Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea Magistrale in Informatica

DATA SANITIZATION IN A CLEARING SYSTEM FOR PUBLIC TRANSPORT OPERATORS

Tesi di Laurea in Sicurezza

Relatore:
Chiar.mo Prof.
MARCO PRANDINI

Correlatore:
Chiar.mo Prof.
ALDO CAMPI
Chiar.mo Prof.
MORENO MARZOLLA

Presentata da:
ANDREA MELIS

Sessione I
Anno Accademico 2013-2014

...It's just a ride!

(Bill Hicks)

Abstract

In this work we will discuss a project developed by the Regional Government of Emilia-Romagna, regarding the management of public transport. In particular we will perform a data mining analysis on the dataset of this project.
After introducing the WEKA software used to make our analysis, we will discover the most useful data mining techniques and algorithms.
Also, we will show how these results can be used to violate the privacy of these public transport operators.
Furthermore, we will investigate some preventative techniques used to manage these kinds of attacks.

Abstract

In questo lavoro discuteremo alcuni fasi di implementazione di un sistema creato dal governo della Regione Emilia-Romagna riguardante la gestione del trasporto pubblico regionale.
In particolare effettueremo delle analisi di data mining sui dati gestiti da questo stesso sistema.
Successivamente, dopo aver introdotto il software WEKA utilizzato per effettuare le nostre analisi, scopriremo quali sono attualmente allo stato dell'arte le migliori tecniche, e soprattutto algoritmi, di data mining.
Vedremo quindi come è possibile utilizzare questi risultati per violare la privacy degli stessi operatori di trasporto coinvolti nel sistema.
Alla fine, nonostante esuli dallo scopo del nostro lavoro, parleremo in breve su cosa si possa fare per prevenire questo tipo di attacco.

Contents

List of Figures

Chapter 1

Introduction

Recently the digitization of those services managed by the public administration has become a very important aspect in its process of innovation and development.
In this context, a few years ago the Emilia-Romagna Regional Government initiated a project in order to create a general system capable of unifying all single private transport companies under a unique control system.
The final decision was to develop a generalized system.
This would enable passengers to purchase a singular ticket that could be used for buses and trains across all the various operators.
The application of one broad system would pose some crucial questions.
The first of which being:
How do we re-distribute the revenue from tickets sales to the single operators, in accordance to their incurred costs?
It would be necessary to implement a clearing system. Designed to compute all the economic evaluations based on the data generated by the operators.
Of course, in order to implement this system, operators need to be capable of registering by the appropriate digital device. By their own means the operators would log all of their activities. Such as:
Ticket sales, their obliteration, record of time, location, etc.
This data would then be stored and managed by the clearing system. Which

in turn generates the problem of data security. A suitable clearing system must be able to guarantee that the privacy of all its users is preserved.
Also, in consideration of a competitive environment, the computation of the special economic evaluations must not benefit any one the operators over another.
The thesis focuses on the overall analysis of the clearing system, and aim to discover and identify various potential threats and weaknesses to its security.

My objectives in this thesis are:

1. To discuss the development of this project, initiated by Emilia-Romagna Regional Government.

2. To research and implement data mining tests on the database provided. The purpose is to discover patterns and information that may otherwise be hidden.

3. Using the information in these forms we will simulate various privacy attacks. These scenarios will include attacks carried out not only by external vectors, but also from the respective transport operators themselves.

This scheduling is also the result of discussions after a state of the art phase about data mining and the usage of a software called WEKA; which has been our choice to provide the data mining tests.
At the explicit request of E.R. government the project will be based on an already existing project developed by the business company StartRomagna [25], in 2009 like a fusion of three local business companies of three different areas (Rimini, Forli, Cesena), and now works in this little part of the E.R. Region.
The idea is to extend this model to the whole region.

Chapter 2

System structure

In this chapter we will look into the clearing system's structure. Particularly we will highlight some weaker points in its security, and further outline some developmental aspects.

2.1 Requirements

The clearing system must allow for the integration of relevant data from the operators under a single collection system. This allows appropriate access to involved operators.
Five fundamental security principles this system respects are:

Confidentiality: The restriction of the content. Insuring that information is accessible only to the correct users.

Availability: The service must always be available. With authorization and consistent verification of the supporting parties.

Authorization: Verify the consistence of the supports

Integrity: With the highest levels of assurance that the Data is authenticated and has not been modified.

Non-repudiation: The safeguard, non-repudiation, is a crucial aspect of all results.

System structure must be respected by all of its components. Including the message exchange.
Another important system requirement is that the reproducibility of its operative results will be preserved. This particular bureaucratic requirement is an essential element. The system must retain the ability to prove previous results of any particular economic evaluation.
The technique used in achieving this, is what we refer as a snapshot. The snapshots are an instantaneous representation, using a portion of the clearing system's database.
All the economic evaluations will be done through this snapshot. Every snapshot is saved and encrypted with its meta information along with its hash result in order to guarantee that it will not be modified or deleted in the future.
At a later point, we will further discuss these details in more depth.

2.2 Architecture

The clearing system is based on the distributed architecture composing its different parts. Operators are only responsible for the management and maintenance of their own data. This data is then needed for computation.
In a single processing centre it is collected in order to avoid disturbing production systems of the operators.
The processing centre represents the clearing system. The database is composed of the operators input data.
It is necessary for the computation of the information to use the system's back-end directly. Users have direct access to the front-end of the clearing system via interface to compute economic evaluations.
In figure 2.1 we can see the block structure of the system:

1. On the first level we have the production databases of the single opera-

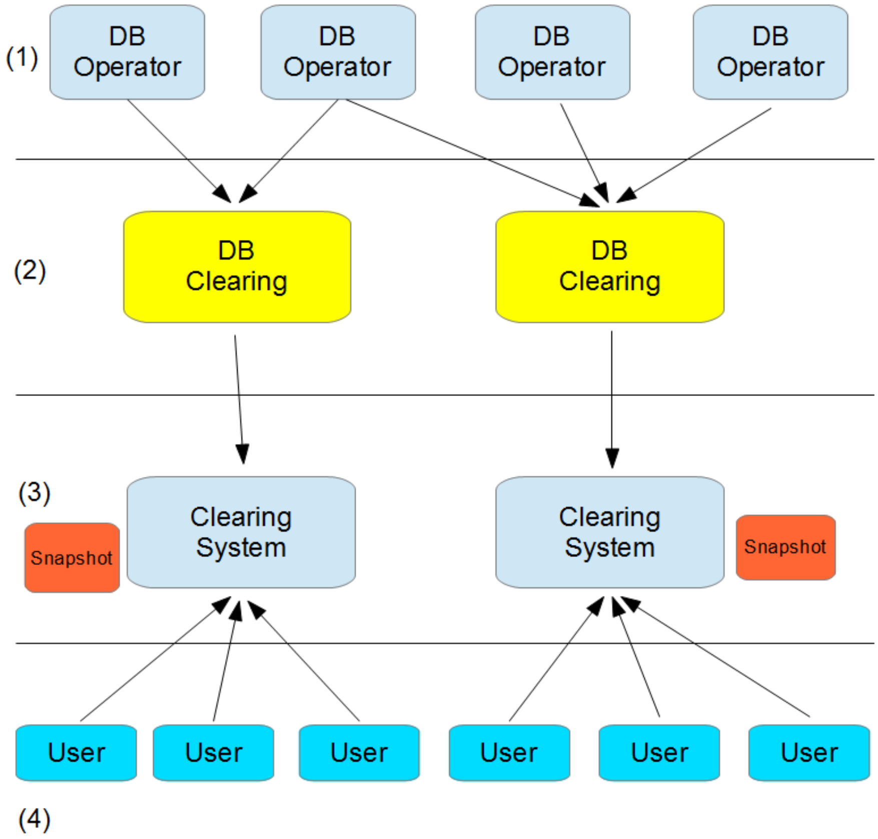


Figure 2.1: The clearing system architecture structured in 4 different levels

tors. This is where all the entries are constantly updated. The feeding system collects data periodically between all the databases of the operators concerned. The acquisition is performed using simple SQL query to the second level. At this stage it is important to ensure security during communication using the appropriate VPN.

2. At the second level we have the database clearing, this collects only the information needed clearing. The data collected is filtered at its origin to allow data that does not affect the system to exit the individual operator.

3. Here the back-end uses the database's data clearing to create the snap-

shots on a local database, with the data required to compute the clearing functions.

4. The final users access the front-end to compute economic evaluations.

2.2.1 Creation and management of the snapshots

To maintain the consistency of data, a snapshot, once it is created, it cannot be changed. This snapshot is immutable over time. Even if the basis of the original data clearing changes, the snapshot itself cannot.
To ensure this property remains together with the snapshots the following elements are saved and hashed:

- A generic description of the snapshot data for unique identification. And a description of the parameters used to create it.
- The specified users, roles and permissions. The origin of the created snapshot and the users or roles that have access and permissions.
- A time-stamp. Recording dates and times. Start and end.
- Data filters to limit the samples taken, only to the relevant portions of the overall data.
- For security, this being a description of the keys used for data encryption or digital signatures. This will digitally sign the snapshot at the time of its rescue.

Once the snapshot has been created, it is immediately saved as a file along with the safety systems required. Then the snapshot can be restored, or the stored data can be used on the file system.
The snapshots are saved and digitally signed to prevent unauthorized modification of the data.
Each snapshot maintains specific permissions to restrict access. This is to prevent unauthorized use of information.

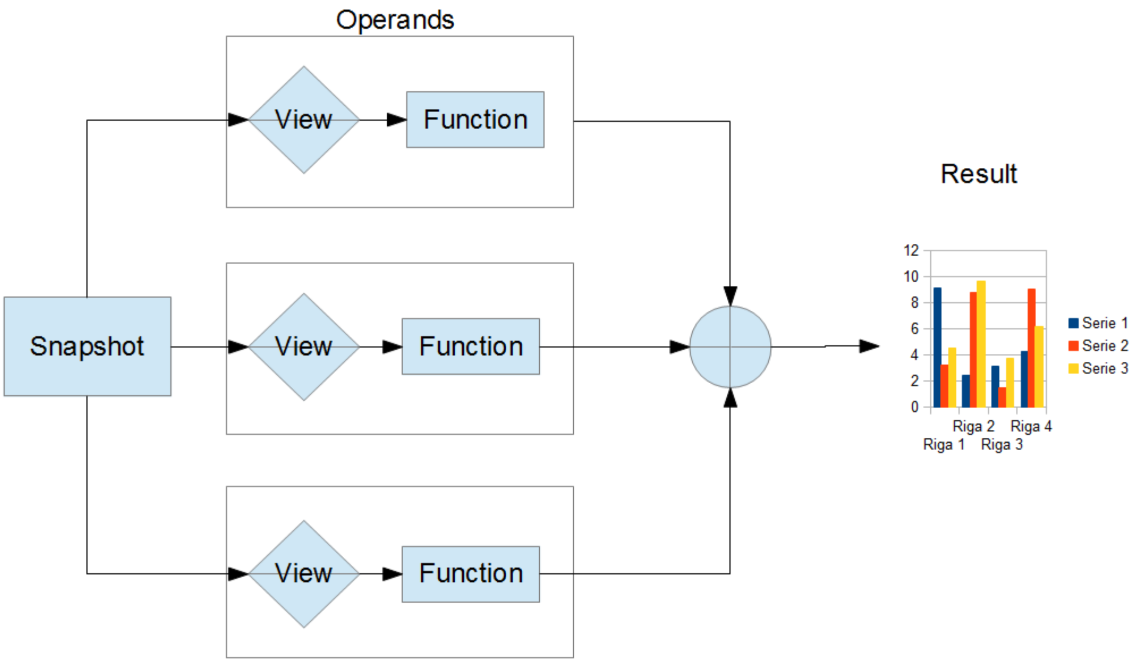


Figure 2.2: The structure of the algorithm of the clearing function

2.3 The clearing functions

Systems that create clearing functions are composed of different elements. These are administered both locally and globally to each and all operators. The purpose being that functions may be available to the different operators. This allows the investigation of the data, including the economic and clearing elements.

Of which may be sold, worn or obliterated.

The system of functions is made up of modules. These modules are to be programmed and then run, for the processing of clearing operations [2]. Each part or module represents a single computational step in which a system is

created.
It is this that allows the investigation of the clearing rules.
Generally, the creation of clearing operations framework consists of logical components shown in 2.2.
Each operation is broken down into different operands. The interpolation algorithm, in the different operands, produces the result of the clearing operation. The starting point of each operation is a snapshot that represents a portion of the imported data from the database operators.
Resulting from the clearing operation, the input and output data is represented by the snapshot. This is represented in tabular form.

The clearing operations consist of the following components: [2]

- Operand
- View
- Function

In any given clearing operation, the operand consists of a view and a function. The clearing operations are algorithmic interpolations of one or more operands. Each operand is characterized by the following components stored on the database containing the relevant permits.
The operands need a formal description. This must define the crucial details of input and output data, the permissions, the users and the roles. This formal description is necessary to enable the identification of data in the formalization of the clearing operations.
The view is a portion of the snapshot data. This filters the input data and returns a representation of the processed input data, functional to the operand.
It is defined by the input data, as the snapshot returns the result in tabular form. It is an SQL query to be applied by a snapshot, which returns a portion of the input data as a tabular representation. The view is used to filters the data source, this must be applied to a function. Which then calculates the

operand.

Because these are planned within the database administrator, there is no way to create or modify a view from the software interface. The functions of the operands, access the second view of the permission set. The view of an operand returns, in tabular form, a portion of the snapshot Data, to which it applies a function constituted by a series of algorithmic operations to the input data.

The function is an algorithm. When applied to the view it defines the output of the operand. The function must have the following characteristics:

- Access to local and global parameters of the system
- Must be stored within the database associated with an operand and its permissions.
- Must be programmed, with a programming language, capable of describing cycles and operations. Including conditional access as well as selectively viewed data.

Chapter 3

The Stimer's database description and the objective of our work

3.1 The database

After discussing the system's composition, we should now look into the chosen solution and the data itself. The database is at the fundamental heart of this work, particularly its composition.
The goal of the Stimer's Project is to track every instance of public transport. Obviously, the system will track all available information about the individual obliterations of each user.
Which can be seen in relation these contracts: [2]

CODE_TYPE_TITRE_CONTRAT Type of the contract (monthly, weekly, ecc)

NUM_CONTRAT Contract position on the support (1 for calypso ecc..)

EXPL_PROPRIETAIRE_CONTRAT The geographic contract type (regional, urban, ecc)

There are many indeed parameters regarding the relative information. These pertaining to the single use ticket holder are discussed in terms of a contract:

CODE_PROFIL_USAGER User profile (student, ecc)

DEV_PROFIL_USAGER Date user profile

NUM_SERIE_SUPPORT ID of the contract.

But most importantly, we have plenty of relevant information regarding the trace and exact locations of the user. Including the ticket, the validation zone and the purchasing zone etc. For example:

LIGNE_VALIDATION The Bus Ligne where the ticket has been validated.

NB_PASSAGERS_VALIDATION The number of possible transshipments.

NB_ZONE_VALIDATION The number of zone where the ticket it's valid.

ZONE_VALIDATION The zone where the ticket it's validated.

SS_LIGNE_VALIDATION The sub type of the zone.

SENS_VALIDATION The direction of the travel.

The Appendix will contain a detailed description of the database, including parameters and attributes.

3.2 Our work

In this section we want to explain what is the goal of our work and his motivation. We saw what would be the final goal of the Stimer's project; create a clearing system that incorporate all the public transport operators of the Emilia-Romagna Regional Government.

Naturally this system would have many obstacles, including: infrastructural,

hardware and various software problems.
As our topic of discussion is software, it is impossible to ignore the systemic security issues that arise. Considering bulk amounts of private data are being managed. The aspects of crucial importance would be:

- Are the means of this data management secure?
- Is it possible to break the security defences and violate the company's privacy?
- Should we place more focus on companies rather than single users?

As showed in the figure 3.1 systems such as these have two possible security attack vectors.
The first one is the classical intrusion of third party in which the data are subtracted from the primary database; but this aspect isn't our topic.
It is perhaps what would be regarded as the second most common attack vector that is the focus of our work. This is called an insider attack. Also referred to as an insider threat. This type of attack arises due to a malicious threat from inside the respective organizations themselves.
This raises the question:
Which means of attack shall we need to contend with?
It may be necessary begin with some particular assumptions.
In our case, the organization is a union of the public transport operators.
Within this unique system (Stimer), a single passenger, being a ticket holder.
Will be referred to as a single company.
The system periodically carries out economic evaluations based on data. This data is derived from the individual operators and then distributed corresponding to the profits. Importantly, one should note that the system realizes these evaluations and not the companies. This is because the individual companies would hold little information in relation to competitors. Of course subsequent information is encoded within the parameters and attributes of the database.

This means that the companies cannot know, for example, which is the contract type most sold in a certain area, which is instead the most used and so on; information of this nature would be very useful in order to compete. But now, what if a single company try to attack his competitors of the same system?

As attack we get on is possible that single operators, with particular analysis, are able to obtain sensitive information on the work of other companies?

Finally we shift to the primary focus of our work.

In this thesis, we aim to demonstrate whether it is possible that by using this system a company can obtain sensitive information revealing the actions of its competitors, and in turn use this information fraudulently.

Consequently, the next question we must answer is:

How should we proceed?

Motivated by this subject, in the next chapter we will operate a data mining analysis. With the available data, our analysis will involve tests using the famous data-mining tool called WEKA. Which will be used to provide information that otherwise should be hidden.

After that we will offer explanations as to how this information can be used. Thus, creating these possible scenarios. Following this we will discuss feasible preventative measures that could be employed for these kinds of attacks.

Next, we should look at our chosen strategies.

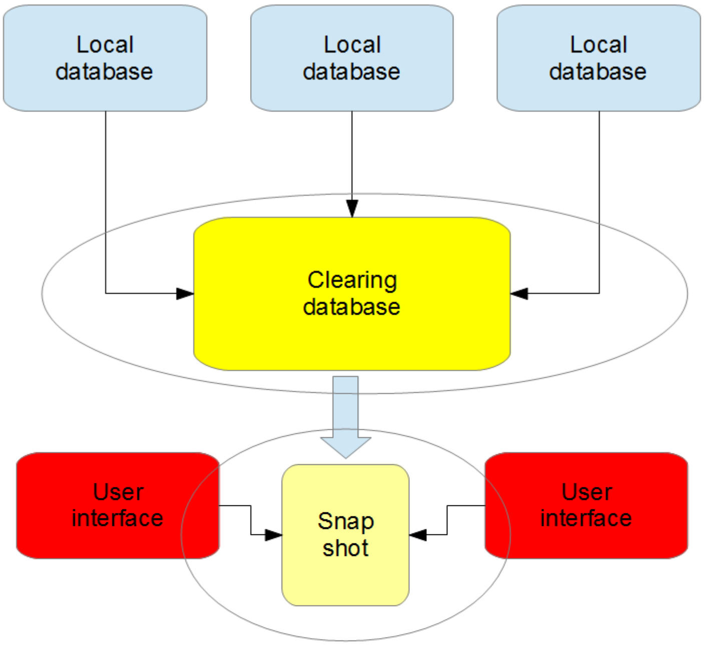


Figure 3.1: Circled, the possible attack vector

Chapter 4

Data Mining in the literature

In the last years we have seen the advent of big data, with the exponential growing of gigabytes of data that were shared all around the world.
Firms, companies and even public administration compete each other in order to make the largest possible amount of user information.
The underlining belief now, is that one can know anything, learn any aspect, and discover any detail of our lives through these digital footprints and various profiles.
But what can happen when, thankfully, digital privacy is respected?
Or, when we simply cannot access certain information?
A solution to obtaining such information can be data mining.
Data-mining techniques have attracted a great deal of interest from the information industry. In recent years this has also captured the wider interest of a society in general. This is due to the imminent need to transform these huge amounts of data into useful information. With the results of this often leading to far greater knowledge of any given topic.
Many of these gains are immediately applicable, ranging from market analysis, fraud detection, customer retention, production controls and science exploration.
A logical question to ask is: what exactly is data mining? [18] [21]
Data mining is defined as, the process of discovering patterns in data and,

extracting knowledge from large amounts of data. The process must be automatic or, more usually, semi-automatic.

But what can Data Mining Do? As [3] explained very well although data mining is still in its infancy, companies in a wide range of industries including retail, finance, heath care, manufacturing transportation, and aerospace are already using data mining tools and techniques to take advantage of historical data.

Using technologies that rely on pattern recognition along with statistical and mathematical techniques. The data is processed, essentially sifting through warehoused information.

This data mining helps analysts recognize significant facts, relationships, trends, patterns, exceptions and anomalies that may otherwise have gone unnoticed.

For businesses, data mining is used to discover patterns and relationships in the data. This then helps to make more informed business decisions.

Data mining can help spot sales trends. Thus, being used to develop smarter marketing campaigns and improve accuracy when predicting customer loyalty. Specific uses of data mining include[1]:

Market segmentation Indication and correlation of the common characteristics of customers who buy the same products from a company.

Customer churn This predicts which customers are likely to leave your company and move to a competitor.

Fraud detection Identification of the transactions that are most likely to be fraudulent.

Direct marketing Specifying which prospects should be included in a mailing list to obtain the highest response rate.

Interactive marketing Used for predictions of what individuals accessing a web site are more likely to be interested in viewing.

Market basket analysis This aims to increase the understanding of products and services that are commonly purchased together.

Trend analysis Reveals the difference between a typical customer, within monthly, quarterly and yearly cycle.

Understanding of data mining in this way directs us to the next question: How could the use of these techniques, or similar, become a security threat to our system?

Firstly, by defining the patterns of hidden information between these companies. Following this, we can decipher the relevant information and devise methods to penetrate the security walls. Making it simple to violate the privacy of user and/or companies. As we've previously outlined, data mining is a process. This leads us to ask:
What the relative steps? In fact, the process of data mining will be part of a wider process.
Being, data processing; in which we find a pre-process phase, this will include:

1. Data cleaning: This removes incorrect and null values, referred to noise.

2. Data integration: In which we can unify multiple sources or complete missing data.

3. Data selection: This is where only the relevant values of interesting are selected in accordance the objectives of our analysis.

4. Data transformation: Data may be transformed or consolidated into forms that are appropriate for data mining algorithms. Doing so by performing summary or aggregation operations. Including discretization, etc.

Next is a test phase. This includes:

1. Data mining: Being the heart of our analysis. Applying intelligent methods and algorithms to then extract data patterns.

2. Pattern evaluation: Involving the interpretation of these results, in order to justify the measurements.

3. Knowledge presentation: Techniques to represent this knowledge, including visualizations can then present the mined data.

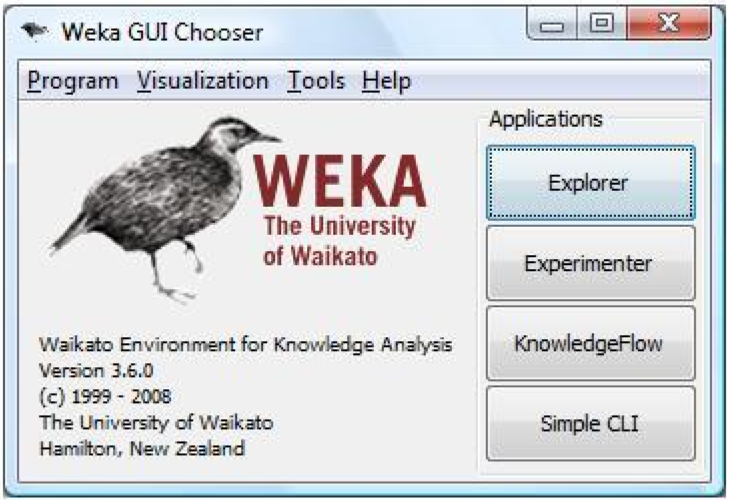
Previously we have been discussing patterns.
So, let's clarify: What are patterns? During our data mining analysis we have seen that using probabilistic methods and algorithms can determine, or even better predict, the values, parameters and trends using only the subset values. In which case, such patterns are the parameters of prediction, along with association rules in the prediction of trends. In all cases, high quality coefficient would be required.
With reference to the previous chapter:
How was the data mining process used? In this phase we will illustrate how we have discovered patterns in the Stimer's Database system for public transport operators.
In particular, will describe how such a pattern can be used to derive the information that theoretically should be completely hidden. Of course, this then could be employed as a means to violate the security of the system.

Chapter 5

WEKA

In this chapter we begin our analysis.
As mentioned previously we choose a very powerful software called WEKA. [16] WEKA is an open source software issued under the GNU General Public License.
"WEKA" stands for the Waikato Environment for Knowledge Analysis, which was developed at the University of Waikato in New Zealand. WEKA like every open-source project it's freely downloaded, and it's written in Java, so it could runs on almost every platform.
To further specify: what is WEKA?
WEKA is a collection of machine learning algorithms for data mining tasks written in Java.
The algorithms can either be applied directly to a dataset or called from your own Java code. WEKA contains specific tools for every data mining step that we have discussed; tools for data pre-processing, classification, regression, clustering, association rules, and visualization.
WEKA is used for research, education, and applications. Among the various choices at our disposal, we focused on WEKA because gathers a comprehensive set of data pre-processing tools, learning algorithms and evaluation methods, graphical user interfaces (incl. data visualization) and environment for comparing learning algorithms. [5]

In order to better understand the usage of WEKA we will see a practical example of the WEKA's structure. The Gui shows in 5.1 the possible applications that we have:



Waikato Environment for Knowledge Analysis
Version 3.6.0
(c) 1999 - 2008
The University of Waikato
Hamilton, New Zealand

Figure 5.1: The WEKA GUI Interface

The buttons can be used to start the following applications:

Explorer An environment for exploring data with WEKA (Almost all the work was done with this application)

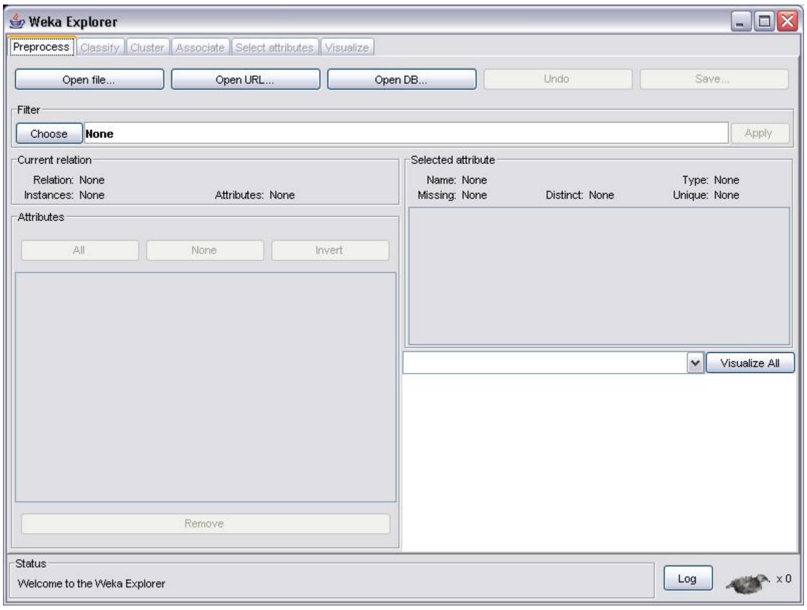
Experimenter An environment for performing experiments and conducting statistical tests between learning schemes and to compare the results of the algorithms

KnowledgeFlow This environment supports essentially the same functions as the Explorer but with a drag-and-drop interface.

SimpleCLI Provides a simple command-line interface that allows direct execution of WEKA commands for operating systems that do not provide their own command line interface.

5.1 Explorer

The Explorer application is the most important. It has been a center from which to base our work. And it is where all the WEKA functions have been developed.
As you can see on the picture 5.2 the explorer GUI is divided in six tabs: PreProcess, Classify, Cluster, Associate, Select attributes, Visualize.



Filter: None

Relation: None
Instances: None
Attributes: None

Name: None
Missing: None
Distinct: None
Type: None
Unique: None

Welcome to the Weka Explorer

Figure 5.2: The WEKA GUI Explorer Interface

5.1.1 PreProcessing

Firstly, we must of course upload our dataset.
At this stage we have multiple options. These are:

1. To open a file containing our data the description. This could be a csv file, an arff (the WEKA data type) or even sql.

2. Open from a URL where the resource data is located.

3. Open a database that reads the date directly from the database.

Once our data has been loaded we can start the pre-process phase using the pre-process panel. This shows a variety of information.
As shown in picture 5.3 there is a currently loaded data box that has three entries:

Relation This names the relation, as given in the file it was loaded from. Filters (described below) modify the name of a relation.

Instances The number of instances (data points/records) in the data.

Attributes The number of attributes (features) in the data.

Of course in this step we should work with the attributes, modifying them with all the techniques described in the next chapter.
Below the Current relation box there is a box titled Attributes, where can find a list of the attributes:

No A number that identifies the attribute in the order they are specified in the data file.

Selection tick boxes These allow you to select which attributes are present in the relation.

Name The name of the attribute, as declared in the data file.

When you click on different rows in the list of attributes, the fields change in the box to the right titled Selected attribute. This box displays the characteristics of the currently selected attribute in the list:

Name The name of the attribute, the same as that given in the attribute list.

Type The type of attribute, most commonly Nominal or Numeric.

Missing The number (and percentage) of instances in the data for which this attribute is missing (unspecified).

Distinct The number of different values that the data contains for this attribute.

Unique The number (and percentage) of instances in the data having a value for this attribute that no other instances have.

Below the statistics is a list. This displays more information about the values stored in this attribute, differing depending on their type. If the attribute is nominal, the list consists of each possible value for the attribute along with the number of instances that have that value.
If the attribute is numeric the list will give four statistics describing the distribution of values in the data. These are: The minimum, maximum, mean and standard deviation.
And below these statistics there is a colored histogram, color-coded according to the attribute. Which is chosen as the class, using the box above the histogram. This box will bring up a drop-down list of available selections when clicked.
Finally, after pressing the visualize all button, histograms for all the attributes in the data are shown in a separately unless, as in our case, there are attributes with too many values to show.
Returning to the attribute list in the picture 5.3 it should be noticed that there are three buttons that can also be used to change the selection:

All All the boxes become ticked.

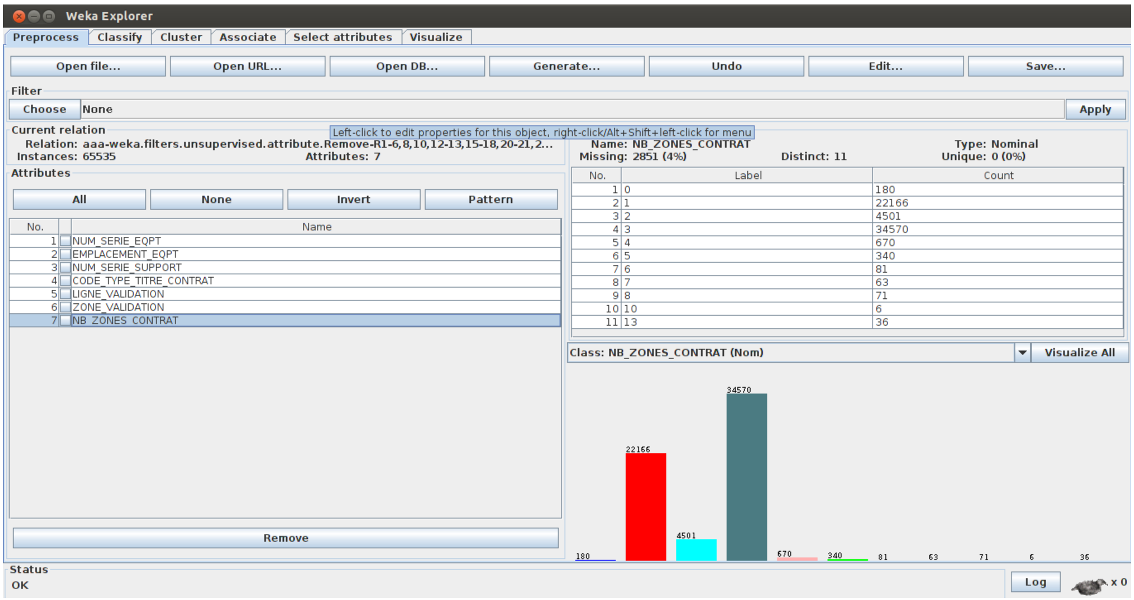
None All the boxes are cleared (un-ticked).

Invert This changes ticked boxes to become un-ticked boxes. And vice versa.

Once the desired attributes have been selected, they can be removed by clicking the Remove button below the list of attributes.

The last important option of the PreProcess application is that we can apply PreProcess algorithms by simply choosing one from the filter bar; choose

the attributes and click to apply on the right corner, and after that the Preprocess panel will then show the transformed data.
Here there is an important observation to make. It is possible to set up some parameters and the value of this algorithm. This is done on the filter bar after choosing the filter algorithm.
It is possible to see an example of the numeric to nominal filter shown here in picture 5.4.



Filter: None

Current relation
Relation: aaa-weka.filters.unsupervised.attribute.Remove-R1-6,8,10,12-13,15-18,20-21,2...
Instances: 65535 Attributes: 7

Attributes

No.		Name
1	□	NUM_SERIE_EQPT
2	□	EMPLACEMENT_EQPT
3	□	NUM_SERIE_SUPPORT
4	□	CODE_TYPE_TITRE_CONTRAT
5	□	LIGNE_VALIDATION
6	□	ZONE_VALIDATION
7	□	NB_ZONES_CONTRAT

Name: NB_ZONES_CONTRAT Type: Nominal
Missing: 2851 (4%) Distinct: 11 Unique: 0 (0%)

No.	Label	Count
1	0	180
2	1	22166
3	2	4501
4	3	34570
5	4	670
6	5	340
7	6	81
8	7	63
9	8	71
10	10	6
11	13	36

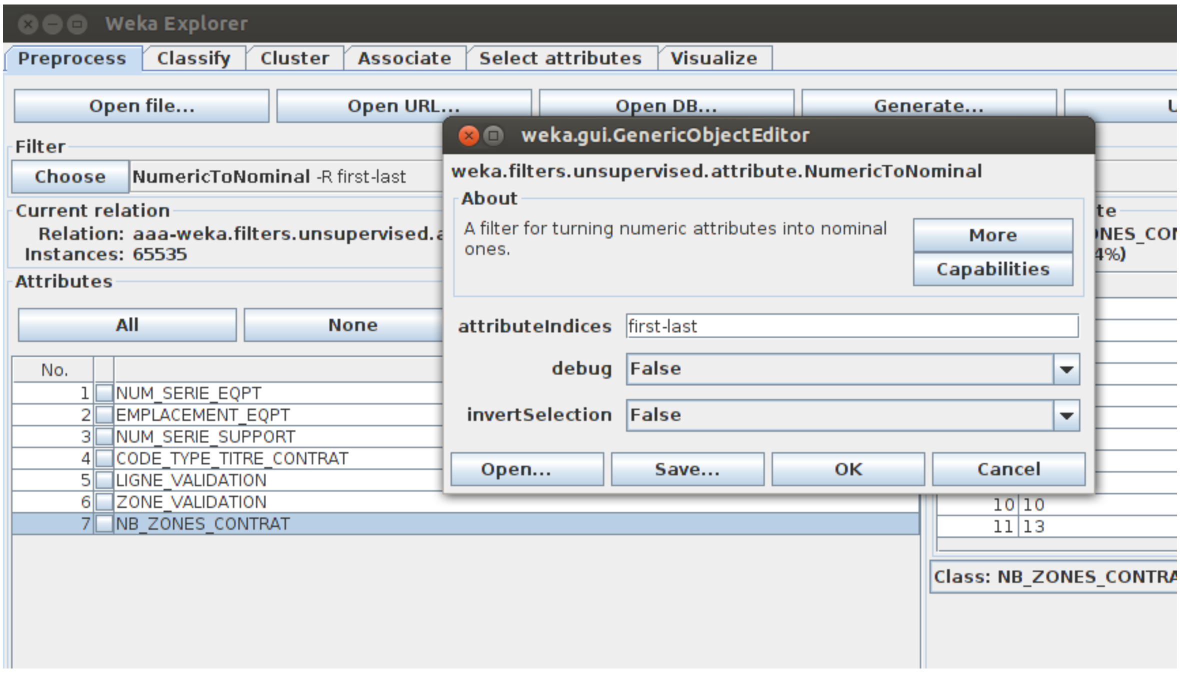
Class: NB_ZONES_CONTRAT (Nom)

Status
OK

Figure 5.3: The WEKA PreProcess tab with the data loaded

5.1.2 Classify

The second, and most important, application that we used is the classify tab; the one on which we will do some experiments with the data mining algorithms.
At the top of the classify section we have the Classifier box. This box has a text field that names the currently selected classifier and its options. When selecting a classifier you have also the opportunity to choose many filters or different parameters that will be used in the tests. The result of applying



NumericToNominal -R first-last

Current relation
Relation: aaa-weka.filters.unsupervised.a
Instances: 65535

No.		
1	□	NUM_SERIE_EQPT
2	□	EMPLACEMENT_EQPT
3	□	NUM_SERIE_SUPPORT
4	□	CODE_TYPE_TITRE_CONTRAT
5	□	LIGNE_VALIDATION
6	□	ZONE_VALIDATION
7	□	NB_ZONES_CONTRAT

weka.filters.unsupervised.attribute.NumericToNominal

About
A filter for turning numeric attributes into nominal ones.

attributeIndices first-last
debug False
invertSelection False

10	10
11	13

Class: NB_ZONES_CONTRA

Figure 5.4: The filter Numeric to Nominal application on the dataset

the chosen classifier will be tested according to the options that are set in the Test options box. We have three Different choices with which to test our data with the classifier.
These choices are the following:

Use training set The classifier is evaluated on how well it predicts the class of the instances it was trained on.

Supplied test set The classifier is evaluated on how well it predicts the class of a set of instances loaded from a file.

Cross-validation The classifier is evaluated by cross-validation, using the number of folds that are entered in the Folds text field.

Percentage split The classifier is evaluated on how well it predicts a certain percentage of the data which is held out for testing.

The amount of data in the percentage is split into the number of folds, and so on. Here we can also see there is an influential parameter for the results.

Once we know how to launch the algorithms, we need to know what we will search for.
The classifiers algorithms implemented in WEKA are designed to predict a single "class" attribute, that is to say a single parameter, which is the target for prediction.
In terms of prediction, there are numerous differences. This is because some classifiers only learn nominal classes, whereas, others can only learn numeric classes. An example is linear regression. Then there are others that can learn both.
By default, the class is taken to be the last attribute of the data. But if you want to set the classifier in order to predict a different attribute, they may be chosen from the box below the test options box.5.5

Once the classifier, the test options and class have all been set. Finally the learning process can start, by simply clicking on the Start button.
The classifier output area to the right of the display is filled with text describing the results of training and testing. A new entry appears in the result list box. We can look at the result list below. But first we try to understand the output text.
The text in the classifier output area allows you to browse the results. The most important thing to notice here is that the text output is divided in several parts. Each one will represent particular information:

- Run information. A list giving the learning scheme options. Including the relation name, the instances, the attributes and test mode that were involved in the process.
- Classifier model (full training set). This is a textual representation of the classification model that was produced from the full training data.
- The results of the chosen test.
- Summary. This is a list of statistics summarizing how accurately the classifier was able to predict the true class of instances under the chosen test mode.

- Detailed Accuracy By Class. A more detailed per-class breakdown of the classifier's prediction accuracy.
- Confusion Matrix. This will show how many instances have been assigned to each class.

Finally we should be able to have a good representation, and storage of the results. As there are many different choices [20] picture 5.6:

View in main window This shows the output in the main window (left-clicking the entry).

View in separate window This opens a new independent window for viewing the results.

Save result buffer This will bring up a dialog allowing you to save a text file containing the textual output.

Load model This loads a pre-trained model object from a binary file.

Save model This saves a model object to a binary file. Objects are saved in Java 'serialized object' form.

Re-evaluate model on current test set This takes the model that has been built and tests its performance on the data set that has been specified with the set button under the supplied test set option.

Visualize classifier errors This opens a visualization window that plots the results of the classification. The correctly classified instances are represented as crosses. Whereas, incorrectly classified ones are shown as squares.

Visualize tree or Visualize graph This opens a graphical representation classifier models structure, when possible (i.e. for decision trees or Bayesian networks). The graph visualization option only appears if a Bayesian network classifier has been built. In the tree visualizer you can bring up a menu by right-clicking a blank area. Then, pan around by dragging the mouse seeing the training instances at each node, by clicking on it. CTRL-clicking zooms the view out, while SHIFT- dragging a box zooms the view in. Basically, the graph visualizer should be self-explanatory.

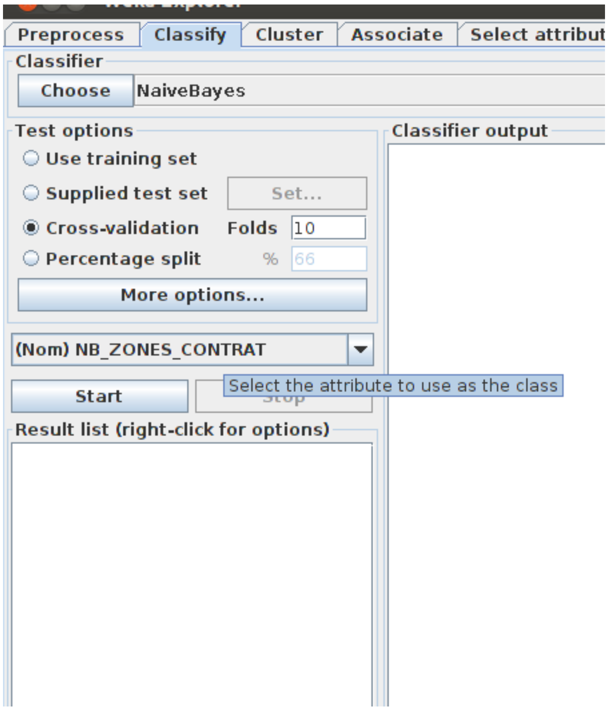
Visualize margin curve This will generate a plot illustrating the prediction margin. The margin is defined as the difference between its probabilities, as predicted for the actual class, and the other class's highest probability prediction.
For example:
Boosting algorithms may achieve a better performance on test data by increasing the margins on the training data.

Visualize threshold curve This generates a plot illustrating the tradeoffs in a prediction that is obtained by varying the threshold value between the classes.
For example:
With a default threshold value of 0.5. The predicted probability of a positive must be greater than 0.5, for this prediction to be positive. The plot can be used to visualize the precision/recall tradeoffs. For ROC curve analysis (true positive rate opposed to false positive rate), and for other types of curves.

Visualize cost curve The plot that is generated gives an explicit representation of the expected cost.

5.1.3 Clustering

Looking to the second tab, we find the Cluster application.
The cluster mode box is used to choose the cluster, and the evaluation of the results.
The first three options are all used for classification. These are: Use training



NaiveBayes

○ Use training set

○ Supplied test set

● Cross-validation Folds 10

○ Percentage split % 66

(Nom) NB_ZONES_CONTRAT

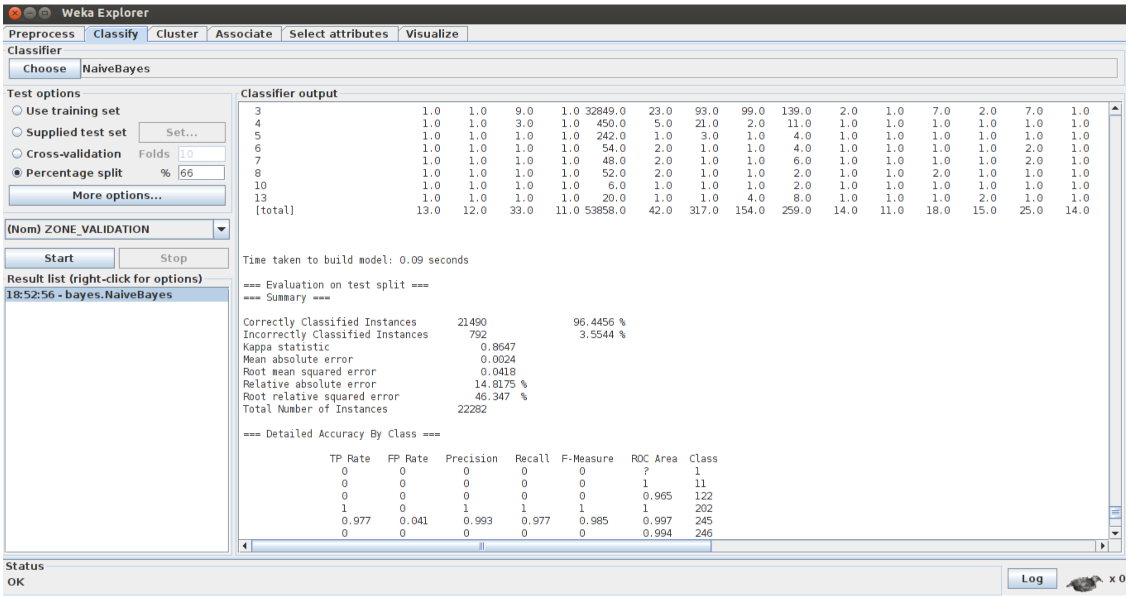
Figure 5.5: The possible options for the dataset on the classification tab

set, Supplied test set and Percentage split.

In these, the data is assigned to clusters instead of trying to predict a specific class.

The fourth mode is the Classes to clusters evaluation. This makes a comparative assessment of how the chosen clusters match the pre-assigned data class.

There is an additional option in the Cluster mode box. This is the Store clusters for visualization tick box. This determines whether or not it will be possible to visualize the clusters once training is complete. When dealing with these large datasets, computational memory and processing power can become a real problem. To manage this it may be helpful to disable some options. Unfortunately, this can often be the case. This being so, some attributes in the data should be ignored when clustering. It is the ignore attributes button that will open a small window allowing for the selection of which attributes to ignore.



Classifier: NaiveBayes

Test options:
○ Use training set
○ Supplied test set
○ Cross-validation Folds 10
● Percentage split % 66

(Nom) ZONE_VALIDATION

Result list: 18:52:56 - bayes.NaiveBayes

Status: OK

Classifier output

```
  3                        1.0    1.0    9.0    1.0 32849.0   23.0   93.0   99.0  139.0    2.0    1.0    7.0    2.0    7.0    1.0
  4                        1.0    1.0    3.0    1.0   450.0    5.0   21.0    2.0   11.0    1.0    1.0    1.0    1.0    1.0    1.0
  5                        1.0    1.0    1.0    1.0   242.0    1.0    3.0    1.0    4.0    1.0    1.0    1.0    1.0    1.0    1.0
  6                        1.0    1.0    1.0    1.0    54.0    2.0    1.0    1.0    4.0    1.0    1.0    1.0    1.0    2.0    1.0
  7                        1.0    1.0    1.0    1.0    48.0    2.0    1.0    1.0    6.0    1.0    1.0    1.0    1.0    2.0    1.0
  8                        1.0    1.0    1.0    1.0    52.0    2.0    1.0    1.0    2.0    1.0    1.0    2.0    1.0    1.0    1.0
  10                       1.0    1.0    1.0    1.0     6.0    1.0    1.0    1.0    2.0    1.0    1.0    1.0    1.0    1.0    1.0
  13                       1.0    1.0    1.0    1.0    20.0    1.0    1.0    4.0    8.0    1.0    1.0    1.0    2.0    1.0    1.0
  [total]                 13.0   12.0   33.0   11.0 53858.0   42.0  317.0  154.0  259.0   14.0   11.0   18.0   15.0   25.0   14.0



Time taken to build model: 0.09 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances       21490               96.4456 %
Incorrectly Classified Instances       792                3.5544 %
Kappa statistic                          0.8647
Mean absolute error                      0.0024
Root mean squared error                  0.0418
Relative absolute error                 14.8175 %
Root relative squared error             46.347  %
Total Number of Instances            22282

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0         0         0          0         0          ?        1
                 0         0         0          0         0          1        11
                 0         0         0          0         0          0.965    122
                 1         0         1          1         1          1        202
                 0.977     0.041     0.993      0.977     0.985      0.997    245
                 0         0         0          0         0          0.994    246
```

Figure 5.6: The Classification output

Why should we ignore some attributes? The various motivations can be:

1. The number of instance are so many that it is impossible to make cluster analysis on the whole dataset; this is why someone must be eliminated.

2. There is not enough distance in the clustering to determine the clusters of interest. This being so, we must restrict our dataset.

3. There are many attributes that are not of interest, due to redundancies.

Once the cluster algorithm is selected, we can enter the correspondent parameters. Next is the percentage split (the dataset) and then launch the application. It is then time to manage these results.
The cluster section, like the classify section, has Start/Stop buttons. It has a result text area and a result list. These all behave like their classification counterparts.
Right-clicking an entry in the result list brings up a similar menu, except that it shows only two visualization options: Visualize cluster assignments

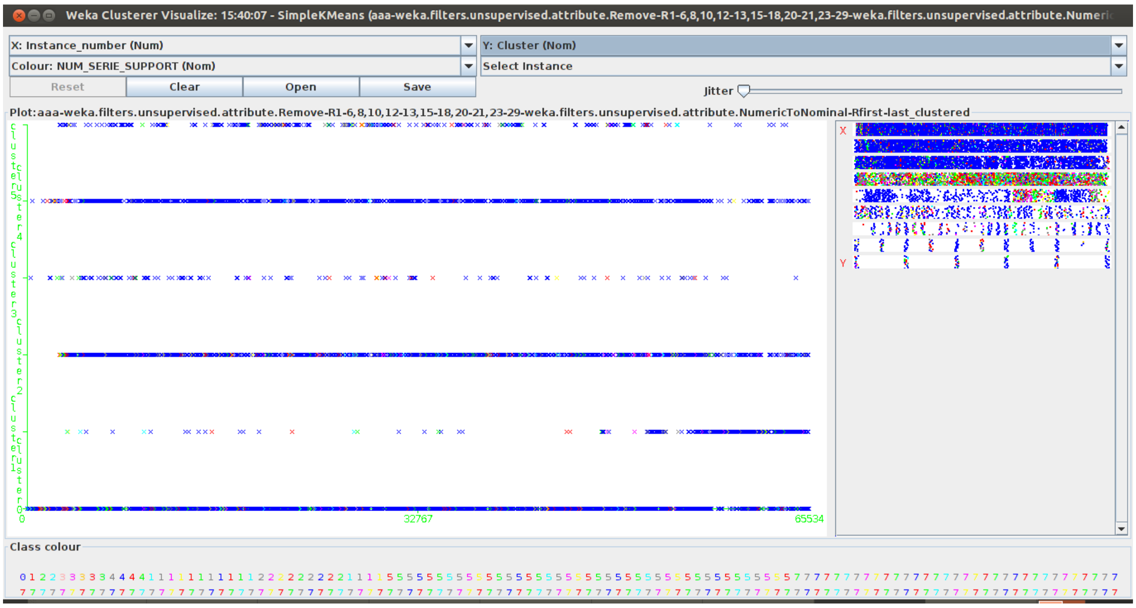
and visualize tree which opens other windows all where the cluster graphics are shown.5.7

5.1.4 Associating

The second panel of the explorer application is the association panel. This panel contains schemes for learning association rules. These learners are chosen and configured in the same way as the clusters, filters, and classifiers in the other panels.
As we have previously noted, we only need to choose the association algorithms that we want to use, and sets the parameters. Then finally the results, in particular the association evidence is shown on the right panel of this tab. The rules are easy to understand. To better understand this, we could draw a mental picture of this being a series of nested statements. Resulting with a certain correctness coefficient.5.8

5.2 Experimenter

Lastly, the second application we have used is called the experimenter. Once we have completed our numerous analyses, and compared the results with the algorithms. The experimenter is very useful when comparing the efficiency of different algorithms.
The experimenter operates by combining the launch of two different classifiers, directly outputting the result of its comparison.
As displayed, on image 5.9, once the dataset is loaded and the experiment type is specified (cross fold, percentage split), we can set the algorithms that are to be compared. These are added singularly in the relevant box in the tab.
We can now save this model to launch the analysis. In the relevant tab the file containing our configuration can be loaded. Then the algorithm comparison process can start. In the right box the results are shown, each algorithm is evaluated with a efficiency coefficient, here we can see the differences.



Weka Clusterer Visualize: 15:40:07 - SimpleKMeans (aaa-weka.filters.unsupervised.attribute.Remove-R1-6,8,10,12-13,15-18,20-21,23-29-weka.filters.unsupervised.attribute.Numeri

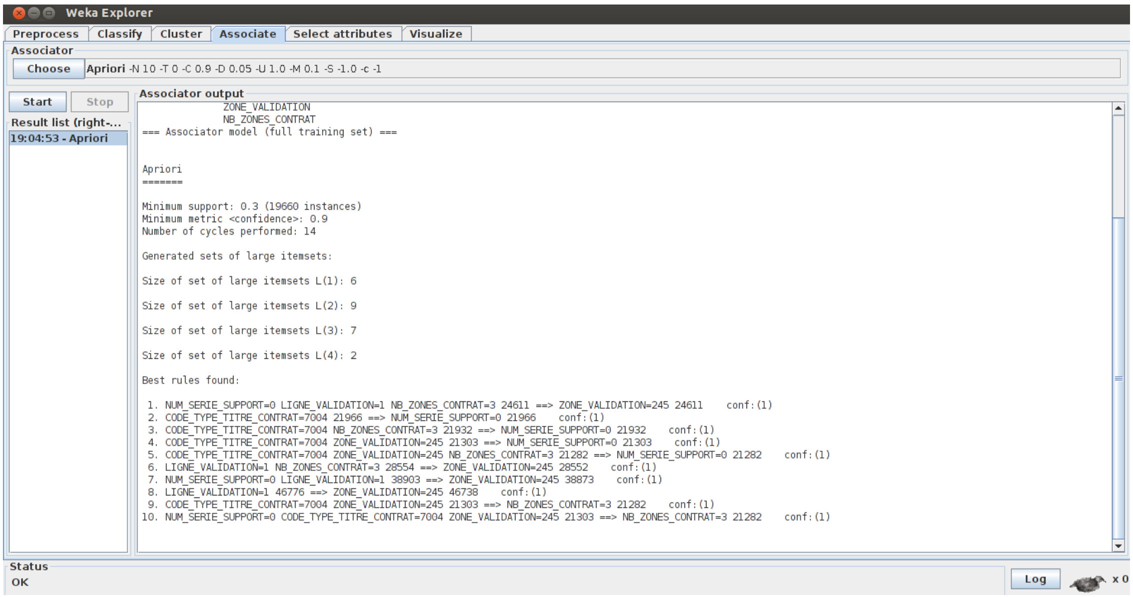
X: Instance_number (Num)

Y: Cluster (Nom)

Colour: NUM_SERIE_SUPPORT (Nom)

Plot:aaa-weka.filters.unsupervised.attribute.Remove-R1-6,8,10,12-13,15-18,20-21,23-29-weka.filters.unsupervised.attribute.NumericToNominal-Rfirst-last_clustered

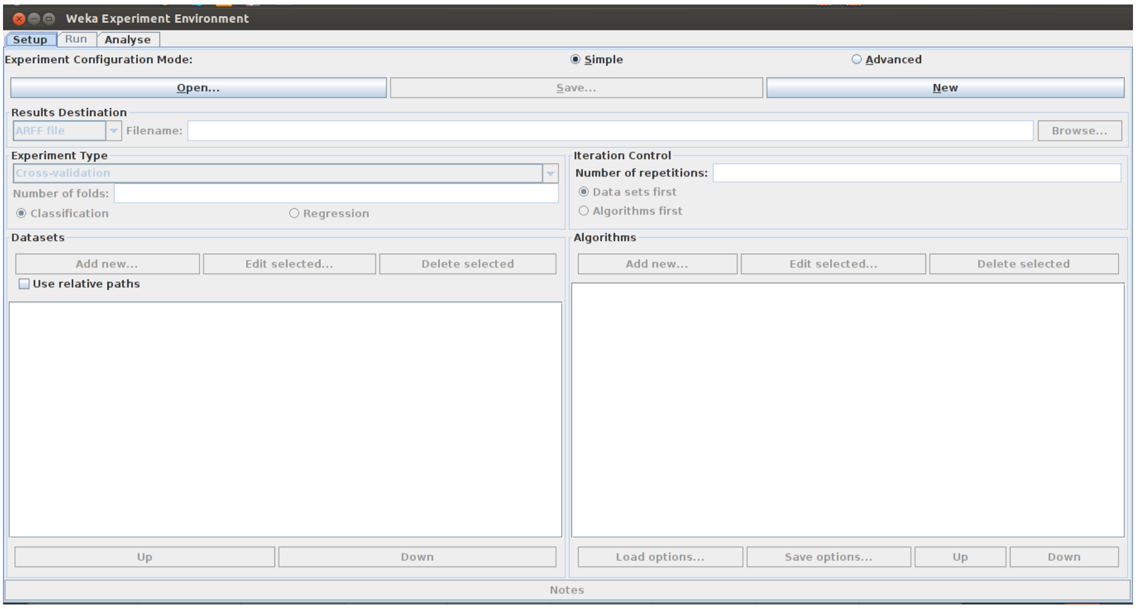
Figure 5.7: The Cluster analysis



Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

```
                 ZONE_VALIDATION
                 NB_ZONES_CONTRAT
=== Associator model (full training set) ===


Apriori
=======

Minimum support: 0.3 (19660 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 14

Generated sets of large itemsets:

Size of set of large itemsets L(1): 6

Size of set of large itemsets L(2): 9

Size of set of large itemsets L(3): 7

Size of set of large itemsets L(4): 2

Best rules found:

 1. NUM_SERIE_SUPPORT=0 LIGNE_VALIDATION=1 NB_ZONES_CONTRAT=3 24611 ==> ZONE_VALIDATION=245 24611    conf:(1)
 2. CODE_TYPE_TITRE_CONTRAT=7004 21966 ==> NUM_SERIE_SUPPORT=0 21966    conf:(1)
 3. CODE_TYPE_TITRE_CONTRAT=7004 NB_ZONES_CONTRAT=3 21932 ==> NUM_SERIE_SUPPORT=0 21932    conf:(1)
 4. CODE_TYPE_TITRE_CONTRAT=7004 ZONE_VALIDATION=245 21303 ==> NUM_SERIE_SUPPORT=0 21303    conf:(1)
 5. CODE_TYPE_TITRE_CONTRAT=7004 ZONE_VALIDATION=245 NB_ZONES_CONTRAT=3 21282 ==> NUM_SERIE_SUPPORT=0 21282    conf:(1)
 6. LIGNE_VALIDATION=1 NB_ZONES_CONTRAT=3 28554 ==> ZONE_VALIDATION=245 28552    conf:(1)
 7. NUM_SERIE_SUPPORT=0 LIGNE_VALIDATION=1 38903 ==> ZONE_VALIDATION=245 38873    conf:(1)
 8. LIGNE_VALIDATION=1 46776 ==> ZONE_VALIDATION=245 46738    conf:(1)
 9. CODE_TYPE_TITRE_CONTRAT=7004 ZONE_VALIDATION=245 21303 ==> NB_ZONES_CONTRAT=3 21282    conf:(1)
10. NUM_SERIE_SUPPORT=0 CODE_TYPE_TITRE_CONTRAT=7004 ZONE_VALIDATION=245 21303 ==> NB_ZONES_CONTRAT=3 21282    conf:(1)
```

Figure 5.8: The Association rules



Experiment Configuration Mode: ● Simple ○ Advanced

Results Destination: ARFF file

Experiment Type: Cross-validation

● Classification ○ Regression

Iteration Control: ● Data sets first ○ Algorithms first

Figure 5.9: The Experimenter tool

Chapter 6

Pre-Process phase

In this chapter begins the description of work that is central to the thesis. Given that we have our database and our data WEKA mining software.
How should we then wish to proceed? To initiate our analysis we first start the pre-processing of data. Preprocessing phase consists of manipulation and transformation of data before to applying the data mining algorithms. This transformation is made using several techniques that will be applied either manually or directly on the dataset, or, through some WEKA filters.
Why is this step necessary? Fundamentally there are two reasons:

1. Firstly because our data may be 'dirty'. This can mean that the data is incomplete, inconsistent or incorrect. This then will need the cleaning step. [10]

2. Secondly, we will apply the data-mining algorithm. As we have a huge amount of data and the algorithms are quite complex. In order to improve the performance of the algorithm and, thus, our result, we need to implement some algorithmic pre-process transformation such as discretization or categorization. These algorithms act to rearrange the data with specific criteria. We can illustrate that WEKA can help us in this.

6.1 Elimination

The first, and rather simple analysis shows us that some of the attributes do not have a useful range of possible values for us. As seen in picture 6.1 there are 23 further examples of this, although. These do have a broad range of possible values. Typically over 100, these represent only few of them (less than 2). When the values are not useful we need to create association rules. Attributes with too fewer values can be indicative that we do not have enough differences. More than 99% of the attributes instances are represented only by a single value. We have decided to completely eliminate these attributes.

6.2 Missing

Some of the attributes have shown large proportions of missing data. This problem can be managed in two different ways, in turn leading to two different interpretations of the results.

1. We can add a category that incorporates all the empty values. Using this as a new nominal value.

2. We can replace the empty values with a possible trend that statistically reflects the values already present. Done in such a way that the general distribution is not altered.

In the first solution shown we do not change the values. Since it is as if we were creating a new possible nominal value during the creation of the association rules. We could have false positives in many of these cases.
Regarding the second solution, we can see that by adjusting the values and introducing a coefficient error during the classification and association process can, in some cases, be tolerable.
We found it is suitable to use this solution without large quantities of empty values, when there were, we had decided to use the first solution.

6.3 Incorrect

Using the same principle even in a case where we had detected the incongruence from certain values. In particular, not corresponding to the type of parameter. Here we decided to use the previous techniques.

6.4 Categorization

Another important technique that we used is called categorization.[23]
Categorization is a part of cognition and is fundamental to the process of comprehension. Categories play a central role in perceptions, learning, communication and thinking. Categorization is used to group objects together into classes, based on similarities. These classes are called categories or concepts.

Group values under different categories are very important. Without these, every object or incidence would appear inimitable. Hence, there would be no way to predict new unknown objects. Thus, categories can be seen as a product of interactions, and can help us in the performance result.

There are many techniques of categorization. Most of these are aimed to build knowledge.

What we have used is the classical sense, is that claims that have categories are discrete entities. These are characterized by a set of properties and shared by their members. These properties assume the established conditions. These are necessary and need to be succinct in order to capture the meaning of a category.

Within these conditions our example is divided into some attributes such as the Code Type Titre Contract attribute. This division could be compared to a triage process, depending on three macro categories. Which are already described in [2]; where the contract are considered as:

- Contratto Ordinario Corsa Semplice
- Contratto Ordinario Multicorsa

- Carta Valore Corsa Semplice
- Carta Valore Multicorsa

6.5 Discretization

Some of the algorithms that we used to create data mining models in WEKA, required specific content types in order to function correctly. An example of this is the Naive Bayes algorithm. This cannot use continuous columns as input and cannot predict continuous values. Additionally, some columns may contain so many values that the algorithm cannot easily identify its patterns of interest in the data from which it creates a model. In these cases, you can discretize the data in the columns to enable the use of the algorithms that produce a mining model.

Discretization is the process of putting values into buckets so there are a limited number of possible states.

The buckets themselves are treated as ordered and discrete values. You can discretize both numeric and string columns.

There are several methods to use when discretizing data.

Fortunately, all this methods are present in the WEKA pre-process tool. This operation can be applied by the use of a filter, consisting of this discretization method.

We can demonstrate these methods, and the data we have used.

Basically the discretization learning methods are divided in two categories. These are supervised and unsupervised.

As defined in [9]: Supervised learning is also often referred to as directed data mining.

In this method, variables under investigation can be split into two groups. And these are explanatory variables and one or more dependent variables.

The target of the analysis is to specify a relationship between the explanatory variables and the dependent variable as done in the regression analysis. To apply directed data mining techniques the values of the dependent variable must be known for a sufficiently large part of the data set.

Otherwise, the unsupervised learning is closer to the exploratory split of data mining. In unsupervised learning situations all variables are treated in the same way. There is no distinction between explanatory and dependent variables. In contrast to its name, undirected data mining does have a target to achieve. This target might be as general as data reduction, or specific, such as clustering.

Supervised learning requires the target variable to be well defined and that a sufficient number of its values are given. Unsupervised learning typically has either the target variable as unknown. Or it has only been recorded for small a number of cases.

Despite the apparent complexity of this method, WEKA offers us a powerful filter in the pre-process tab. This allows us to compute both types (supervised and unsupervised) of discretization filter. Shown in picture 6.2 the discretization filter application is in our dataset. In picture 6.3 the final result of the pre-process phase with the WEKA tool is shown.

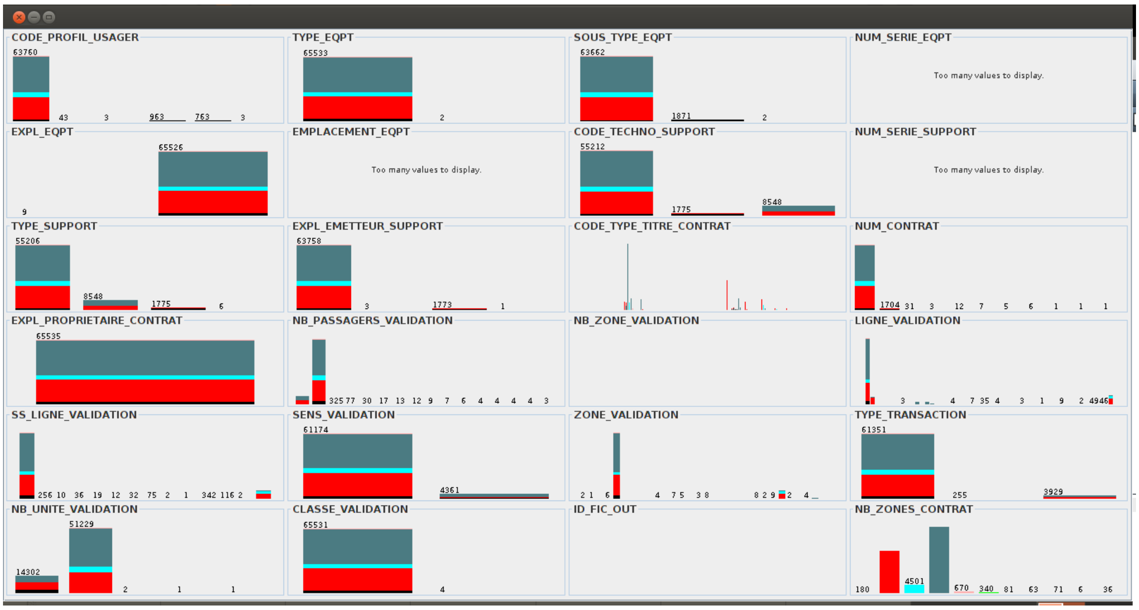
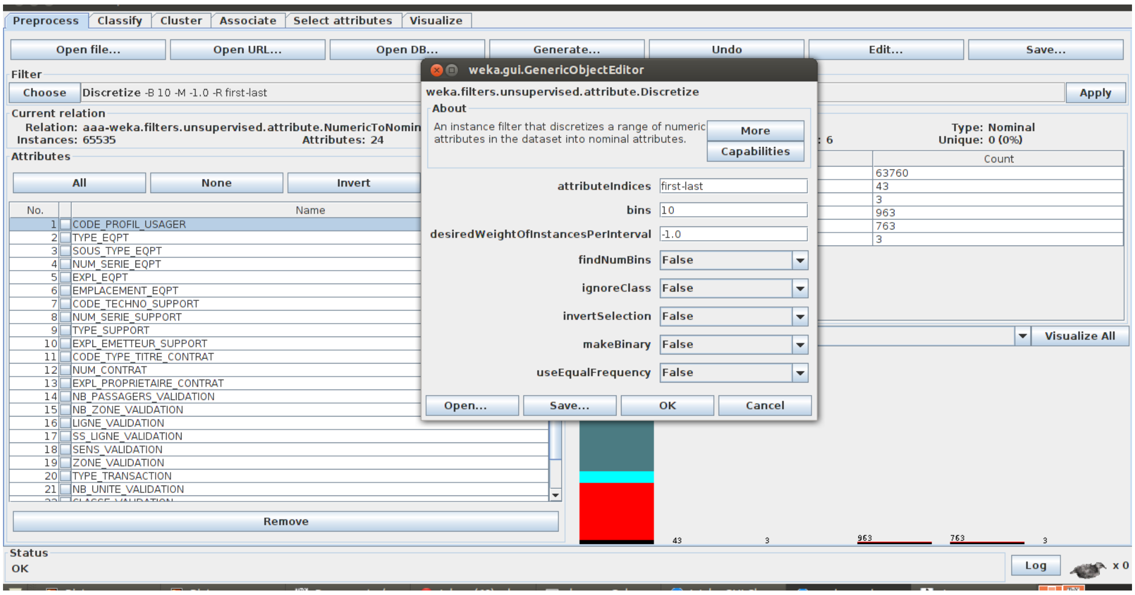


Figure 6.1: The histogram of all the attributes



Filter: Discretize -B 10 -M -1.0 -R first-last

Current relation
Relation: aaa-weka.filters.unsupervised.attribute.NumericToNomin
Instances: 65535 Attributes: 24

No.	Name
1	CODE_PROFIL_USAGER
2	TYPE_EQPT
3	SOUS_TYPE_EQPT
4	NUM_SERIE_EQPT
5	EXPL_EQPT
6	EMPLACEMENT_EQPT
7	CODE_TECHNO_SUPPORT
8	NUM_SERIE_SUPPORT
9	TYPE_SUPPORT
10	EXPL_EMETTEUR_SUPPORT
11	CODE_TYPE_TITRE_CONTRAT
12	NUM_CONTRAT
13	EXPL_PROPRIETAIRE_CONTRAT
14	NB_PASSAGERS_VALIDATION
15	NB_ZONE_VALIDATION
16	LIGNE_VALIDATION
17	SS_LIGNE_VALIDATION
18	SENS_VALIDATION
19	ZONE_VALIDATION
20	TYPE_TRANSACTION
21	NB_UNITE_VALIDATION

weka.gui.GenericObjectEditor
weka.filters.unsupervised.attribute.Discretize
About
An instance filter that discretizes a range of numeric attributes in the dataset into nominal attributes.

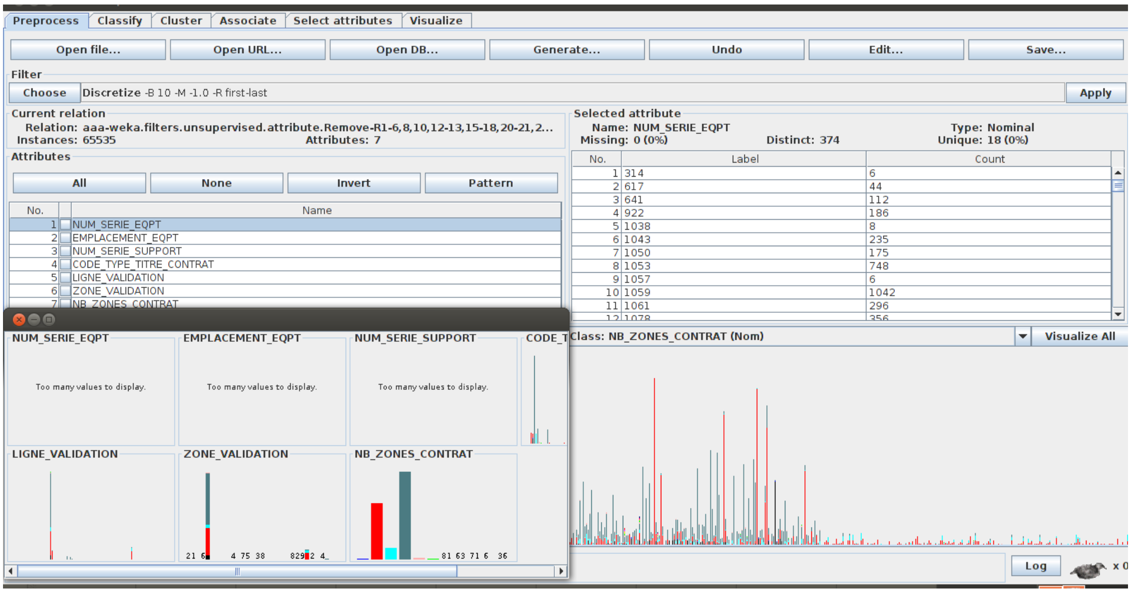
attributeIndices: first-last
bins: 10
desiredWeightOfInstancesPerInterval: -1.0
findNumBins: False
ignoreClass: False
invertSelection: False
makeBinary: False
useEqualFrequency: False

: 6 Type: Nominal
Unique: 0 (0%)

Count
63760
43
3
963
763
3

Status
OK

Figure 6.2: The discretization filter application



Filter: Discretize -B 10 -M -1.0 -R first-last

Current relation
Relation: aaa-weka.filters.unsupervised.attribute.Remove-R1-6,8,10,12-13,15-18,20-21,2...
Instances: 65535 Attributes: 7

No.	Name
1	NUM_SERIE_EQPT
2	EMPLACEMENT_EQPT
3	NUM_SERIE_SUPPORT
4	CODE_TYPE_TITRE_CONTRAT
5	LIGNE_VALIDATION
6	ZONE_VALIDATION
7	NB_ZONES_CONTRAT

Selected attribute
Name: NUM_SERIE_EQPT Type: Nominal
Missing: 0 (0%) Distinct: 374 Unique: 18 (0%)

No.	Label	Count
1	314	6
2	617	44
3	641	112
4	922	186
5	1038	8
6	1043	235
7	1050	175
8	1053	748
9	1057	6
10	1059	1042
11	1061	296
12	1078	356

Class: NB_ZONES_CONTRAT (Nom)

Figure 6.3: The final result of the PreProcess phase

Chapter 7

Data Mining Algorithms

The pre-process step is done in preparation for submitting data to the data-mining algorithms, implemented by WEKA.
The questions to be asked here are:
Which algorithms, and how do they work? [3] [8]
Within this chapter we will show the data mining algorithms that have been used. Here we have used three fundamental classes of algorithm.

7.1 Clustering

Within this context, the use of the word clustering and the term cluster analysis was introduced by Robert Tryon in 1939. Clustering or cluster analysis is a set of techniques for multivariate data analysis that aims to select and group homogeneous elements in a data set. The goal of these algorithms is grouping the items based on a relative measurement of similarity.
This measurement is called distance. The technique, after the identification of a group, will insert each element in each group according to its distance. The measurement and the distance threshold give the accuracy of the algorithm; this is an input factor in each group. In turn, these clustering algorithms are divided according to two approaches.
These approaches are:

Bottom-Up: In this, the elements are considered as clusters themselves. Then the algorithm proceeds to unify the nearest cluster until a fixed minimum number of clusters is reached. Or, when the minimum distance between clusters does not exceed a certain value.

Top-down: This approach begins with a unique large cluster containing all the elements. Within algorithms there are provisions that divide this cluster and in turn create others according to a distance value, until a termination condition is reached. As an example, this could be a fixed number of clusters.

In order to find the clusters of interest in our analysis we have used a famous algorithm called K-Means. We have relied on WEKA's explorer tool to implement this.

The algorithm K-Means is a clustering algorithm, which allows the partition to divide a set of objects into K groups based on their attributes.

With clustering partition (also called non-hierarchical) we define the membership of a group by a distance from the point representative of the cluster (centroid, etc.).

Having fixed the number of groups in the partition result, it is then a variant of algorithm expectation-maximization (EM) that has the objective to determine the K sets of data generated by Gaussian distributions. It is assumed that the attributes of the objects can be represented as vectors. Thus, forming a vector space.

In a nutshell, the purpose of this algorithm is the following: [12]

Each cluster is identified by a centroid or middle-point. The algorithm follows an iterative procedure. First of all it creates K partitions and assigns at each partition an entry point.

This is done so in a random way or by using some heuristic information. It then calculates the centroid of each group, and constructs a new partition associating each entry point to the cluster whose centroid is the closest. Then the centroids are recalculated for the new cluster. This process continues until the algorithm converges.

Why did we choose this algorithm? Our reasoning here is that this algorithm may have a more suitable convergence time, with particular parameters. Particularly, when there is not too much partitioned data.
This is true in our case, as stated we have approximately only 7 relevant attributes. As we have chosen the number of clusters we want to find in advance, this algorithm will converge very quickly. In our case, we found this to be true. Even with massive amounts of data.[7]

7.2 Decision trees

The decision tree is a graph model of decisions and the possible consequences.
In data mining decision trees are used to classify data, as in fact, they are also referred to as classification trees.
This type of graph can describe a decision structure, where nodes may represent the instance classification. And the ramifications represent the properties that lead to these classifications.
The leaves, of the tree, will be the final classes of the various decisions that are made within the classification process.
In this tree, as explained in [24] there are two fundamental conditions:

The split condition: This condition is the predicate that is associated with each internal node. This takes place in the beginning where the data is distributed.

The terminal condition: This identifies the maximum depth of the tree. Contrary to what one may initially assume, increasing the depth of this tree, essentially makes the model more complex. Will not always a have a beneficial result.

The decision tree we have used is the C4.5 algorithm. In the WEKA program this is implemented with the name of J48 (Java version) found in the classification tool.[19] C4.5 is an algorithm used to generate a decision tree that was developed by Ross Quinlan. This in particular is an extension of Quinlan's earlier ID3 algorithm. The decision trees generated by C4.5 can be used for classification. For this reason, C4.5 is often referred to as a statistical classifier. For this reason, WEKA's team have chosen to implement a java version of his tools.

As described in [11] C4.5 builds decision trees from a set of training data using the concept of information entropy. The training data is a set of preclassified samples. Each sample consists of a P-dimensional vector. This is where they represent attributes or features of the sample, as well as the class it may fall into.

At each node of the tree, C4.5 chooses the attribute of the data that most effectively splits its set of samples into subsets, enriched in the respective classes. The splitting criterion is the normalized information gain (difference in entropy). The attribute with the highest normalized information gain is chosen to make the decision and the C4.5 algorithm then recurs on the smaller sub-lists.

7.3 Probabilistic Classifier

As the name suggest they represent a probabilistic approach to solving classification problems. [4] Unlike decision trees, in this case it determines within a certain probability, the membership of an object to a particular class. Given a sample input, the probabilistic classifier is able to predict a probability distribution over a set of classes, rather than only predicting a class for the sample.

Probabilistic classifiers provide classification with a degree of certainty, which can be useful when combining them into larger systems.

Basically the probabilistic classifiers fall into two broad categories. Some models such as logistic regression are conditionally trained.

This means that they optimize the conditional probability directly on a training set. Whereas, some models such as Naive Bayes, which we will use, are trained generatively. Meaning that when the class-conditional distribution and the class prior are found, the conditional distribution is derived using Bayes' rule, at the time of training.[6] [17]

In our analysis we used the most famous Naive Bayes Algorithm.

The Bayesian Classification represents a supervised learning method as well as a statistical method for classification. It assumes an underlying probabilistic model and it allows us to capture uncertainty about the model in a principled way by determining probabilities of the outcomes. It can solve diagnostic and predictive problems.

This Classification is named after Thomas Bayes (1702-1761), who proposed the Bayes Theorem.

Bayesian classification provides practical learning algorithms where prior knowledge and observed data can be combined.

Bayesian Classification provides a useful perspective for understanding and evaluating many learning algorithms. It calculates explicit probabilities for hypothesis, and is robust in its tolerance of noise in its input data.

Why did we choose the Naive Bayes Algorithm for this analysis? This model is simple for us to build. It is an algorithm without complicated iterative parameter estimation. So this is particularly useful in very large datasets, which the case in our analysis. Despite its simplicity, the Naive Bayesian classifier often does surprisingly well. It is widely used because it often outperforms more sophisticated classification methods.

Chapter 8

The test phase

Finally, we can proceed to the test phase.

As we have our data pre-processed, our algorithms, and our data mining tools, we can make our analysis. The analysis follows the WEKA organization. To begin with, we will provide some classification analysis, following this we will search for cluster results. Then finally we will discover some association rules.

For each step and each result, we continue to make improvements. This affects the result of the algorithm changing its input dataset. By this we are referring to the re-application of these pre-process techniques.

We will confirm the results using different algorithms that have the same purpose. The structure of this chapter will follow on from this notion.

We will dedicate section for each experiment.

And each of these experiments will include:

Attribute to use As covered previously, for various reasons, we will not use all the attributes. In each instance we will choose a set of different attributes. Although, since the quantities of useful attributes aren't too high. We should find some possible combinations that are particularly fruit-full.

Attribute to predict This objective is the prediction of a single value that should be understandable, theoretically. Of course not every attribute

is a good candidate for our purpose. Therefore, we should proceed only with attributes deemed to be of real interest to us.
And of course, it is important to remember. The more interesting we find these attributes to be, the more dangerous they are. Because potentially, these may be utilized, becoming a threat to our system.

Algorithm For each technique we appropriate an algorithm. In many cases we use different tests for different settings of the algorithm

Dataset For each test we need to set the dataset input, according to what the data-mining algorithm learns.

8.1 Classification Tests

The first, and most important tests we provide are the classification tests. With some classifiers, the algorithm aims to predict a certain attribute, using a subset of our dataset that is pre-processed. Our aim here is to guess a value. Therefore, the value of output that we need to pay most attention to is the Correctly Classified Instances percentage.
And, where it is possible, the Detailed Accuracy By Class.

Test 1

Configuration Input:

Algorithm Naive Bayes

Attributes input (NUM_SERIE_EQPT, EMPLACEMENT_EQPT, NUM_SERIE_SUPPORT, CODE_TYPE_TITRE_CONTRAT, LIGNE_VALIDATION, ZONE_VALIDATION, NB_ZONES_CONTRACT)

Dataset Percentuage split 50.0%

Attribute to predict NUM_SERIE_SUPPORT

Text output:

```
=== Run information ===

Scheme:weka.classifiers.bayes.NaiveBayes
Relation:     aaa-weka.filters.unsupervised.attribute.Remove-R1-6,
8,10,12-13,15-18,20-21,23-29-weka.filters.unsupervised.attribute.
NumericToNominal-Rfirst-last
Instances:    65535
Attributes:   7
              NUM\_SERIE\_EQPT
              EMPLACEMENT\_EQPT
              NUM\_SERIE\_SUPPORT
              CODE\_TYPE\_TITRE\_CONTRAT
              LIGNE\_VALIDATION
              ZONE\_VALIDATION
              NB\_ZONES\_CONTRAT
Test mode:split 50.0% train, remainder test


Time taken to build model: 0.04 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances       27701               84.5393 %
Incorrectly Classified Instances      5066               15.4607 %
Kappa statistic                          0.2363
Mean absolute error                      0.0002
Root mean squared error                  0.0104
Relative absolute error                 51.6412 %
Root relative squared error             85.7945 %
```

```
Total Number of Instances            32767
```

This prediction could have a high level of positive results. In this case we believe we can improve them even further.
So we continue to perform the same test on the same prediction attribute but we change some parameters.

Test 2

Configuration Input:

Algorithm Naive Bayes

Attributes input (NUM_SERIE_EQPT, EMPLACEMENT_EQPT, NUM_SERIE_SUPPORT, CODE_TYPE_TITRE_CONTRAT, LIGNE_VALIDATION, ZONE_VALIDATION, NB_ZONES_CONTRACT)

Dataset Cross-validation 10 folds

Attribute to predict NUM_SERIE_SUPPORT

Text output:

```
=== Run information ===

Scheme:weka.classifiers.bayes.NaiveBayes
Relation:     aaa-weka.filters.unsupervised.attribute.Remove-R1-6,8,
10,12-13,15-18,20-21,23-29-weka.filters.unsupervised.attribute.
NumericToNominal-Rfirst-last-weka.filters.unsupervised.attribute.Remove-R5
Instances:    65535
Attributes:   6
              NUM_SERIE_EQPT
              EMPLACEMENT_EQPT
              NUM_SERIE_SUPPORT
              CODE_TYPE_TITRE_CONTRAT
              ZONE_VALIDATION
              NB_ZONES_CONTRAT
Test mode:split 66.0% train, remainder test

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances       18917               84.8981 %
Incorrectly Classified Instances      3365               15.1019 %
Kappa statistic                          0.3183
Mean absolute error                      0.0002
Root mean squared error                  0.0094
Relative absolute error                 54.9755 %
Root relative squared error             78.2344 %
Total Number of Instances            22282
```

Test 3

Configuration Input:

Algorithm Naive Bayes

Attributes input (NUM_SERIE_EQPT, EMPLACEMENT_EQPT, NUM_SERIE_SUPPORT, CODE_TYPE_TITRE_CONTRAT, LIGNE_VALIDATION, ZONE_VALIDATION, NB_ZONES_CONTRACT)

Dataset Cross-validation 10 folds

Attribute to predict NUM_SERIE_SUPPORT

Text output:

```
=== Run information ===

Scheme:weka.classifiers.bayes.NaiveBayes
Relation:     aaa-weka.filters.unsupervised.attribute.Remove-R1-6,
8,10,12-13,15-18,20-21,23-29-weka.filters.unsupervised.attribute.
NumericToNominal-Rfirst-last
Instances:    65535
Attributes:   7
              NUM_SERIE_EQPT
              EMPLACEMENT_EQPT
              NUM_SERIE_SUPPORT
              CODE_TYPE_TITRE_CONTRAT
              LIGNE_VALIDATION
              ZONE_VALIDATION
              NB_ZONES_CONTRAT
Test mode:10-fold cross-validation


Time taken to build model: 0.02 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances       63127               96.3256 %
Incorrectly Classified Instances      2408                3.6744 %
Kappa statistic                          0.8611
Mean absolute error                      0.0025
Root mean squared error                  0.0431
Relative absolute error                 15.5571 %
Root relative squared error             48.0361 %
Total Number of Instances            65535
```

Test 4

Configuration Input:

Algorithm Naive Bayes -D (useSupervisedDiscretization)

Attributes input (NUM_SERIE_EQPT, EMPLACEMENT_EQPT, NUM_SERIE_SUPPORT, CODE_TYPE_TITRE_CONTRAT, LIGNE_VALIDATION, ZONE_VALIDATION, NB_ZONES_CONTRACT)

Dataset Cross-validation 10 folds

Attribute to predict NUM_SERIE_SUPPORT

Text output:

```
=== Run information ===

Scheme:weka.classifiers.bayes.NaiveBayes -D
Relation:     aaa-weka.filters.unsupervised.attribute.Remove-R1-6,
8,10,12-13,15-18,20-21,23-29-weka.filters.unsupervised.attribute.
NumericToNominal-Rfirst-last-weka.filters.unsupervised.instance.Randomize-S42
Instances:    65535
Attributes:   7
              NUM_SERIE_EQPT
              EMPLACEMENT_EQPT
              NUM_SERIE_SUPPORT
              CODE_TYPE_TITRE_CONTRAT
              LIGNE_VALIDATION
              ZONE_VALIDATION
              NB_ZONES_CONTRAT
Test mode:10-fold cross-validation
```

```
Time taken to build model: 0.27 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances      55322               84.416  %
Incorrectly Classified Instances    10213               15.584  %
Kappa statistic                         0.3333
Mean absolute error                     0.0002
Root mean squared error                 0.0096
Relative absolute error                55.4426 %
Root relative squared error            79.1399 %
Total Number of Instances           65535
```

Test 5

Configuration Input:

Algorithm Naive Bayes

Attributes input All the attributes of the original dataset

Dataset Percentage split 66%

Attribute to predict CODE_TYPE_TITRE_CONTRAT

Text output:

```
=== Run information ===

Scheme:weka.classifiers.bayes.NaiveBayes
Relation:     aaa-weka.filters.unsupervised.attribute.
NumericToNominal-Rfirst-last-weka.filters.unsupervised.
```

```
attribute.Remove-R1-weka.filters.unsupervised.attribute.
Remove-R26-weka.filters.unsupervised.attribute.Remove-R3-4,7,17,23
Instances:    65535
Attributes:   23
              DTHR_VALIDATION
              CODE_PROFIL_USAGER
              SOUS_TYPE_EQPT
              NUM_SERIE_EQPT
              EMPLACEMENT_EQPT
              CODE_TECHNO_SUPPORT
              NUM_SERIE_SUPPORT
              TYPE_SUPPORT
              EXPL_EMETTEUR_SUPPORT
              CODE_TYPE_TITRE_CONTRAT
              NUM_CONTRAT
              EXPL_PROPRIETAIRE_CONTRAT
              NB_PASSAGERS_VALIDATION
              LIGNE_VALIDATION
              SS_LIGNE_VALIDATION
              SENS_VALIDATION
              ZONE_VALIDATION
              TYPE_TRANSACTION
              NB_UNITE_VALIDATION
              CLASSE_VALIDATION
              DTHR_INSERTION
              DTHR_MODIFICATION
              NB_ZONES_CONTRAT
Test mode:split 66.0% train, remainder test

Time taken to build model: 0.08 seconds
```

```
=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances       20323               91.2082 %
Incorrectly Classified Instances      1959                8.7918 %
Kappa statistic                          0.8958
Mean absolute error                      0.0018
Root mean squared error                  0.0352
Relative absolute error                 11.87   %
Root relative squared error             40.2428 %
Total Number of Instances            22282
```

Test 6

Configuration Input:

Algorithm Naive Bayes -D (useSupervisedDiscretization)

Attributes input All the attributes of the original dataset

Dataset Percentage split 66%

Attribute to predict CODE_TYPE_TITRE_CONTRAT

Text output:

```
=== Run information ===

Scheme:weka.classifiers.bayes.NaiveBayes -D
Relation:     aaa-weka.filters.unsupervised.attribute.
NumericToNominal-Rfirst-last-weka.filters.unsupervised.
attribute.Remove-R1-weka.filters.unsupervised.attribute.
Remove-R26-weka.filters.unsupervised.attribute.Remove-R3-4,7,17,23
Instances:    65535
```

```
Attributes:   23
              DTHR_VALIDATION
              CODE_PROFIL_USAGER
              SOUS_TYPE_EQPT
              NUM_SERIE_EQPT
              EMPLACEMENT_EQPT
              CODE_TECHNO_SUPPORT
              NUM_SERIE_SUPPORT
              TYPE_SUPPORT
              EXPL_EMETTEUR_SUPPORT
              CODE_TYPE_TITRE_CONTRAT
              NUM_CONTRAT
              EXPL_PROPRIETAIRE_CONTRAT
              NB_PASSAGERS_VALIDATION
              LIGNE_VALIDATION
              SS_LIGNE_VALIDATION
              SENS_VALIDATION
              ZONE_VALIDATION
              TYPE_TRANSACTION
              NB_UNITE_VALIDATION
              CLASSE_VALIDATION
              DTHR_INSERTION
              DTHR_MODIFICATION
              NB_ZONES_CONTRAT
Test mode:split 66.0% train, remainder test

Time taken to build model: 0.1 seconds

=== Evaluation on test split ===
=== Summary ===
Correctly Classified Instances       20323               91.2082 %
Incorrectly Classified Instances      1959                8.7918 %
Kappa statistic                          0.8958
Mean absolute error                      0.0018
Root mean squared error                  0.0352
Relative absolute error                 11.87   %
Root relative squared error             40.2428 %
Total Number of Instances            22282
```

Test 7

Configuration Input:

Algorithm Naive Bayes -D (useSupervisedDiscretization)

Attributes input (NUM_SERIE_EQPT, EMPLACEMENT_EQPT, NUM_SERIE_SUPPORT, CODE_TYPE_TITRE_CONTRAT, LIGNE_VALIDATION, ZONE_VALIDATION, NB_ZONES_CONTRACT)

Dataset Percentage split 70%

Attribute to predict CODE_TYPE_TITRE_CONTRAT

Text output:

```
=== Run information ===

Scheme:weka.classifiers.bayes.NaiveBayes -D
Relation:     aaa-weka.filters.unsupervised.attribute.
Remove-R1-6,8,10,12-13,15-18,20-21,23-29-weka.filters.
unsupervised.attribute.NumericToNominal-Rfirst-last
Instances:    65535
Attributes:   7
              NUM_SERIE_EQPT
              EMPLACEMENT_EQPT
              NUM_SERIE_SUPPORT
              CODE_TYPE_TITRE_CONTRAT
              LIGNE_VALIDATION
              ZONE_VALIDATION
              NB_ZONES_CONTRAT
Test mode:split 70.0% train, remainder test

Time taken to build model: 0.05 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances       15963               81.1953 %
Incorrectly Classified Instances      3697               18.8047 %
Kappa statistic                          0.7724
Mean absolute error                      0.0044
Root mean squared error                  0.0514
Relative absolute error                 28.8097 %
Root relative squared error             58.7466 %
Total Number of Instances            19660
```

Test 8

Configuration Input:

Algorithm Naive Bayes -D (useSupervisedDiscretization)

Attributes input (NUM_SERIE_EQPT, EMPLACEMENT_EQPT, NUM_SERIE_SUPPORT, CODE_TYPE_TITRE_CONTRAT, LIGNE_VALIDATION, ZONE_VALIDATION, NB_ZONES_CONTRACT)

Dataset Percentage split 70%

Attribute to predict CODE_TYPE_TITRE_CONTRAT

Text output:

```
=== Run information ===

Scheme:weka.classifiers.bayes.NaiveBayes -D
Relation:     aaa-weka.filters.unsupervised.attribute.Remove-R1-6,
8,10,12-13,15-18,20-21,23-29-weka.filters.unsupervised.attribute.
NumericToNominal-Rfirst-last
Instances:    65535
Attributes:   7
              NUM_SERIE_EQPT
              EMPLACEMENT_EQPT
              NUM_SERIE_SUPPORT
              CODE_TYPE_TITRE_CONTRAT
              LIGNE_VALIDATION
              ZONE_VALIDATION
              NB_ZONES_CONTRAT
Test mode:10-fold cross-validation

Time taken to build model: 0.05 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances       53575               81.7502 %
Incorrectly Classified Instances     11960               18.2498 %
Kappa statistic                          0.7803
Mean absolute error                      0.0043
Root mean squared error                  0.0505
Relative absolute error                 27.9816 %
Root relative squared error             57.6305 %
```

```
Total Number of Instances           65535
```

Test 9

Configuration Input:

Algorithm J48 with parameters confidenceFactor 0.25 (The confidence factor used for pruning smaller values incur more pruning); minNumObj 20 (The minimum number of instances per leaf.)

Attributes input (NUM_SERIE_EQPT, EMPLACEMENT_EQPT, NUM_SERIE_SUPPORT, CODE_TYPE_TITRE_CONTRAT, LIGNE_VALIDATION, ZONE_VALIDATION)

Dataset Percentage split 50%

Attribute to predict ZONE_VALIDATION

Text output:

```
=== Run information ===

Scheme:weka.classifiers.trees.J48 -C 0.25 -M 20
Relation:     aaa-weka.filters.unsupervised.attribute.Remove-R1-6,
8,10,12-13,15-18,20-21,23-29-weka.filters.unsupervised.attribute.
NumericToNominal-Rfirst-last-weka.filters.unsupervised.attribute.Remove-R4
Instances:    65535
Attributes:   6
              NUM_SERIE_EQPT
              EMPLACEMENT_EQPT
              NUM_SERIE_SUPPORT
              LIGNE_VALIDATION
              ZONE_VALIDATION
              NB_ZONES_CONTRAT
Test mode:split 50.0% train, remainder test
```

```
Time taken to build model: 0.59 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances       32069               97.8698 %
Incorrectly Classified Instances       698                2.1302 %
Kappa statistic                          0.9132
Mean absolute error                      0.0019
Root mean squared error                  0.0312
Relative absolute error                 11.4708 %
Root relative squared error             34.7229 %
Total Number of Instances            32767
```

Test 10

Configuration Input:

Algorithm J48 with parameters confidenceFactor 0.5 (The confidence factor used for pruning smaller values incur more pruning); minNumObj 40 (The minimum number of instances per leaf.)

Attributes input (NUM_SERIE_EQPT, EMPLACEMENT_EQPT, NUM_SERIE_SUPPORT, CODE_TYPE_TITRE_CONTRAT, LIGNE_VALIDATION, ZONE_VALIDATION)

Dataset Percentage split 50%

Attribute to predict ZONE_VALIDATION

Text output:

```
=== Run information ===
Scheme:weka.classifiers.trees.J48 -C 0.5 -M 40
Relation:     aaa
Instances:    65535
Attributes:   7
              NUM_SERIE_EQPT
              EMPLACEMENT_EQPT
              NUM_SERIE_SUPPORT
              CODE_TYPE_TITRE_CONTRAT
              LIGNE_VALIDATION
              ZONE_VALIDATION
              NB_ZONES_CONTRAT
Test mode:10-fold cross-validation


Time taken to build model: 0.5 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances       64329               98.1598 %
Incorrectly Classified Instances      1206                1.8402 %
Kappa statistic                          0.9253
Mean absolute error                      0.0016
Root mean squared error                  0.0284
Relative absolute error                  9.7244 %
Root relative squared error             31.6413 %
Total Number of Instances            65535
```

Test 11

Configuration Input:

Algorithm NaiveBayes

Attributes input (NUM_SERIE_EQPT, EMPLACEMENT_EQPT, NUM_SERIE_SUPPORT, CODE_TYPE_TITRE_CONTRAT, ZONE_VALIDATION)

Dataset Percentage split 66%

Attribute to predict ZONE_VALIDATION

Text output:

```
=== Run information ===

Scheme:weka.classifiers.bayes.NaiveBayes
Relation:     aaa-weka.filters.unsupervised.attribute.Remove-R1-6,
8,10,12-13,15-18,20-21,23-29-weka.filters.unsupervised.attribute.
NumericToNominal-Rfirst-last-weka.filters.unsupervised.attribute.Remove-R5
Instances:    65535
Attributes:   6
              NUM_SERIE_EQPT
              EMPLACEMENT_EQPT
              NUM_SERIE_SUPPORT
              CODE_TYPE_TITRE_CONTRAT
              ZONE_VALIDATION
              NB_ZONES_CONTRAT
Test mode:split 66.0% train, remainder test

Time taken to build model: 0.07 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances       21024               94.3542 %
Incorrectly Classified Instances      1258                5.6458 %
```

```
Kappa statistic                          0.7833
Mean absolute error                      0.0042
Root mean squared error                  0.0549
Relative absolute error                 26.2272 %
Root relative squared error             60.9577 %
Total Number of Instances            22282
```

Test 12

Configuration Input:

Algorithm NaiveBayes discretized

Attributes input (NUM_SERIE_EQPT, EMPLACEMENT_EQPT, NUM_SERIE_SUPPORT, CODE_TYPE_TITRE_CONTRAT, ZONE_VALIDATION)

Dataset Cross-validation 10 folds

Attribute to predict ZONE_VALIDATION

Text output:

```
=== Run information ===

Scheme:weka.classifiers.bayes.NaiveBayes -D
Relation:     aaa-weka.filters.unsupervised.attribute.Remove-R1-6,
8,10,12-13,15-18,20-21,23-29-weka.filters.unsupervised.attribute.
NumericToNominal-Rfirst-last-weka.filters.unsupervised.attribute.Remove-R5
Instances:    65535
Attributes:   6
              NUM_SERIE_EQPT
              EMPLACEMENT_EQPT
              NUM_SERIE_SUPPORT
              CODE_TYPE_TITRE_CONTRAT
              ZONE_VALIDATION
              NB_ZONES_CONTRAT
Test mode:10-fold cross-validation

Time taken to build model: 0.31 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances       61199               93.3837 %
Incorrectly Classified Instances      4336                6.6163 %
Kappa statistic                          0.7642
Mean absolute error                      0.0045
Root mean squared error                  0.0561
Relative absolute error                 27.8668 %
Root relative squared error             62.5709 %
Total Number of Instances            65535
```

Test 13

Configuration Input:

Algorithm NaiveBayes discretized

Attributes input (NUM_SERIE_EQPT, EMPLACEMENT_EQPT, NUM_SERIE_SUPPORT, LIGNE_VALIDATION, ZONE_VALIDATION)

Dataset Percentuage split 66%

Attribute to predict ZONE_VALIDATION

Text output:

```
=== Run information ===
Scheme:weka.classifiers.bayes.NaiveBayes -D
Relation:     aaa-weka.filters.unsupervised.attribute.Remove-R1-6,
8,10,12-13,15-18,20-21,23-29-weka.filters.unsupervised.attribute.
NumericToNominal-Rfirst-last-weka.filters.unsupervised.attribute.Remove-R4
Instances:    65535
Attributes:   6
              NUM_SERIE_EQPT
              EMPLACEMENT_EQPT
              NUM_SERIE_SUPPORT
              LIGNE_VALIDATION
              ZONE_VALIDATION
              NB_ZONES_CONTRAT
Test mode:split 66.0% train, remainder test


Time taken to build model: 0.26 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances       21688              97.3342 %
Incorrectly Classified Instances       594               2.6658 %
Kappa statistic                          0.8951
Mean absolute error                      0.002
Root mean squared error                  0.0364
Relative absolute error                 12.3703 %
Root relative squared error             40.4151 %
Total Number of Instances            22282
```

8.2 Clustering Tests

After the classification tests we provide some Clustering test in order to try to discover (in particular) trends between zone validation, contract type and id card.

Test 1

Configuration Input:

Number of clusters 6

Test mode Classes to clusters evaluation on training data

Attributes input (NUM_SERIE_EQPT, EMPLACEMENT_EQPT, NUM_SERIE_SUPPORT, CODE_TYPE_TITRE_CONTRAT, LIGNE_VALIDATION, ZONE_VALIDATION)

Attribute Ignored Zone_Validation

Text output:

```
=== Run information ===

Scheme:weka.clusterers.SimpleKMeans -N 6 -A "weka.core.EuclideanDistance
 -R first-last" -I 500 -S 10
Relation:     aaa-weka.filters.unsupervised.attribute.Remove-R1-6,8,10,
12-13,15-18,20-21,23-29-weka.filters.unsupervised.attribute.
NumericToNominal-Rfirst-last
Instances:    65535
Attributes:   7
              NUM_SERIE_EQPT
              EMPLACEMENT_EQPT
              NUM_SERIE_SUPPORT
              CODE_TYPE_TITRE_CONTRAT
              LIGNE_VALIDATION
              NB_ZONES_CONTRAT
Ignored:
              ZONE_VALIDATION
Test mode:Classes to clusters evaluation on training data
=== Model and evaluation on training set ===


kMeans
======


Number of iterations: 3
Within cluster sum of squared errors: 164197.0
Missing values globally replaced with mean/mode

=== Model and evaluation on training set ===

Clustered Instances

0      35521 ( 54%)
1       7162 ( 11%)
2      12013 ( 18%)
3       1490 (  2%)
4       4260 (  7%)
5       5089 (  8%)
```

Test 2

Configuration Input:

Number of clusters 4

Test mode Classes to clusters evaluation on training data

Attributes input (NUM_SERIE_EQPT, EMPLACEMENT_EQPT, CODE_TYPE_TITRE_CON LIGNE_VALIDATION, ZONE_VALIDATION)

Attribute Ignored NUM_SERIE_SUPPORT

Text output:

```
=== Run information ===

Scheme:weka.clusterers.SimpleKMeans -N 4 -A "weka.core.EuclideanDistance
 -R first-last" -I 500 -S 10
Relation:     aaa-weka.filters.unsupervised.attribute.Remove-R1-6,8,10,
12-13,15-18,20-21,23-29-weka.filters.unsupervised.attribute.
NumericToNominal-Rfirst-last
Instances:    65535
Attributes:   7
              NUM_SERIE_EQPT
              EMPLACEMENT_EQPT
              CODE_TYPE_TITRE_CONTRAT
              LIGNE_VALIDATION
              ZONE_VALIDATION
              NB_ZONES_CONTRAT
Ignored:
              NUM_SERIE_SUPPORT
Test mode:Classes to clusters evaluation on training data
=== Model and evaluation on training set ===


kMeans
======

Number of iterations: 3
Within cluster sum of squared errors: 167302.0
```

```
Missing values globally replaced with mean/mode

Cluster centroids:
                                        Cluster#
Attribute                   Full Data         0         1          2         3
                              (65535)   (39254)    (7133)    (17658)    (1490)
==========================================================================
NUM_SERIE_EQPT                   1181      2994      4872       1181      1089
EMPLACEMENT_EQPT                    1        77      7933          1       416
CODE_TYPE_TITRE_CONTRAT          7004      7004      8001       7100      7004
LIGNE_VALIDATION                    1         1      9999          1         1
ZONE_VALIDATION                   245       245       270        245       245
NB_ZONES_CONTRAT                    3         3         1          1         3




Time taken to build model (full training data) : 0.67 seconds
```

8.3 Association Tests

Test 1

Configuration Input:

Algorithm FilteredAssociator

Attributes input (NUM_SERIE_EQPT, EMPLACEMENT_EQPT, NUM_SERIE_SUPPORT, CODE_TYPE_TITRE_CONTRAT, LIGNE_VALIDATION, ZONE_VALIDATION)

Text output:

```
Associator Model
```

```
Apriori
=======

Minimum support: 0.3 (19660 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 14

Best rules found:

 1. NUM_SERIE_SUPPORT=0 LIGNE_VALIDATION=1 NB_ZONES_CONTRAT=3 25977 ==>
 ZONE_VALIDATION=245 25977    conf:(1)
 2. CODE_TYPE_TITRE_CONTRAT=7004 21966 ==>
  NUM_SERIE_SUPPORT=0 21966    conf:(1)
 3. CODE_TYPE_TITRE_CONTRAT=7004 21966 ==>
 NB_ZONES_CONTRAT=3 21966    conf:(1)
 4. CODE_TYPE_TITRE_CONTRAT=7004 NB_ZONES_CONTRAT=3 21966 ==>
 NUM_SERIE_SUPPORT=0 21966    conf:(1)
 5. NUM_SERIE_SUPPORT=0 CODE_TYPE_TITRE_CONTRAT=7004 21966 ==>
 NB_ZONES_CONTRAT=3 21966    conf:(1)
 6. CODE_TYPE_TITRE_CONTRAT=7004 21966 ==>
 NUM_SERIE_SUPPORT=0 NB_ZONES_CONTRAT=3 21966    conf:(1)
 7. CODE_TYPE_TITRE_CONTRAT=7004 ZONE_VALIDATION=245 21303 ==>
  NUM_SERIE_SUPPORT=0 21303    conf:(1)
 8. CODE_TYPE_TITRE_CONTRAT=7004 ZONE_VALIDATION=245 21303 ==>
 NB_ZONES_CONTRAT=3 21303    conf:(1)
 9. CODE_TYPE_TITRE_CONTRAT=7004 ZONE_VALIDATION=245
 NB_ZONES_CONTRAT=3 21303 ==> NUM_SERIE_SUPPORT=0 21303    conf:(1)
10. NUM_SERIE_SUPPORT=0 CODE_TYPE_TITRE_CONTRAT=7004
ZONE_VALIDATION=245 21303 ==> NB_ZONES_CONTRAT=3 21303    conf:(1)
```

8.4 Experimenter Tests

In this section our experiments were conducted in order to evaluate the quality of these data mining algorithms.
We continue evaluating them with the same set of attribute, using the Experimenter tool.

Test 1

Configuration Input:

Algorithm Compared J48, Naive Bayes

Criteria Percent Correct Predicted

Attribute Predicted Zone_Validation

Attributes input (NUM_SERIE_EQPT, EMPLACEMENT_EQPT, NUM_SERIE_SUPPORT, CODE_TYPE_TITRE_CONTRAT, LIGNE_VALIDATION, ZONE_VALIDATION)

Text output:

```
Tester:     weka.experiment.PairedCorrectedTTester
Analysing:  Percent_correct
Datasets:   1
Resultsets: 2
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       5/7/14 11:23 AM


Dataset                   (1) bayes.Na | (2) trees
---------------------------------------------------
'aaa-weka.filters.unsuper(100)   97.56 |   99.69 v
---------------------------------------------------
```

```
                               (v/ /*) |   (1/0/0)


Key:
(1) bayes.NaiveBayes '' 5995231201785697655
(2) trees.J48 '-C 0.25 -M 2' -217733168393644444
```

The result is clear and we can say that with the same parameters the decision tree algorithm is more precise that the Naive Bayes.

Chapter 9

Results Analysis

At this stage we can begin to interpret these results.
As we have conducted numerous tests that have provided us with our evidence. We need to form understandings of the results in great depth in order to make determinations regarding the possible outcomes.
Are these results of interest and can we establish the relative levels of danger these results may pose? From the classification tests, we can conclude that our provisions are very positive. On average exceeding 90% of the correctness coefficient.
Regarding these three attributes: (see the pictures A.1 A.2 for more details)

- Zone Validation
- Code type Contract
- Num Serie Support

From the clustering test we will see how it's possible to retrieve a trend from a group of population (in our case these are the parameters).
Unfortunately the last tests involving the association rules failed to bring us an interesting association. This is because the only rule that may associate the area of ticket validation is with the contractual area of operation, which, is of logical consequence. Thus, it is derivable without any data mining analysis.

Now we should revisit the original purpose of our work.
Is it possible that a company could co-ordinate an insider attack on our system and violate these privacy rules? Theoretically, and in light of these specifications, companies should not be able to understand, nor investigate the frequencies of business from other companies/parties.
Basically, a company should not be able to answer certain questions, such as:
Where does company-A sell more contract type-B despite type contract-C?
Which is the most desirable zone to sell a ticket of type contract-X?
How should prices be modified in order to create a negative concurrence?
Let us look into some possible scenarios whereby companies could break the security rules of the Stimer's system using the articles and techniques featured in our data mining analysis.

9.1 Scenario 1

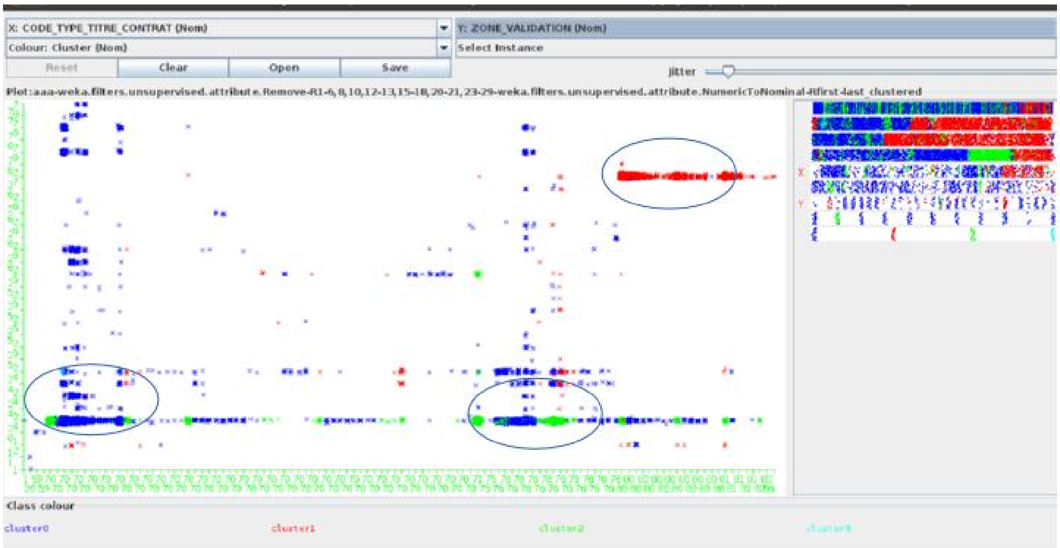
With the first and lasts classification test we had shown that theoretically, knowing only the contract type and the validation zone we were able to progress to the correct Num Serie Support. Which would reveal the identity of the contract. Of course, for the safety of our system, this should be impossible and absolutely private.
This is because a company could determine the history of an identity. This history would be revealing the type of contract. Along with, the movements, and the general choices of single identity (This does not necessarily mean a single person but a single 'carta contratto').
This could then lead to intercepting the offer being directed to the single persons with, of course, different and cheaper contracts. Obviously, this being regarded as unfair competition.

9.2 Scenario 2

With the tests 7 and 8 we are able to provide the type ticket contract. In this situation a company could easily determine the distribution of the type contract within the zones.
How could this information be used?
Reconstructing the distribution of contracts to a company can subtract the different types of contract area with a different offer, this being 'ad hoc'. This also constitutes unfair competition.

9.3 Scenario 3

In this scenario, as we will outline, thing can get a little bit complicated. When we discuss cluster analysis we refer to a set of exploratory techniques that aim to group the unity of a population in statistics on the basis of their similarity. Done in terms of values entered by the observed variables.
Ideally, what we will not do is to partition the population in a way that will unite parts of the same group that are very similar. Otherwise, a unit of different groups can be very different from each other.
This is particularly true in our case, as we stated, we want to discover very similar behaviours within a group, trend or particular attribute. The attribute here is the contract type, and what we aim to discover is the trend of the contract type in a group of Different zones.
Using the two clustering tests, as shown in picture 9.1, we are able to trace the trend of the contract type used in a group of specific zones. This means that we can understand what the trend is and use different types of contracts in areas that are consecutive. Here also, we are able to discover the contract distribution of a competitor. So by intercepting this request, once again, the opportunity to engage strategies deemed as unfair competition may arise.



X: CODE_TYPE_TITRE_CONTRAT (Nom)

Y: ZONE_VALIDATION (Nom)

Colour: Cluster (Nom)

Plot:aaa-weka.filters.unsupervised.attribute.Remove-R1-6,8,10,12-13,15-18,20-21,23-29-weka.filters.unsupervised.attribute.NumericToNominal-Rfirst-last_clustered

Figure 9.1: Circled the clusters discovered

Chapter 10

Defence

The topic discussed in chapter will centre on certain countermeasures applied for the prevention of security threats. Understanding that threats, of this nature, would take the form or data-mining analysis. A key feature of any respectable security strategy is to establish the best possible solution, specific to our requirements. As outlined previously, our system requires the various parties to perform the necessary evaluations and at the same time, our system must prevent is data to be accessed by these parties. The system's organization chief is of course crucial to these functions. Another key objective of this system is to perform evaluations calculating the corresponding ticket values and their sales for each company in order to redistribute of the profits. It may seem unusual to design a system that allows data to clear, whereby, the computation of economic evaluations are performed by parties using the same accessible data. Simultaneously, this system must not allow these parties to perform any alternative analysis using information from this database. How would this be made possible? Unfortunately simply encrypting the data is not a viable option. The solution involves masking the data. We can do this by using some masking/sanitization functions that will modify the data. This is done so in a way that prohibits the data from being analysed in ways that would threaten privacy and security. Although, this would retain some information, these sufficient properties being used to

compute the economic evaluation required. [18] [21]
What exactly are these functions and how we should we use them? [15] In a previous work we defined what data masking means, along with most important and useful techniques.
Some of most renowned we have referenced: [13] [14] [22]

Substitution This technique randomly replaces the contents of a data column with information that looks similar but is completely unrelated to the real details.

Truncate Values This will simply delete a column of data by replacing it with NULL values

Increase/Decrease values Working in particular with numeric values. We can increase or decrease the number for a given value that introduces an acceptable error.

Shuffling Essentially the data in a column is randomly moved between rows until there is no longer any reasonable correlation with the remaining information in these rows.

We also applied our work to hypothetical database structures, with effectively the same problems of our database, but with some limitations.
Unfortunately, in addition to having the same limits described in the previously, we must also be aware of the clash between most of our data, which is nominal type. This means that the values are not correspondent to the physical quantities, but they are, in fact just a representation of them. We can substitute this value with a random one that will not change anything.
So, why this is a problem?
We perform our analysis understanding this aspect. This is an analysis that operates in addition, without considering the actual meaning of each value. But instead to its relative importance to the whole set of values. This means that even if we use data masking techniques and we modify them, we do not have the certainty that we are preventing data mining analysis. This is

because we are not working with the specific meanings of each single value but with the global representation of them.
Although, this could be different if we are to use a legend of this database. This could provide assistance to our analysis, enabling us to cross this data with other externals. In which case, the large-scale application of data masking techniques could prevent this type of vulnerability. These understandings, of course, could be applied in future work of this kind.

Chapter 11

Future work and Conclusions

This last chapter takes an outlook on some future steps. As we have acknowledged throughout this work, implementing a high quality defence mechanism is vital in order to prevent these kinds of attacks. In the previous chapter implementations of sound data masking techniques could be a reasonable security wall. Although in our case, this is not a simple operation. This is because there are limitations deriving from the structure of the database. And, from the type of the analyses we could make.

Another important aspect is the possible scenario of attacks from within the organisations, this being the insider threat. Also be aware of these as 'external' attacks to our system. A concept being relevant in many instances would pose the pressing question:

What if our database fell in the wrong hands?

And, what could they discover?

With the understanding that data can be cross-referenced with relevant external data, another important question to ask is: what then can be found? As we have illustrated, this relevant data, thus being extremely sensitive information, can be discovered. This endangers the acceptable levels of security of our system, not only in relation to generic groups of people. But, also trends and the developments that stem from customers activity, tracing back to single displacements of individual users.

Drawing parallels to these issues, in our work we encountered the deterrent security measures. Although, more accurately described as obstacles, to our analysis.

This is in regarding to the fact that such information will be encoded in some way. In practicality, the individual values of these attributes are not unique for all the companies. As they have different meanings assigned for each company. And therefore up to now, it has not been possible to have access to the legend for this database.

And for this reason no knowledge relating to the real meanings of any of the attribute values. It has not been possible to carry out this type of analysis. This is of course of benefit to its future development.

In summary, we have conducted an analysis of the Stimer's database. In order to understand and then discuss its possible vulnerabilities we performed data-mining analysis of this database. For these operations we have used the data analysis tool WEKA. Throughout this project we have demonstrated the management of these threats and insider attacks, by using these analyses.

Acknowledgements

Sono tante le persone che vorrei ringraziare in questa tesi.

In primo luogo, ovviamente, ringrazio enormemente il prof. Marco Prandini e il prof. Aldo Campi per tutto il lavoro e il supporto datomi in questi mesi.

Ringrazio inoltre il prof Moreno Marzolla per la sua disponibilità.

Come non ringraziare poi la mia famiglia: mio padre, mia madre e la mia fantastica sorella che mi hanno permesso di diventare quello che sono.

Per questo motivo questa tesi è dedicata a loro, la migliore famiglia che potessi desiderare.

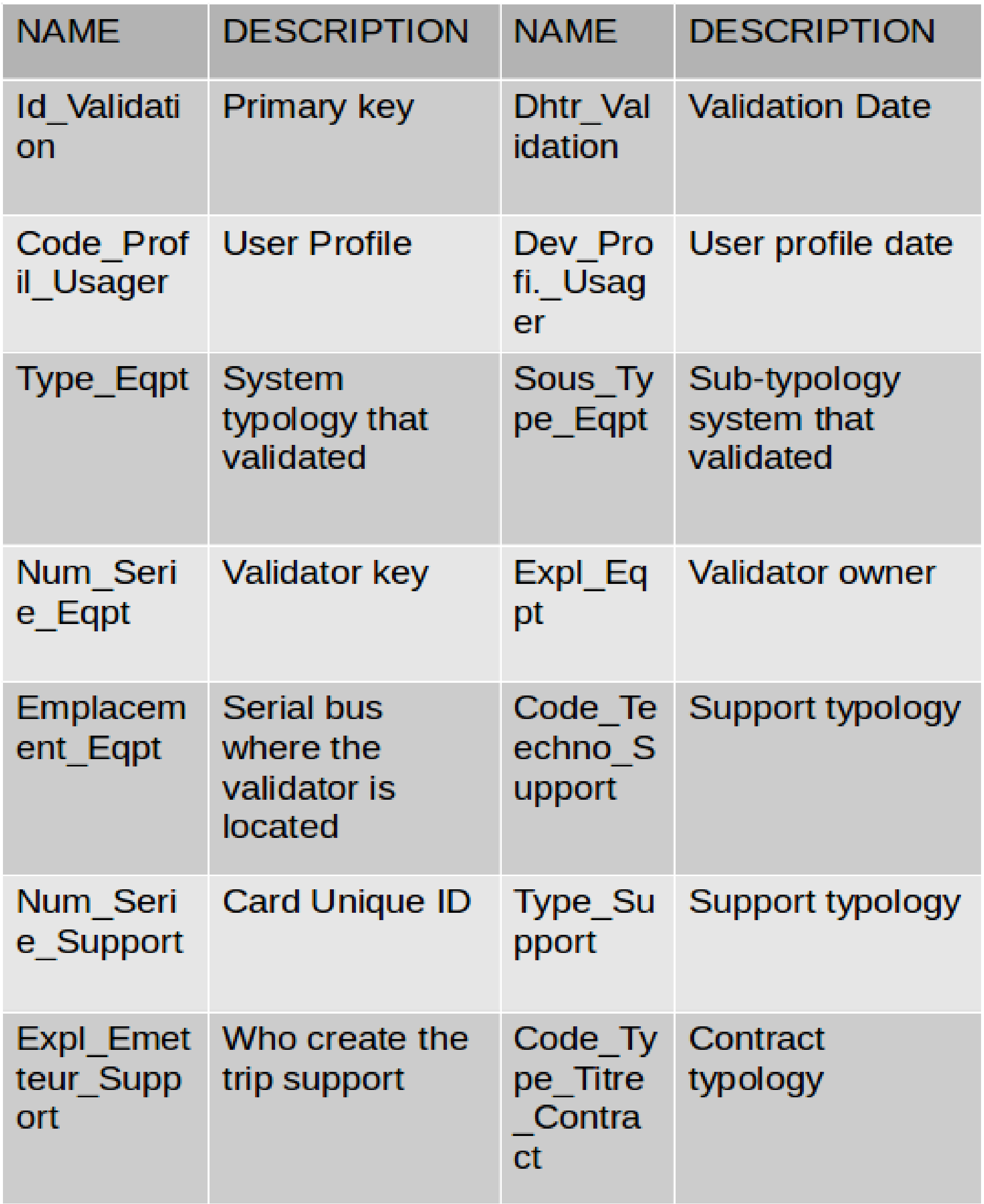
Ringrazio i miei amici d'infanzia, non bastano 1000 chilometri per rompere un'amicizia e ovunque andrò sarete sempre presenti come lo siete stati in tutti questi anni.

Ringrazio infine tutti i miei colleghi, gli amici, e i coinquilini conosciuti in questi lunghi sei anni all'Università di Bologna. Sei anni fatti di migliaia di esperienze, avventure e momenti magici; conditi da una particolare esperienza all'estero, che mai sarebbero stati così fantastici senza tutti voi; i migliori che potessi incontrare.

Appendix A

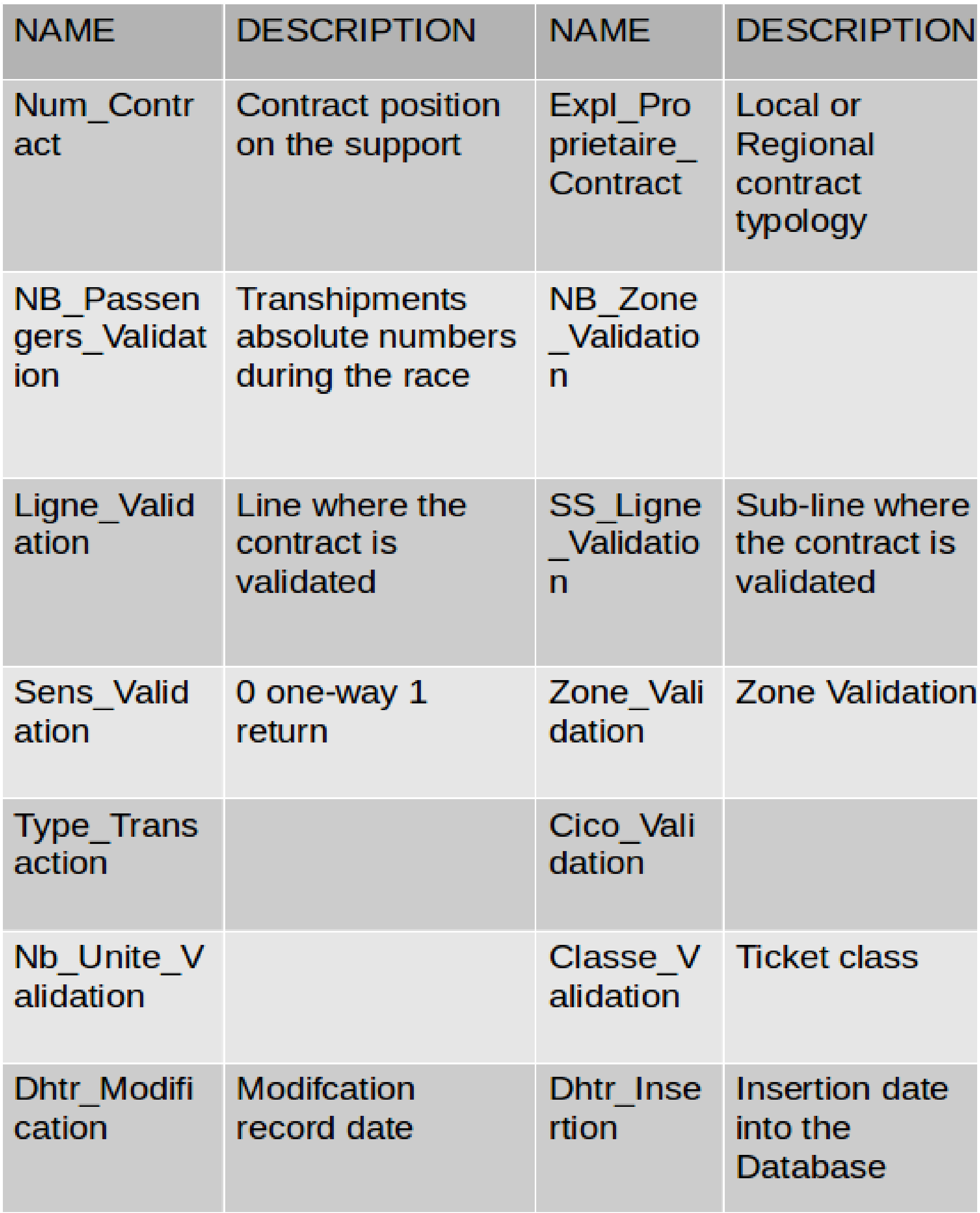
Appendix

In this appendix there is a description of all the parameters of the database with which we worked.
Picture A.1 A.2.



NAME	DESCRIPTION	NAME	DESCRIPTION
Id_Validation	Primary key	Dhtr_Validation	Validation Date
Code_Profil_Usager	User Profile	Dev_Profi._Usager	User profile date
Type_Eqpt	System typology that validated	Sous_Type_Eqpt	Sub-typology system that validated
Num_Serie_Eqpt	Validator key	Expl_Eqpt	Validator owner
Emplacement_Eqpt	Serial bus where the validator is located	Code_Teechno_Support	Support typology
Num_Serie_Support	Card Unique ID	Type_Support	Support typology
Expl_Emetteur_Support	Who create the trip support	Code_Type_Titre_Contract	Contract typology

Figure A.1: The full database parameters description (1)



NAME	DESCRIPTION	NAME	DESCRIPTION
Num_Contract	Contract position on the support	Expl_Proprietaire_Contract	Local or Regional contract typology
NB_Passengers_Validation	Transhipments absolute numbers during the race	NB_Zone_Validation	
Ligne_Validation	Line where the contract is validated	SS_Ligne_Validation	Sub-line where the contract is validated
Sens_Validation	0 one-way 1 return	Zone_Validation	Zone Validation
Type_Transaction		Cico_Validation	
Nb_Unite_Validation		Classe_Validation	Ticket class
Dhtr_Modification	Modifcation record date	Dhtr_Insertion	Insertion date into the Database

Figure A.2: The full database parameters description (2)

Bibliography

[1] Rakesh Agrawal and Ramakrishnan Srikant. Privacy-preserving data miningprivacy-preserving data mining. In ACM, editor, *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, 2000.

[2] Walter Cerroni Aldo Campi. Progetto sistema di clearing; centro interdipartimentale di ricerca industriale sul ict, 29 May 2012.

[3] Doug Alexander. Data mining http://www.laits.utexas.edu/ anorman/bus.for/course.mat/alex/.

[4] Christopher M. Bishop. Pattern recognition and machine learning, 2006.

[5] Remco R. Bouckaert, Eibe Frank, Mark A. Hall, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. Weka experiences with a java open-source project. 2010.

[6] Prabhakar Raghavan Christopher D. Manning and Hinrich Schütze. Introduction to information retrieval, 2008.

[7] Sergei Vassilvitskii David Arthur. How slow is the k-means method?, 2006.

[8] G. Holmes, A. Donkin, and Ian H. Witten. Weka: a machine learning workbench. In *Intelligent Information Systems,1994. Proceedings of the 1994 Second Australian and New Zealand Conference on*, pages 357–361, Nov 1994.

[9] Yuichi Mori James E. Gentle, Wolfgang Karl Hardle. Handbook of computational statistics: Concepts and methods, 2012.

[10] PhD Jay Urbain. Data mining: Preprocessing.

[11] Kotsiantis. Supervised machine learning: A review of classification techniques, 249-268, 2007.

[12] J. B. MacQueen. Some methods for classification and analysis of multivariate observations, Retrieved 2009.

[13] S.Peisert A.Singh B.Bhumiratana D.Agarwal M.Bishop, J.Cummins. Relationship and data sanitization: A study in scarlet. 2010.

[14] B.Trammell E.Boschi B.Plattner M.Burkhart, D.Schatzmann. The role of network trace anonymization under attack. January 2010.

[15] Andrea Melis. Rapport de stage de fin d'études, 2013.

[16] Dr. Ing. Cristian Mihaescu. Description of weka (java-implemented machine learning tool), 2011.

[17] Dr. Ing. Cristian Mihaescu. Naive-bayes classification algorithm, 2011.

[18] Huw Price. *Scrambling, Masking and Obfuscating Production Data.* 2012.

[19] J. R. Quinlan. C4.5: Programs for machine learning. morgan kaufmann publishers, 1993.

[20] Mark Hall Richard Kirkby Peter Reutemann Remco R. Bouckaert, Eibe Frank. Weka manual, 2014.

[21] Camouflage Data Masking Specialist. *Data Masking Best Pratices.* 2012.

[22] R.Acharya S.R.Ganta. Adaptive data anonymization against information fusion based privacy attacks on enterprise data. 2008.

[23] Gunter Saake Tim Frey, Marius Gelhausen. Categorization of concerns: a categorical program comprehension model. In ;, editor, *PLATEAU*, 2011.

[24] Xindong Wu, Vipin Kumar, J. Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, GeoffreyJ. McLachlan, Angus Ng, Bing Liu, PhilipS. Yu, Zhi-Hua Zhou, Michael Steinbach, DavidJ. Hand, and Dan Steinberg. Top 10 algorithms in data mining. 2008.

[25] www.startromagna.it.