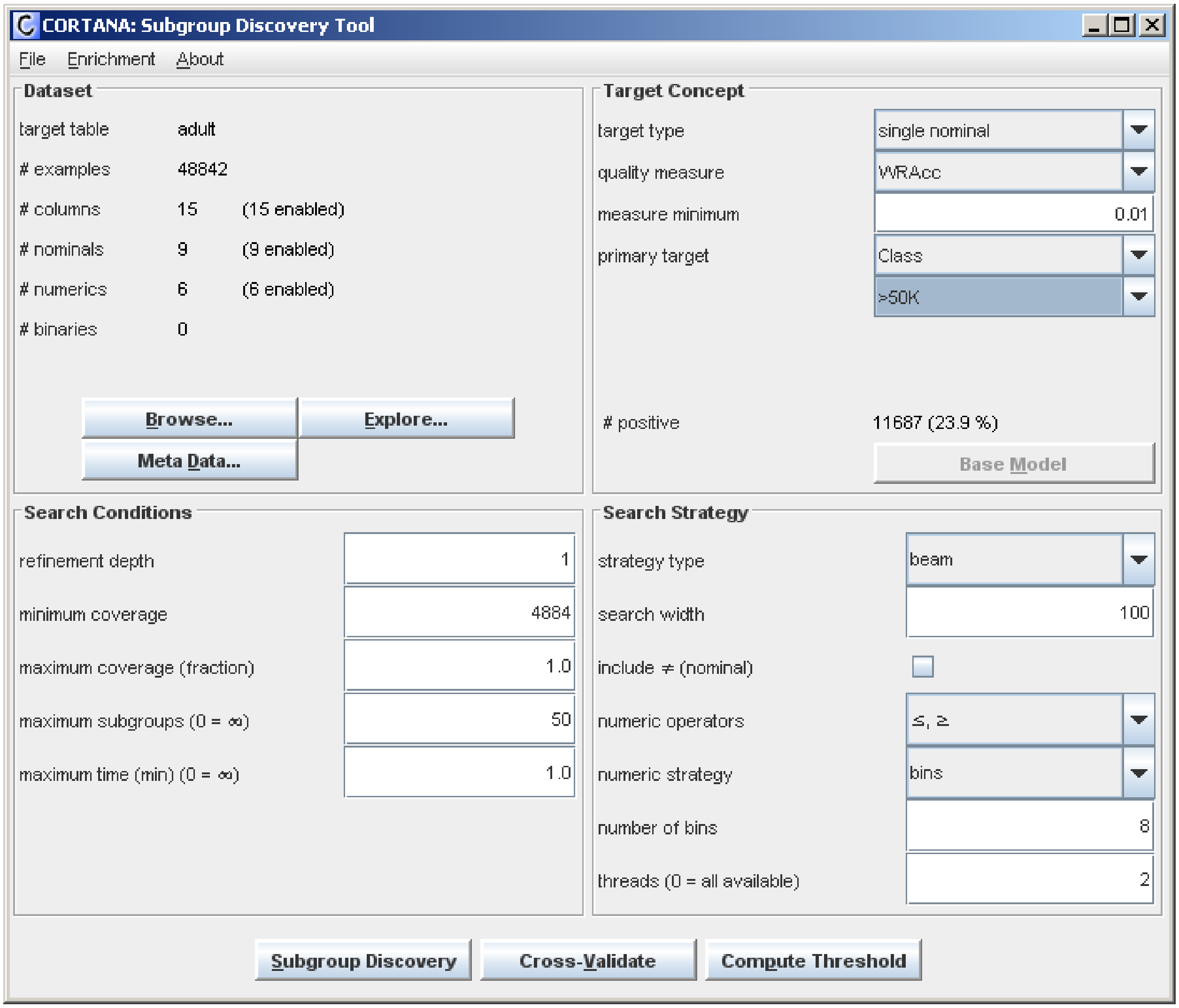
After obtaining and unpacking a copy of Cortana from `http://datamining.liacs.nl/cortana.html`, navigate to the Cortana directory where you will find a cortana.jar file. One could start Cortana by double clicking the jar. However, it is recommended to start Cortana from the command line, since this will feed back information about the Subgroup Discovery process to the user. To start Cortana this way, open a terminal or command window, navigate to the directory containing cortana.jar, and type: *java -jar cortana.jar*. Alternatively, one can use either cortana.bat (for Windows) or cortana.sh (Bash shell script). Be sure to read Appendix B if your computer displays memory issues while running Cortana.

After starting Cortana you are prompted for the file containing the dataset to be analysed, after which Cortana's main screen is shown.

Cortana can handle two kinds of files: ARFF files and plain (comma seperated) text files. An ARFF file combines the data with definitions of the type of each attribute. For text files, Cortana wil try to automatically infer these types. Unfortunately, this may not always deliver the desired outcome. We will discuss how to correct wrong assessments in Section 3.1.3, but in order to prevent the problem, we recommend the use of ARFF files when this option is available.

## 3 Main Screen

The main screen of Cortana is devided into four major panels, *Dataset*, *Target Concept*, *Search Conditions*, and *Search Strategy*.



Dataset

target table adult

# examples 48842

# columns 15 (15 enabled)

# nominals 9 (9 enabled)

# numerics 6 (6 enabled)

# binaries 0

Target Concept

target type: single nominal

quality measure: WRAcc

measure minimum: 0.01

primary target: Class

>50K

# positive 11687 (23.9 %)

Search Conditions

refinement depth: 1

minimum coverage: 4884

maximum coverage (fraction): 1.0

maximum subgroups (0 = ∞): 50

maximum time (min) (0 = ∞): 1.0

Search Strategy

strategy type: beam

search width: 100

include ≠ (nominal): □

numeric operators: ≤, ≥

numeric strategy: bins

number of bins: 8

threads (0 = all available): 2

Figure 1: Cortana's main window.

## 3.1 Dataset Panel

The Dataset panel contains information about the dataset that is currently loaded:

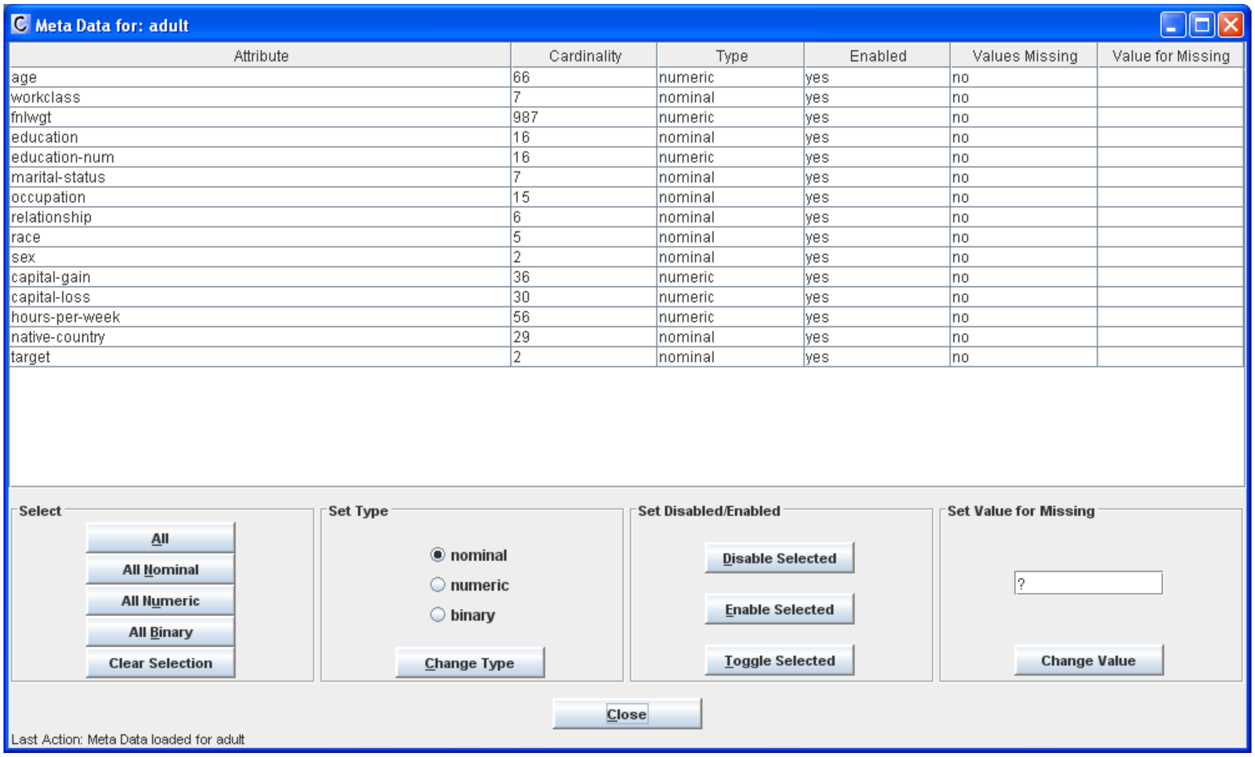
**target table** shows the name of the data file used. For text files this is the filename, and for ARFF files this is the name defined in the '@relation' field;

**# examples** shows the number of examples in the dataset;

**# columns** shows the number of columns (attributes) in the dataset.

Finally, there is a number of fields that indicate the number of attributes from each type: nominal, numeric, and binary. Notice that Cortana cannot handle ordinal data; such attributes can be handled as either nominal or numeric, depending on what kind of subgroups you would want to define on the attribute.

In addition to the fields described above, there are three buttons present on the **Dataset** panel: **Browse. . .** , **Meta Data. . .** and **Explore. . .** .



Meta Data for: adult

| Attribute | Cardinality | Type | Enabled | Values Missing | Value for Missing |
|---|---|---|---|---|---|
| age | 66 | numeric | yes | no | |
| workclass | 7 | nominal | yes | no | |
| fnlwgt | 987 | numeric | yes | no | |
| education | 16 | nominal | yes | no | |
| education-num | 16 | numeric | yes | no | |
| marital-status | 7 | nominal | yes | no | |
| occupation | 15 | nominal | yes | no | |
| relationship | 6 | nominal | yes | no | |
| race | 5 | nominal | yes | no | |
| sex | 2 | nominal | yes | no | |
| capital-gain | 36 | numeric | yes | no | |
| capital-loss | 30 | numeric | yes | no | |
| hours-per-week | 56 | numeric | yes | no | |
| native-country | 29 | nominal | yes | no | |
| target | 2 | nominal | yes | no | |

Set Type: ● nominal ○ numeric ○ binary

Set Value for Missing: ?

Last Action: Meta Data loaded for adult

Figure 2: Meta Data window.

### 3.1.1 Browse Window

Clicking the **Browse** button will open a Browse Window, displaying the data in its current state. You can sort the data by clicking on the column head of an attribute, and save the current state of the data to a file, which is useful if you have made modifications via the Meta Data Window (see Section 3.1.3).

### 3.1.2 Explore Window

Clicking the **Explore** button will open an Explore Window, which allows you to examine the distribution of single attributes, and cross-examine two attributes. This section of Cortana is independent of the Local Pattern Mining for which Cortana was developed; it exists merely to allow you to learn more about the makeup of your data.

### 3.1.3 Meta Data Window

Clicking the **Meta Data** button will open a Meta Data window, which displays information on the attributes of the current dataset and allows changing some of the attribute properties. The upper part of the window shows a table with six columns. The lower part contains a number of panels that allow modification of the data as it is in memory. Note that no modifications are made to the original data file. First the properties shown in the table in the upper part will be described, the purpose of the various data manipulations will be explained after that.

**Meta Data Table** The upper part of the MetaData window displays a table containing the current attribute information. The first column in this table, *Attribute*, lists the attribute names. *Cardinality* gives the number of distinct values for the attribute. *Type* shows its attribute type. *Enabled* indicates whether the attribute is enabled or disabled; this setting affects the search for subgroups, which will be discussed in Section 4. *Values Missing* indicates whether values for the attribute are missing in the dataset, and if so, *Value for Missing* shows the value that is currently used for these missing values. If not, this field is blank.

**Meta Data Functions** The panels in the lower part of the *Meta Data Window* all allow selecting or changing the data. The first, *Select*, allows selecting all attributes of a certain type. This is a convenience method to be used in combination with the functionalities available in other panels.

**Change Type** allows changing the attribute type. Cortana makes an educated guess which type each attribute should have. Occasionally you will want to change this assessment, and if possible this panel will allow it. Select the attributes in the table in the upper part of the MetaData window, choose the radio button corresponding to the desired type, and click the Change Type button. If the requested action can be performed, Cortana will do so, and if not (for instance: if you try to make an attribute binary while it has more than two distinct values), Cortana will do nothing.

**Set Disabled/Enabled** allows to disable or enable an attribute. The relevance of this choice will be discussed in Section 3.4; by default all attributes are enabled.

**Set Value for Missing** can be used to change the value that is currently used as a proxy for missing occurrences of this attribute in the data. Details can be found in the Full Cortana Manual [1].

## 3.2 Target Concept Panel

The settings in the Target Concept panel determine how the Cortana process is *supervised*, as described in Section 1. A Cortana run strives to find subgroups for which a target concept is substantially different than the target concept over the entire dataset. In the Cortana Light Manual, we will focus on the case where this target concept is the distribution of a single attribute. Cortana can also find subgroups for more complex target concepts; these are described in the Full Cortana Manual [1].

Since the type of the target attribute determines the number of parameters to be set, you have to indicate it first. Hence, the first drop-down menu in the Target Concept panel is **target type**, which more generally determines what sort of target concept we are interested in. The *single nominal* and *single numeric* choices correspond to having one single target attribute, either nominal or numeric.

When the target attribute type is nominal, you have to set four additional parameters. In the third drop-down menu (**primary target**) of the Target Concept panel, you can select the target attribute, and in the fourth drop-down menu, you can select the desired value for that target (i.e. the value for which you want to see an increased or decreased presence in the found subgroups). The second drop-down menu (**quality measure**) lists the available quality measures. The default selection, “WRAcc”, will only search for subgroups with an increased presence of the target value. If you are also interested in subgroups with a decreased presence of the target value, you can select either “Abs WRAcc” or “Chi-squared”. Directly below the quality measure drop-down menu, you find a field where you can enter the minimal quality value a subgroup must have (**measure minimum**). Usually the default value should do, but if your subgroup discovery run provides no results, a lower measure minimum might help.

When the target attribute type is numeric, you have to set three additional parameters. In the third drop-down menu (**primary target**) of the Target Concept panel, we have to select the target attribute. Unlike in the nominal case, there is not one particular value for the attribute to be interested in; instead we search for subgroups with relatively high or relatively low values for the target. The second drop-down menu (**quality measure**) lists the available quality measures. The default selection, "Z-Score", will only search for subgroups with a relatively high average target value. If you are interested in subgroups with a relatively low average target value, you can select "Inverse Z-Score", and if you are interested in both, you can select "Abs Z-Score". Again, directly below the quality measure drop-down menu, you can set the **measure minimum**, and again, usually the default value should do, but if your subgroup discovery run provides no results, a lower measure minimum might help.

## 3.3 Search Conditions Panel

The fields in the Search Conditions panel allow setting the search conditions used in the Subgroup Discovery process. The process is discussed in detail in Section 4.

**refinement depth** controls the number of conditions that is used to define subgroups. A refinement depth of 1 would lead to subgroups of the form $x \leq 9.11$, while a refinement depth of 2 would allow for subgroups of the form $x \leq 1.2 \wedge y \geq 3.4$. It is not recommended to set the refinement depth to very high values. Usually, the added conditions on depth 4 or 5 and beyond do not substantially contribute to subgroups, but merely make them harder to interpret while the increased depth does substantially add computational time needed to calculate the result.

**minimum coverage** sets the lower bound for the size of the subgroups that should be reported by Cortana, meaning all reported subgroups have at least this size. The default value is 10% of the dataset size.

**coverage fraction** sets the upper bound for the size of the subgroups that should be reported by Cortana, meaning all reported subgroups have at most this size. The default value is set to 1.0, which allows subgroups to encompass the entire dataset.

**maximum subgroups** is used to control the maximum number of subgroups in the result list generated by Cortana. Setting this parameter to 0 allows Cortana to report all found subgroups.

**maximum time (min)** is used to control the maximum time, measured in minutes, Cortana is allowed to search for new subgroups. If the time limit is reached, Cortana will abort the Subgroup Discovery process and report all subgroups found so far. Usually this implies that a substantial part of the search space has not been explored at all. You can set this parameter to 0 to allow Cortana as much time as necessary.

## 3.4 Search Strategy Panel

The Search Strategy panel determines the terms under which the search process is carried out. Usually the default values will do, apart maybe from the search width and the operator selection parameters. To fully grasp the impact of these parameters, you might want to read about the search process (Section 4) first, and revisit this section afterwards.

**strategy type** allows you to set the overarching search strategy. In this manual we will focus on the "beam" setting; for more on the other strategies, please see the Full Cortana Manual [1]. If you find the reported subgroups too similar to each other, you could alternatively try the "cover-based beam selection" setting [2].

**search width** sets the beam width. See Section 4 for details.

**include ≠ (nominal)** determines the subgroups considered for each nominal attribute. When unchecked, on each nominal attribute only subgroups of the form *attr=value* will be considered. When checked, *additionally* subgroups of the form *attr≠value* will be considered.

Together, **numeric operators**, **numeric strategy**, and **number of bins** determine the subgroups considered for each numeric attribute. Usually, the default selection will do. For more details, see the Full Cortana Manual [1].

**threads** sets the number of hyperthreads used on your computer for the Cortana process. When you run Cortana on a multi-core system, increasing this number will reduce the time necessary to complete the process. You can set the parameter to 0 to allow Cortana to use every available hyperthread.

# 4 Beam Search Process

Cortana's search process is an example of Beam Search, which is a level-wise search largely governed by two parameters: the search depth and the search width. The search depth can be set in the **refinement depth** field in the Search Conditions panel; we denote its value by $d$. The search width can be set in the **search width** field in the Search Strategy panel; we denote its value by $w$.

Each search level starts with a generating pool of subgroups, the beam. The initial beam consists of only one subgroup: the entire dataset. Cortana generates a set of candidate subgroups from the beam, by refining each subgroup in the beam.

For each subgroup to be refined, Cortana considers each enabled non-target attribute in the dataset. Depending on the attribute type, a number of subgroups will be generated from the original subgroup, each by imposing one additional constraint on the current attribute. The quality of these new subgroups is computed, and they are gathered in one large candidate pool.

When all subgroups are refined, the candidate pool is trimmed: the subgroups are ranked by their quality measure values, and only the best $w$ (the search width) subgroups are kept. These subgroups form the beam for the next level.

This process is repeated until $d$ (the search depth) beams are refined. That implies that the resulting subgroups are defined as a conjunction of at most $d$ conditions, each on a single attribute. The best subgroups considered during this entire process are then reported in a Result Window.

# 5 Result Window

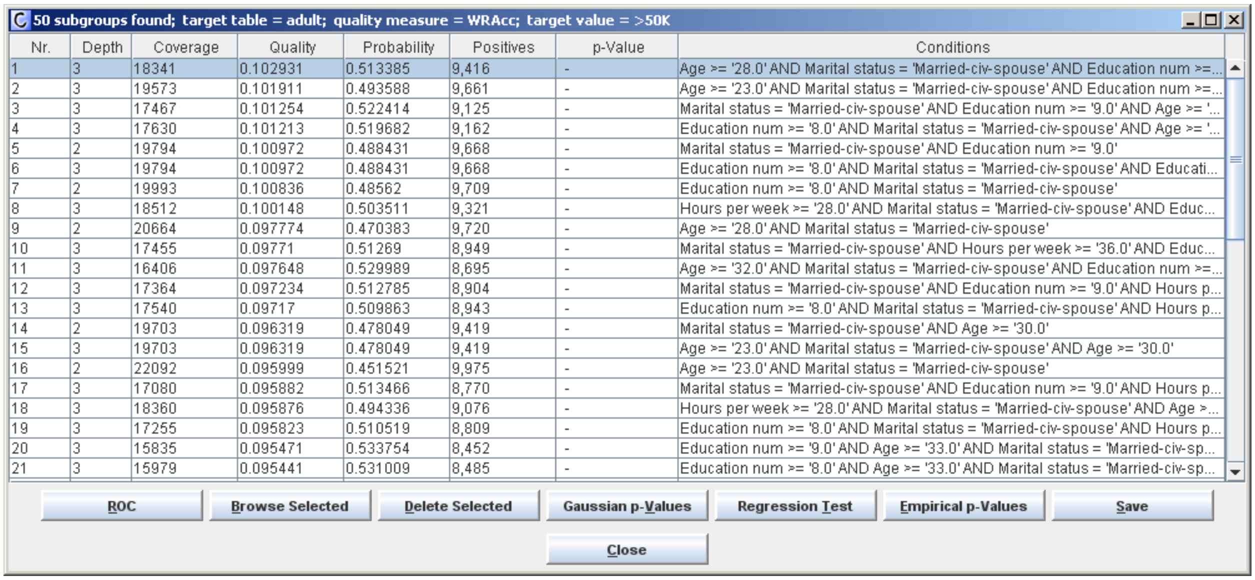
When all parameters are set, clicking the **Subgroup Discovery** button in the Main Window will set the Beam Search process in motion as described in the previous section. At the end of the process, you will be presented with a Result Window. An example of such a window is displayed in Figure 3. The window consists of a table displaying the found subgroups, and buttons to perform actions on the subgroups.

## 5.1 Result Set Table

This section will address all columns of the table displaying the found subgroups, and explains how to interpret the information they display.

**Nr.** shows the rank of the subgroup by quality. Ties are ignored.

**Depth** shows the search depth at which the subgroup was found.


50 subgroups found; target table = adult; quality measure = WRAcc; target value = >50K

| Nr. | Depth | Coverage | Quality | Probability | Positives | p-Value | Conditions |
|---|---|---|---|---|---|---|---|
| 1 | 3 | 18341 | 0.102931 | 0.513385 | 9,416 | - | Age >= '28.0' AND Marital status = 'Married-civ-spouse' AND Education num >=... |
| 2 | 3 | 19573 | 0.101911 | 0.493588 | 9,661 | - | Age >= '23.0' AND Marital status = 'Married-civ-spouse' AND Education num >=... |
| 3 | 3 | 17467 | 0.101254 | 0.522414 | 9,125 | - | Marital status = 'Married-civ-spouse' AND Education num >= '9.0' AND Age >= '... |
| 4 | 3 | 17630 | 0.101213 | 0.519682 | 9,162 | - | Education num >= '8.0' AND Marital status = 'Married-civ-spouse' AND Age >= '... |
| 5 | 2 | 19794 | 0.100972 | 0.488431 | 9,668 | - | Marital status = 'Married-civ-spouse' AND Education num >= '9.0' |
| 6 | 3 | 19794 | 0.100972 | 0.488431 | 9,668 | - | Education num >= '8.0' AND Marital status = 'Married-civ-spouse' AND Educati... |
| 7 | 2 | 19993 | 0.100836 | 0.48562 | 9,709 | - | Education num >= '8.0' AND Marital status = 'Married-civ-spouse' |
| 8 | 3 | 18512 | 0.100148 | 0.503511 | 9,321 | - | Hours per week >= '28.0' AND Marital status = 'Married-civ-spouse' AND Educ... |
| 9 | 2 | 20664 | 0.097774 | 0.470383 | 9,720 | - | Age >= '28.0' AND Marital status = 'Married-civ-spouse' |
| 10 | 3 | 17455 | 0.09771 | 0.51269 | 8,949 | - | Marital status = 'Married-civ-spouse' AND Hours per week >= '36.0' AND Educ... |
| 11 | 3 | 16406 | 0.097648 | 0.529989 | 8,695 | - | Age >= '32.0' AND Marital status = 'Married-civ-spouse' AND Education num >=... |
| 12 | 3 | 17364 | 0.097234 | 0.512785 | 8,904 | - | Marital status = 'Married-civ-spouse' AND Education num >= '9.0' AND Hours p... |
| 13 | 3 | 17540 | 0.09717 | 0.509863 | 8,943 | - | Education num >= '8.0' AND Marital status = 'Married-civ-spouse' AND Hours p... |
| 14 | 2 | 19703 | 0.096319 | 0.478049 | 9,419 | - | Marital status = 'Married-civ-spouse' AND Age >= '30.0' |
| 15 | 3 | 19703 | 0.096319 | 0.478049 | 9,419 | - | Age >= '23.0' AND Marital status = 'Married-civ-spouse' AND Age >= '30.0' |
| 16 | 2 | 22092 | 0.095999 | 0.451521 | 9,975 | - | Age >= '23.0' AND Marital status = 'Married-civ-spouse' |
| 17 | 3 | 17080 | 0.095882 | 0.513466 | 8,770 | - | Marital status = 'Married-civ-spouse' AND Education num >= '9.0' AND Hours p... |
| 18 | 3 | 18360 | 0.095876 | 0.494336 | 9,076 | - | Hours per week >= '28.0' AND Marital status = 'Married-civ-spouse' AND Age >... |
| 19 | 3 | 17255 | 0.095823 | 0.510519 | 8,809 | - | Education num >= '8.0' AND Marital status = 'Married-civ-spouse' AND Hours p... |
| 20 | 3 | 15835 | 0.095471 | 0.533754 | 8,452 | - | Education num >= '9.0' AND Age >= '33.0' AND Marital status = 'Married-civ-sp... |
| 21 | 3 | 15979 | 0.095441 | 0.531009 | 8,485 | - | Education num >= '8.0' AND Age >= '33.0' AND Marital status = 'Married-civ-sp... |

Figure 3: A Result Window

**Coverage** shows the number of examples in the subgroup.

**Quality** shows the score of the subgroup, according to the quality measure used. Different quality measures have different ranges for the scores they asssign to the worst and best possible subgroups. As a result, measure scores resulting from applying different quality measures cannot be compared directly in any sensible manner.

**Probability** is only shown when the target attribute is nominal. It displays the probability that an example has the desired target value, conditional on the fact that the example is in the subgroup. If the target attribute is numeric, this column is replaced with **Average**, which displays the average target value in the subgroup.

**Positives** is only shown when the target attribute is nominal. It displays the number of examples in the subgroup that have the desired target value. If the target attribute is numeric, this column is replaced with **St. Dev.**, which displays the standard deviation of the target values in the subgroup.

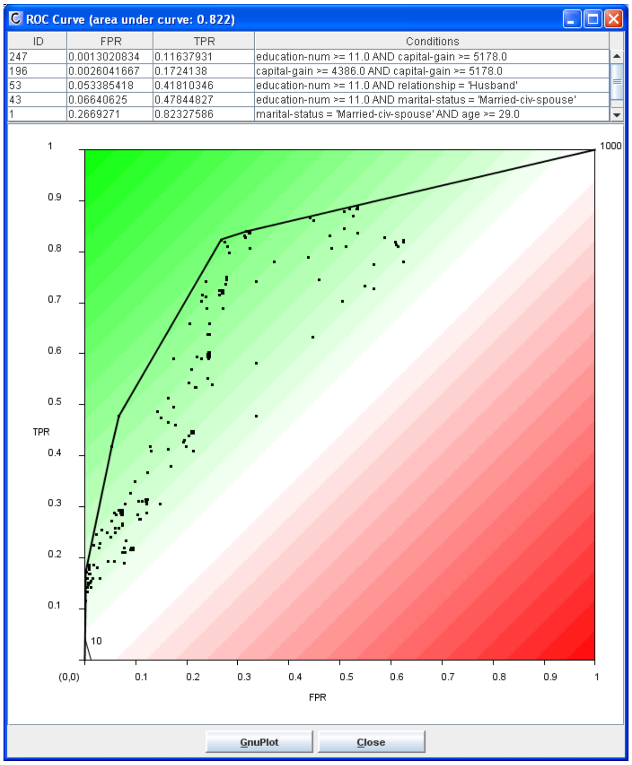
**p-Value** is initially blank. This will change when clicking the **Gaussian p-Values** button, as described in Section 5.2.

**Conditions** shows the conditions that define the subgroup.

## 5.2 Result Set Buttons

The following paragraphs describe the various buttons that can appear in the *Result Window*. Not all buttons are always available; this may depend on the target type setting used for the Subgroup Discovery experiment.

**ROC** is only available for the *single nominal* target type. All subgroups in the Result Set Table will be taken into account to show an *ROC Curve Window*. ROC (Receiver Operating Characteristic) curves have their origin in electrical engineering, where they are used to show the performance of classifiers. In the *single nominal* setting, those examples in a subgroup that have the desired target value are called *true positives*, while the others are called *false positives*. In an ROC plot the *true positive rate* (TPR) of each subgroup is



ROC Curve (area under curve: 0.822)

| ID | FPR | TPR | Conditions |
|---|---|---|---|
| 247 | 0.0013020834 | 0.11637931 | education-num >= 11.0 AND capital-gain >= 5178.0 |
| 196 | 0.0026041667 | 0.1724138 | capital-gain >= 4386.0 AND capital-gain >= 5178.0 |
| 53 | 0.053385418 | 0.41810346 | education-num >= 11.0 AND relationship = 'Husband' |
| 43 | 0.06640625 | 0.47844827 | education-num >= 11.0 AND marital-status = 'Married-civ-spouse' |
| 1 | 0.2669271 | 0.82327586 | marital-status = 'Married-civ-spouse' AND age >= 29.0 |

Figure 4: An ROC window

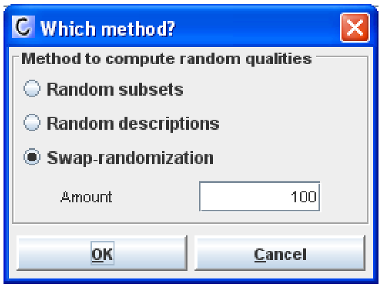
plotted against its *false positive rate* (FPR). A set of found subgroups can be depicted as a set of point in this two-dimensional space. The convex hull of this point set, the *ROC curve*, can be drawn, and the area under this curve can be seen as a measure of the quality of the subgroup set. If the ROC curve is a straight diagonal from the lower left corner to the upper right, then the result is indistinguishable from complete randomness. The closer the curve approaches the upper left corner, the better the result. For a subgroup to reach the upper left corner, its TPR must be 1, and its FPR 0, hence this subgroup would contain precisely all examples in the dataset having the desired target value, and none of the examples not having the desired target value.

Clicking the button shows a plot of the current result set in ROC space with the convex hull, and a table of those subgroups that lie on the convex hull.

**Browse Selected** shows a Browse Window (see Section 3.1.1) for each subgroup selected in the Result Set Table. This window depicts all examples that constitute the subgroup. For the *single nominal* target type one additional button is available, the **True Positives** button. Clicking it will highlight only those instances having the desired target value.

**Delete Selected** deletes all subgroups selected in the Result Set Table from the result set.

**Gaussian p-Values** performs a permutation test to compute p-values, corresponding to the null hypothesis that the found subgroups are false discoveries [3]. After clicking the button, the user is presented with the dialog depicted in Figure 5.2. The **Swap-randomization** radio button is preselected and should be selected (the other two methods are obsolete, and may be removed). The number of repetitions is preset to 100 in the **Amount** field; you could lower the number of repetitions to speed up the p-value computation, at the cost of p-value accuracy. Notice that the test will simulate false discoveries, using a subgroup discovery run with the parameter settings you selected in the Main Window. If these settings are too strict, it might be



Which method?

Method to compute random qualities

○ Random subsets

○ Random descriptions

● Swap-randomization

Amount 100

Figure 5: Choosing the type of randomization to be used for any significance testing

possible that no false discoveries can be found, and the p-value computation could get stuck in an infinite loop.

**Regression Test** is obsolete functionality, and will be removed.

**Empirical p-Values** might be removed.

**Save** allows saving of the whole result set to a file. The file will contain the information as it is depicted in the Result Set Table, and the values for the various columns will be comma-separated.

**Close** closes the *Result Window*, which can not be reopened. Therefore, all necessary actions like, saving or printing the results, should be done before closing the Result Window. All results will be lost, and the only way to get a new Result Window is to run the same subgroup discovery process again. As this may be a very lengthy process, be sure to finish all operations before closing the Result Window.

## References

[1] Cortana Manual. `http://datamining.liacs.nl/cortana/manual.pdf`, to appear.

[2] M. van Leeuwen, A. Knobbe: Non-Redundant Subgroup Discovery in Large and Complex Data. In: Proc. ECML PKDD 2011, pp. 459–474.

[3] W. Duivesteijn, A. Knobbe: Exploiting False Discoveries – Statistical Validation of Patterns and Quality Measures in Subgroup Discovery. In: Proc. ICDM 2011, pp. 151–160.

# Appendices

## A Glossary

**beam** The set of subgroups used as generating pool for the next search level.

**binary** An attribute that can have only two distinct values. These values will be represented in Cortana with 0 and 1. Cortana will form subgroups *attr*=0 and *attr*=1.

**disabled** An attribute that will not be considered for generating candidate subgroups.

**enabled** An attribute that will be considered for generating candidate subgroups.

**nominal** An attribute that can have any countable number of distinct values. No ordering is assumed between these values. Cortana will form subgroups *attr=value*, and *attr≠value* when the "include ≠ (nominal)" checkbox in the Search Strategy Panel is selected (see Section 3.4).

**numeric** An attribute that can have values from a continuous range. Cortana will form subgroups *attr≤value* and *attr≥value*. You can control how many and which values will be selected in the Search Strategy Panel (see Section 3.4).

**ordinal** An attribute that can have any countable number of distinct values. A natural ordering exists between these values. Cortana cannot handle this attribute type. Depending on which kind of subgroups you would want to consider, you can make Cortana handle such an attribute as either nominal or numeric.

**search depth** The number of levels Cortana uses in its mining process. This parameter also restricts the number of conditions on one attribute, that can be imposed on one subgroup.

**search width** The number of subgroups Cortana retains at the end of each level of its mining process, and uses to generate candidate subgroups for the next level.

**type** Determines the range of values an attribute can have. In Cortana, an attribute can have types binary, nominal, or numeric. Attributes in Cortana cannot be ordinal; for these attributes you can either select the nominal or numeric type, depending on what kind of subgroups you would like to see.

## B Memory Issues

Although Cortana is written in Java, and therefore platform independent, it will behave slightly different on different operation systems and/or platforms. These differences arise from small variations in the Java Virtual Machines, used in different situations. The main issue is with 32-bit operating systems (OS). On such systems the maximum amount of memory the Java Virtual Machine (JVM) can use is around 1600 MegaBytes. However, the actual amount depends on the amount of RAM available. The cortana.bat and cortana.sh file included in the Cortana.zip set the maximum amount of memory the JVM can use to 1600 MegaBytes, through the -Xmx option. The value should, at most, be set to half the amount of available RAM, meaning eg. for a 2GB machine to *-Xmx1000m*. For 64-bit OSes no such limit exists, and it should be save to remove the -Xmx. Note that the above means that, especially for 32-bit OSes, not all datasets will fit into memory.