

**ASSIGNMENT COVER SHEET**

**Student Name Michael James**

**ID Number: B00019330**

**Course: Computer Science**

**Year: 1 of 1**

**Lecturer: Geraldine**

**Title of Assignment: Mining a Dataset**

**Due Date: 25/11/14**

**Date Submitted: 14/11/14**

The material contained in this assignment is the author’s original work, except where work quoted is duly acknowledged in the text. No aspect of this assignment has been previously submitted for assessment in any other unit or course.

Signed: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_/\_\_\_\_\_\_/\_\_\_\_\_\_

Contents

[CRISP-DM Phase 1 (Business Understanding) 3](#_Toc403394018)

[CRISP-DM Phase 2 (Data Understanding) 4](#_Toc403394019)

[CRISP-DM Phase 3 (Data Preparation) 10](#_Toc403394020)

[CRISP-DM Phase 4 (Modelling) 18](#_Toc403394021)

[Neural Network Model 19](#_Toc403394022)

[Naïve Bayes Model 20](#_Toc403394023)

[CRISP-DM Phase 5 (Evaluation) 21](#_Toc403394024)

## CRISP-DM Phase 1 (Business Understanding)

The dataset I have been allocated was the “wimbelton” set. It is related to a tennis tournament and has columns such as player name, gender and what the result of a certain match was. The objectives of the project are to mine this dataset to create a model that will accurately predict whether player 1 or player 2 would win a game. Currently the dataset has some major problems which will need to be rectified before a final model can be presented. A major concern I will deal with will also be that the model I create should work to a high level of accuracy on other data sets. Ii would like to see it at over 90% on other data sets as well as my allocated one. This would prove the model is flexible and not over fitted.

* The goal is to have a final model that will have over 90% success rate for this dataset but importantly it should also be flexible to work on other datasets too and not be over-fitted. My project plan is to follow the CRISP-DM industry standard to mine this dataset.

**Technical Objectives**

* The technical objectives are to access the initial dataset, then come up with a series of data mining techniques to essentially optimize and clean the dataset getting rid of any noisy data or excess un-needed data.
* The overall goal is to create a model that will predict the class label “result” of the matches using some suitable algorithms and techniques for this particular dataset.

**Project Plan**

|  |  |  |  |
| --- | --- | --- | --- |
| **Week 1** | **Week 2** | **Week 3** | **Week 4** |
|  |  |  |  |
| Research | Do phase 2 | Think about choosing a model | Choose technique |
| Do phase 1 | Research data types | Do phase 3 | Apply various models but choose 2 best suited to data set |
|  | Look into what makes data noisy | Thoroughly clean and prepare the data for modelling | Optimise the Model |
|  |  | Check against other data sets | Think about why result is what it is |
|  |  |  | Do phase 5, evaluation |
|  |  |  |  |

## CRISP-DM Phase 2 (Data Understanding)

**Describe the data**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Description** | **Type** | **Mean** | **Min** | **Max** | **Deviation** | **Mode** |
|  |  |  |  |  |  |  |
| Result | Binomial | 0.5 | 0 | 1 | NA | 1(122) |
| Player 1 | Polynomial | NA | NA | NA | NA | NDjokovic(7) |
| Player 2 | Polynomial | NA | NA | NA | NA | A.Murray(7) |
| Round | Integer | 2.084 | 1.000 | 40.000 | +/- 2.799 | NA |
| FSP1 | Integer | 64.051 | 43 | 85 | =/- 6.822 | NA |
| FSW1 | Integer | 40.380 | 9 | 102 | +/- 17.958 | NA |
| SSP1 | Integer | 35.949 | 15 | 57 | +/- 6.822 | NA |
| SSW1 | Integer | 16.025 | 3 | 38 | +/-7.632 | NA |
| ACE1 | Integer | 6.970 | 0 | 28 | +/- 6.611 | NA |
| DBF1 | Integer | 3.445 | 0 | 50 | +/-3.789 | NA |
| WNR1 | Integer | 31.658 | 7 | 150 | +/-17.362 | NA |
| UFE1 | Integer | 21.992 | 3 | 60 | +/-10.535 | NA |
| BPC1 | Integer | 8.333 | 0 | 24 | +/-4.873 | NA |
| BPW1 | Integer | 3.139 | 0 | 9 | +/-2.079 | NA |
| NPA1 | Integer | 24.156 | 2 | 96 | +/-15.771 | NA |
| NPW1 | Integer | 15.550 | 1 | 56 | +/-10.362 | NA |
| TPW1 | Polynomial | NA | NA | NA | NA | NA |
| ST11 | Integer | 4.920 | 0 | 7 | +/-1.877 | NA |
| ST21 | Integer | 4.873 | 0 | 7 | +/-1.873 | NA |
| ST31 | Integer | 4.961 | 0 | 9 | +/-1.940 | NA |
| ST41 | Polynomial | NA | NA | NA | NA | NA(71) |
| ST51 | Polynomial | NA | NA | NA | NA | NA(97) |
| FSP2 | Integer | 63.464 | 41 | 82 | +/-6.929 | NA |
| FSW2 | Integer | 40.333 | 6 | 101 | +/-18.388 | NA |
| SSP2 | Integer | 36.536 | 18 | 59 | +/-6.929 | NA |
| SSW2 | Integer | 16.409 | 2 | 47 | +/-7.617 | NA |
| ACE2 | Integer | 7.279 | 0 | 36 | +/-6.709 | NA |
| DBF2 | Integer | 3.199 | 0 | 13 | +/-3.99 | NA |
| WNR2 | Integer | 31.376 | 2 | 95 | +/-15.608 | NA |
| UFE2 | Integer | 22.451 | 2 | 74 | +/-11.915 | NA |
| BPC2 | Integer | 7.667 | 0 | 25 | +/-4.676 | NA |
| BPW2 | Integer | 3.068 | 0 | 9 | +/-2.154 | NA |
| NPA2 | Integer | 24.219 | 1 | 82 | +/-15.903 | NA |
| NPW2 | Integer | 15.814 | 0 | 50 | +/-10.448 | NA |
| TPW2 | Polynomial | NA | NA | NA | NA | Unknown |
| ST12 | Integer | 4.827 | 0 | 7 | +/-1.889 | NA |
| ST22 | Integer | 4.823 | 0 | 7 | +/-1.880 | NA |
| ST32 | Integer | 5.026 | 1 | 9 | +1.804 | NA |
| ST42 | Polynomial | NA | NA | NA | NA | NA(71) |
| ST52 | Polynomial | NA | NA | NA | NA | NA(97) |
| Gender | Binomial | NA | NA | NA | NA | Female(122) |

**Other information**

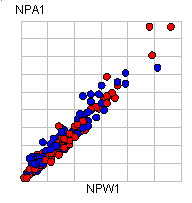
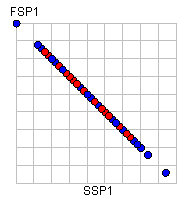
* There are a total of 237 rows of data
* 40 regular attributes
* 1 special attribute
* The class label is “result” and I set this column to binominal on importing the dataset
* Most of the integer attributes are related to scoring in the match so are of the ratio scale
* The “round” attribute is only ever 1,2,3,4,5,6 or 7 so is discrete and of ratio type
* There is a 40 in “round” attribute which I am presuming was an error and will be dealt with later
* The “result” attribute is only ever 1 or 2 so is discrete
* The attribute “gender” is text type and is categorical
* The ”player 1” and “player 2” attributes are of type text and are nominal as they allocated a name

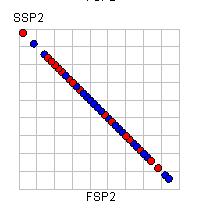
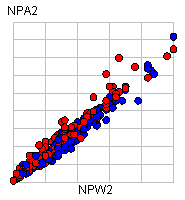
**Initial Exploration of the data**

**Correlated attributes**

Having run a scatter matrix on my dataset I found that the following attributes were highly correlated.

* FSP1 & SSP1
* NPA1 & NPW1
* NPA2 & NPW2
* SSP2 & FSP2





These findings will later lead me to delete one attribute from each pair from the dataset as they essentially hold the same information and so we do not need both of them for our model.

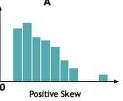
**Attribute irregularities**

* TPW1 and TPW2 have no useful information
* FSW1 has a very large range of 9-102
* WNR1 has a very large range of 7-150
* NPA1 has a very large range of 2-96
* WNR2 has a very large range of 2-95
* Round has a peculiar range of 1-40 while having an average of 2.084 and a deviation of 2.799, this is strange data.

**Using a histogram to find skewed attributes**

The following attributes have a positive skewed histogram curve:

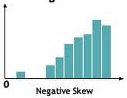
Example:



* Round (as expected) but with interesting outlier
* ACE1,FSW1,DBF1,WNR1,UFE1,BPC1,NPA1,NPW1,FSW2,SSW2,ACE2,DBF2,WNR2,UFE2,BPC2,NPA2,NPW2
* Some of the attributes of course will be skewed by their very nature. For example ACE1 and ACE2 are skewed. This is normal as you would expect there to be more instances of a small number of Aces than instances of very high, as they are not easy to get in Tennis!

The following attributes have a negative skewed histogram curve:

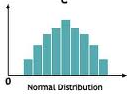
Example:



* ST11,ST21,ST31,ST12,ST22,ST32

The following attributes have a normal histogram curve:

Example:

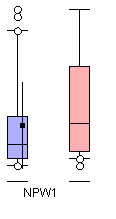


* FSP1 with possible outliers
* SSP1 with possible outliers
* FSP2 with possible outliers
* SSP2 with possible outliers

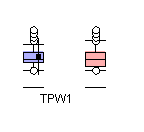
**Predictive attributes**

I used box plots under the Quartile colour matrix heading in Rapid Miner to try and pick out some predictive attributes. The method for doing this is essentially to set “result” as the colour. Then it will show up 2 box plots per attribute. One for a win and one for a loss. If under one attribute the box plots differ a lot, then that attribute is a good predictor!

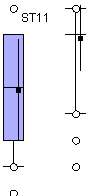
**Results**



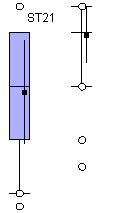
NPW1 has box plots that differ considerably so is a good predictor. It also can be seen to contain some outliers



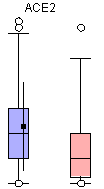
TPW1 can also be seen to differ considerably when comparing each box plot so must also be a good predictor.



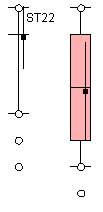
ST11 has box plots that differ considerably so must be a good predictor also.



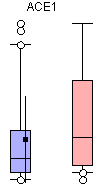
ST21 has box plots that differ and so is a good predictor



ACE2 is a good predictor also



ST22 is a good predictor



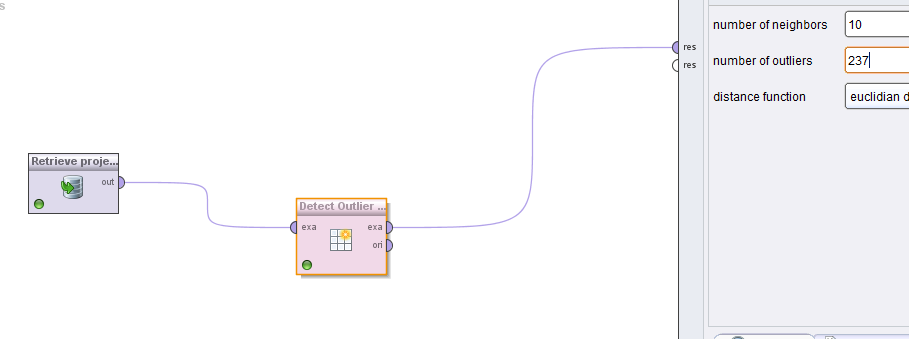
Finally, I believe ACE1 to be a good predictor based on the box plots

**Does the dataset have missing values?**

* All 237 are missing values in the dataset
* This figure is quite a lot
* Attributes TPW2, ST32, ST42, and ST52 are all missing a large amount of values.

**Are outliers a problem?**

* Nearly all the attributes have outliers
* These could be an issue or could simply be normal data that happens to be one extreme for example in a group of 100 people maybe only one or two have a bank balance of over €50,000. The fact that there are a couple of people over this does not necessarily mean the data has errors, further investigation of the outliers is needed to be certain.



My detection of outliers was done using the “detect outlier tool” as well as using histograms to check for the atypical “bell curve” or more so lack of it. I allowed the outlier detection tool to check the 10 neighbours nearest the particular rows attribute to see if we had outliers. There were a vast number of outliers as expected from my histogram analysis which showed the same thing.

**Noise or irregular data**

I found a few instances of what I believe is noise or irregular data, including of course our above outliers. Naturally the next few steps in this project will involve dealing with these outliers and noisy values. Some of these were as follows:

* A value for “round” set at 40.
* A value for WNR1 set to 150, more than double any other instance of it.

**Are they sufficient attributes and examples to achieve my goals?**

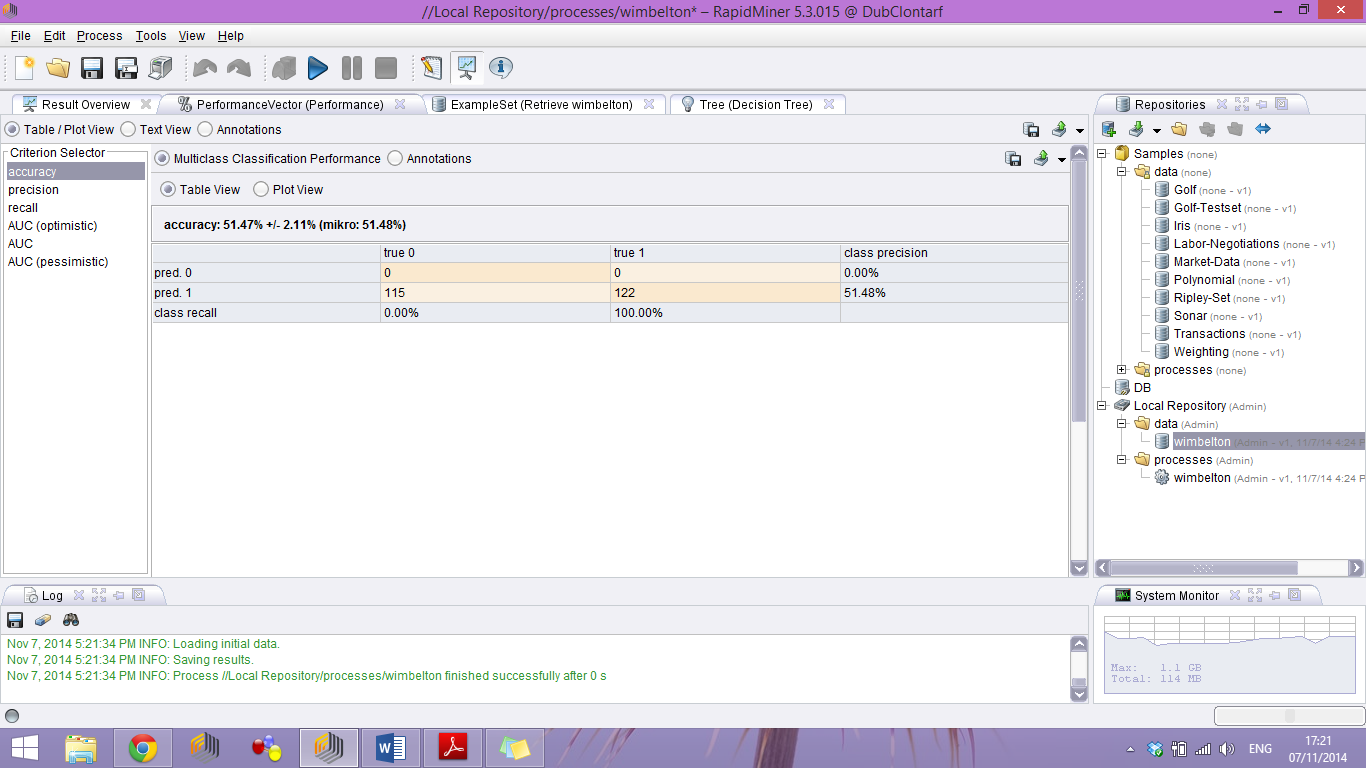
* There are 237 rows of data with 40 attributes and 1 special attribute. Ideally you would actually want more rows than that. Usually you would need for example:

(Attributes) x (Attributes) = 41x41= 1681

* We can see that we are a long way off having enough rows.

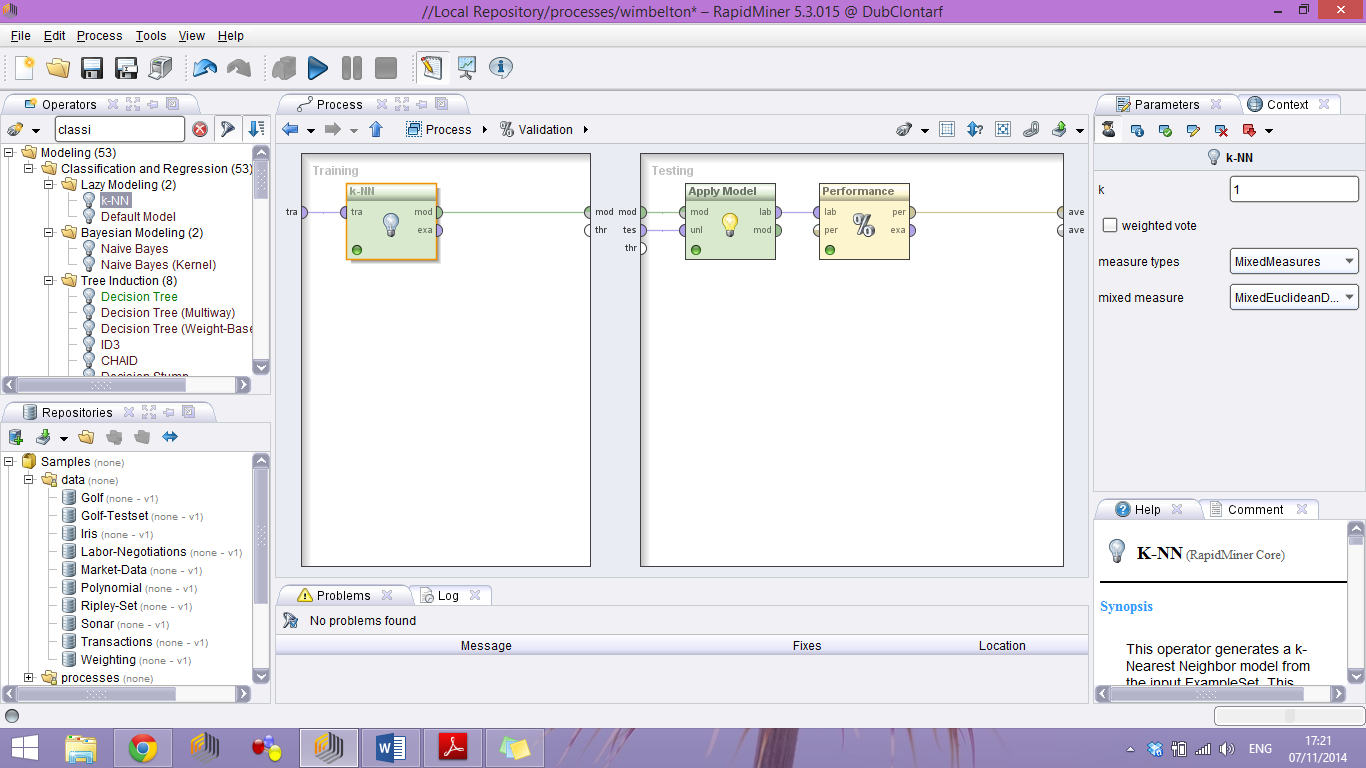
## CRISP-DM Phase 3 (Data Preparation)

Having run the dataset using the default decision tree this was the output.

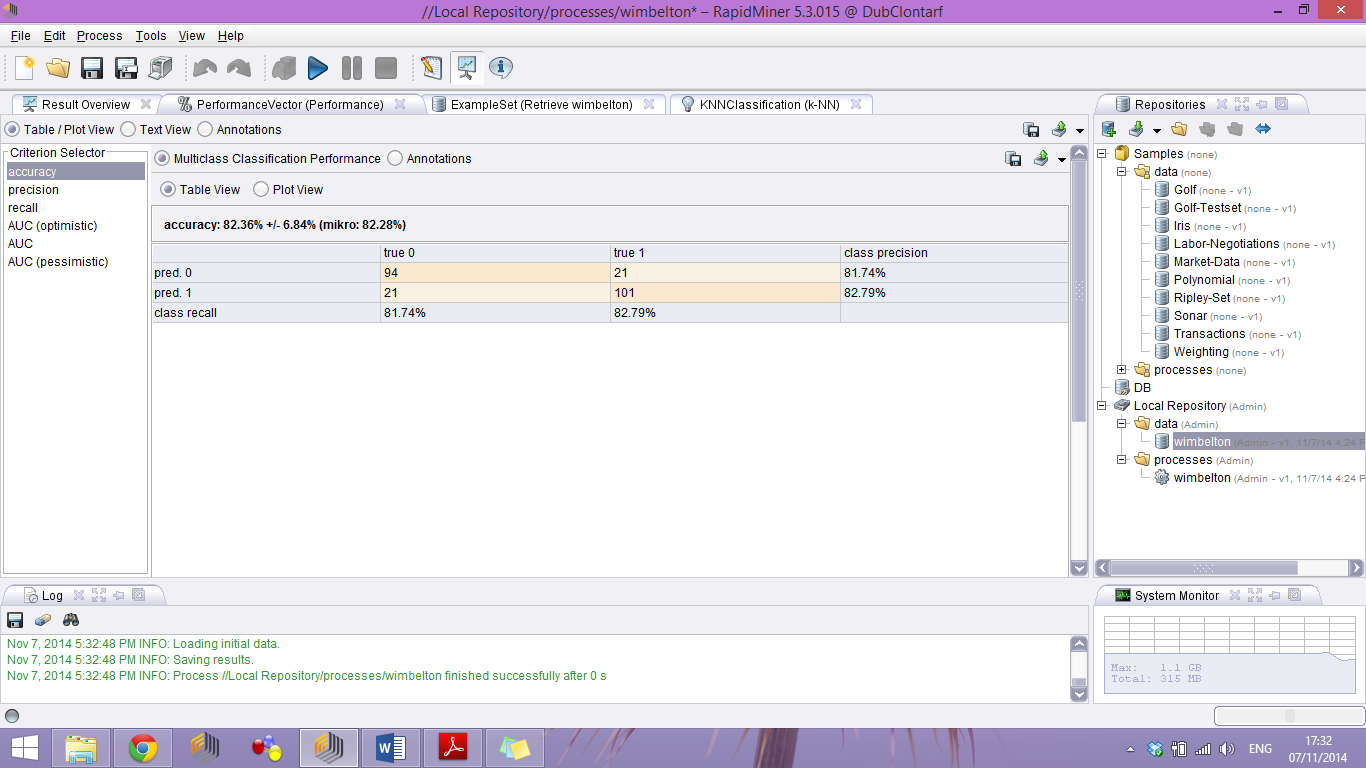


The accuracy I got from this set up was 51.47%

I then ran the model again but this time using Knn.



Having run this I got an accuracy of 78.95% with k as 1. Essentially with 1 being my value for k as in the test sample and a value for n being nearest neighbour to my sample, its value dependant on what I choose. I chose this In part due to my issue with the decision tree but also as kNN is one of the simplest algorithms there is, along with the fact it can be used for both regression and classification



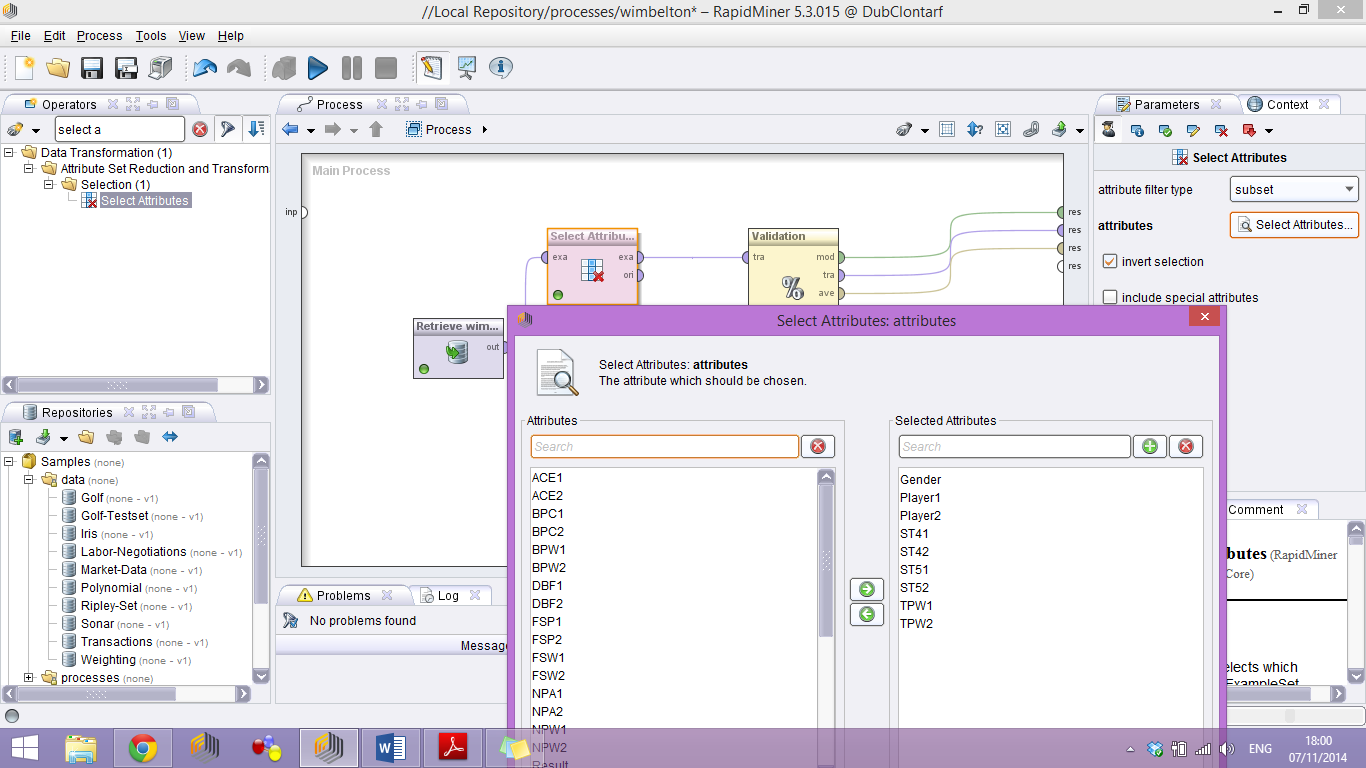
Putting k= 27 gave the best accuracy of 82.58%.

**Reducing rows and columns**

There was need to reduce some of the rows and columns. The following table outlines what was removed and why.

|  |  |  |
| --- | --- | --- |
| Attribute | % Missing | Action |
| Round | Irrelevant to prediction | Removed |
| Player 1 | Irrelevant to prediction | Removed |
| Player 2 | Irrelevant to prediction | Removed |
| TPW1 | 100% | Removed |
| TPW2 | 100% | Removed |
| ST41 | 51% | Removed |
| ST51 | 51% | Removed |
| ST42 | 51% | Removed |
| ST52 | 51% | Removed |
| ST31 | 35% | Replaced Data |
| ST32 | 35% | Replaced Data |
| NPW1 | 20% | Replaced Data |
| ACE1 | 0.016% | Deleted Rows |
| ACE2 | 0.016% | Deleted Rows |
| DBF1 | 0.0042% | Deleted Rows |
| DBG2 | 0.0042% | Deleted Rows |
| Gender | Irrelevant to prediction | Removed |

**Deleting the appropriate columns**



To get rid of some columns from our model I added in the select attributes operator and connected to my retrieval operator. I then selected under “attributes” in the right panel “subset”, so I could remove a number of attributes, they can be seen in the image above. Those attributes are now gone from my model and I notice that the accuracy has gone up a little, less than 1% but it is up. It is also up when run with the decision tree rather than kNN algorithm, by a large amount of 27%.

**Removing correlated attributes.**

Earlier I discovered some attributes were correlated, there were 4 pairs.

* FSP1 & SSP1 - I was led to deleted FSP1 from the dataset
* NPA1 & NPW1 -I was led to delete NPA1 from the dataset
* SSP2 & FSP2 -I was led to delete SSP2 from the dataset
* NPA2 & NPW2 -I was led to delete NPA2 from the dataset

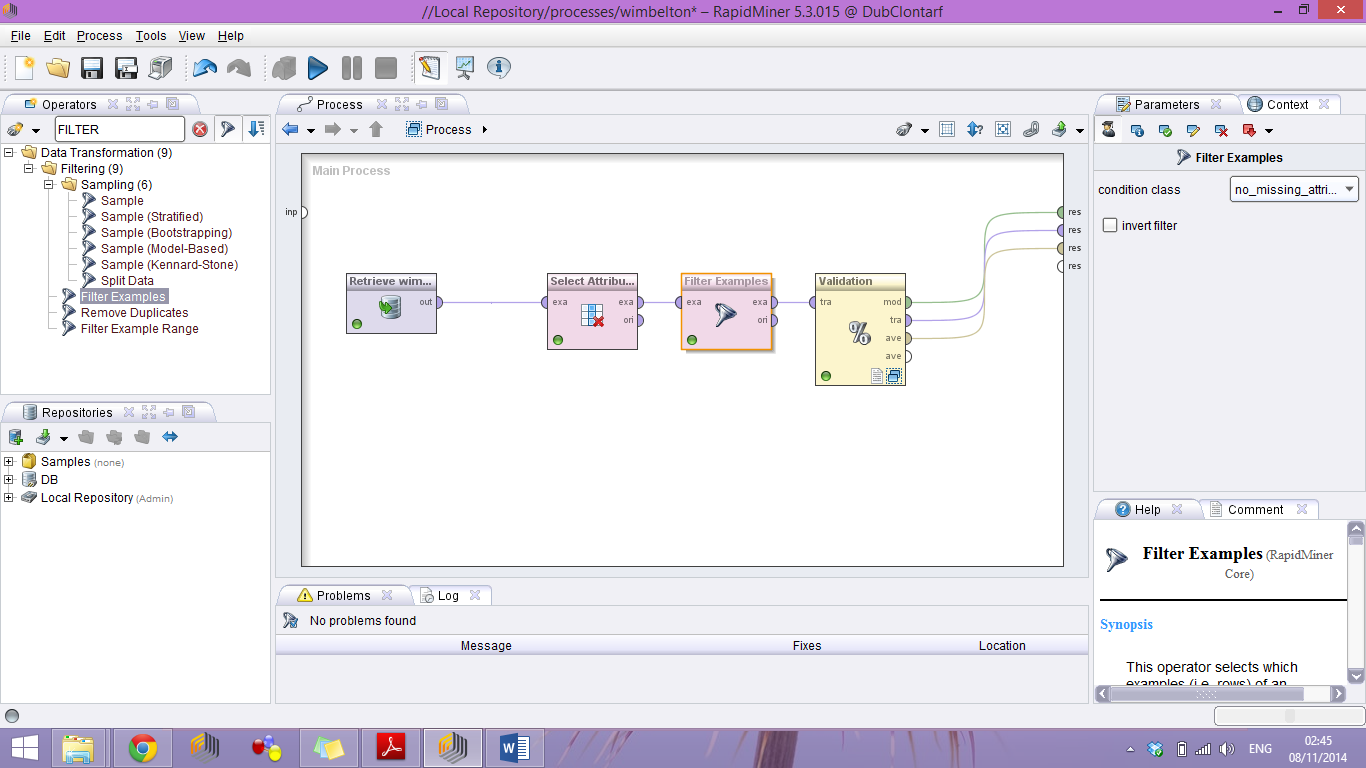
**Replace values**

For some of the rows we can simply replace the values that are missing. Ii do it using an average which I feel is appropriate. We set the “attribute” parameter to “subset” and choose which attributes we wish to replace values for.

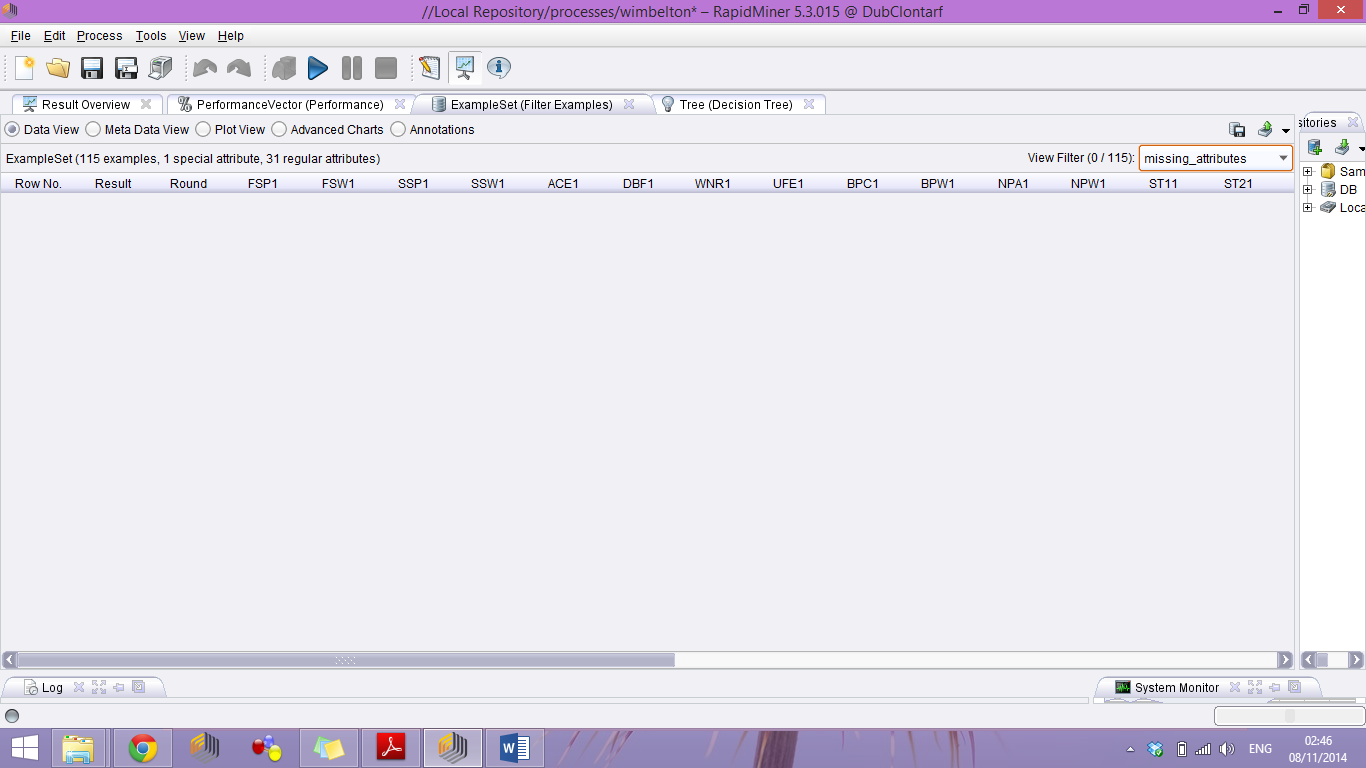
We are now left with 237 rows, but still have 5 rows with missing values.

**Removing some rows**

We need to remove a certain number of rows from the database where there are still missing values. To do this I added a “filter examples” operator and selected the class as “no\_missing\_attributes”.



When this was then run, it took a bit longer than usual, but there are now only 115 rows in the dataset and there are 0 missing attributes. The accuracy has increased to 84.17%.

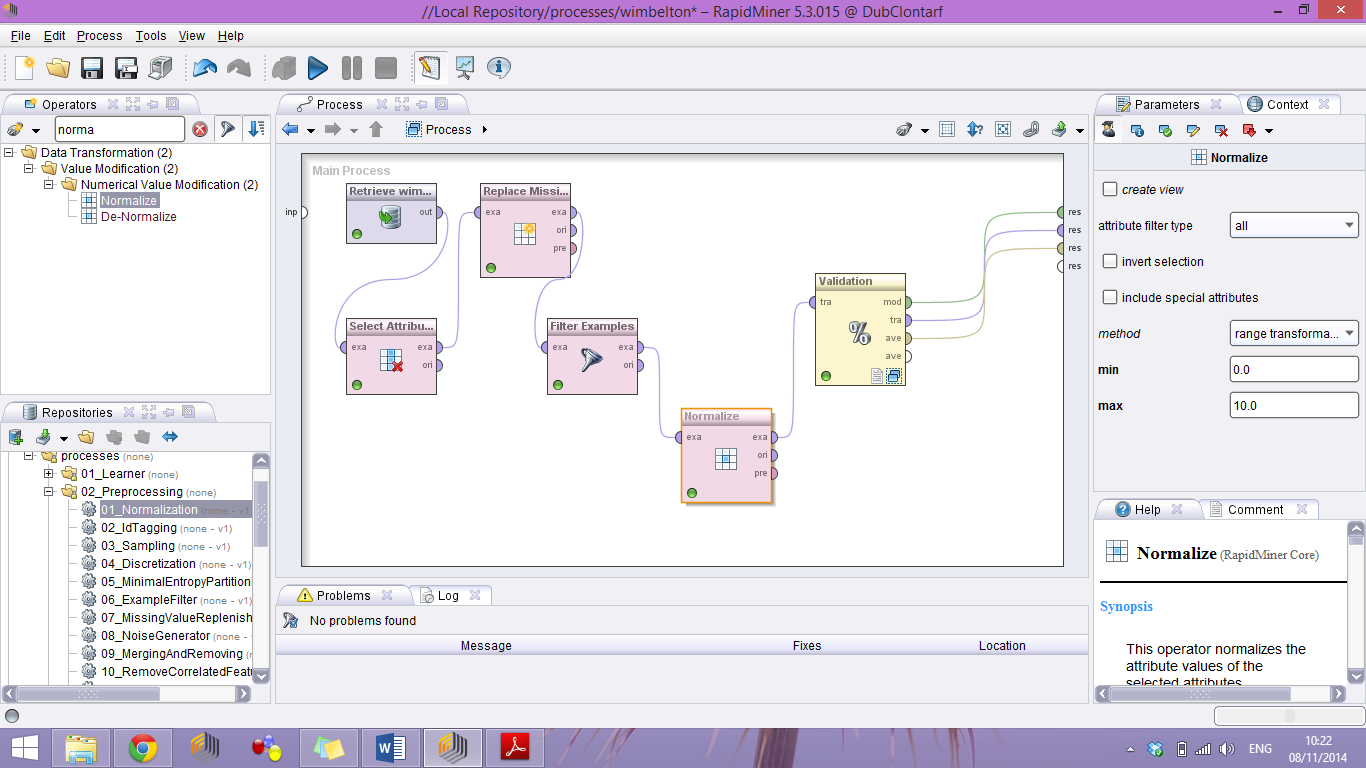


**Tweaking the X-Validation to optimise accuracy**

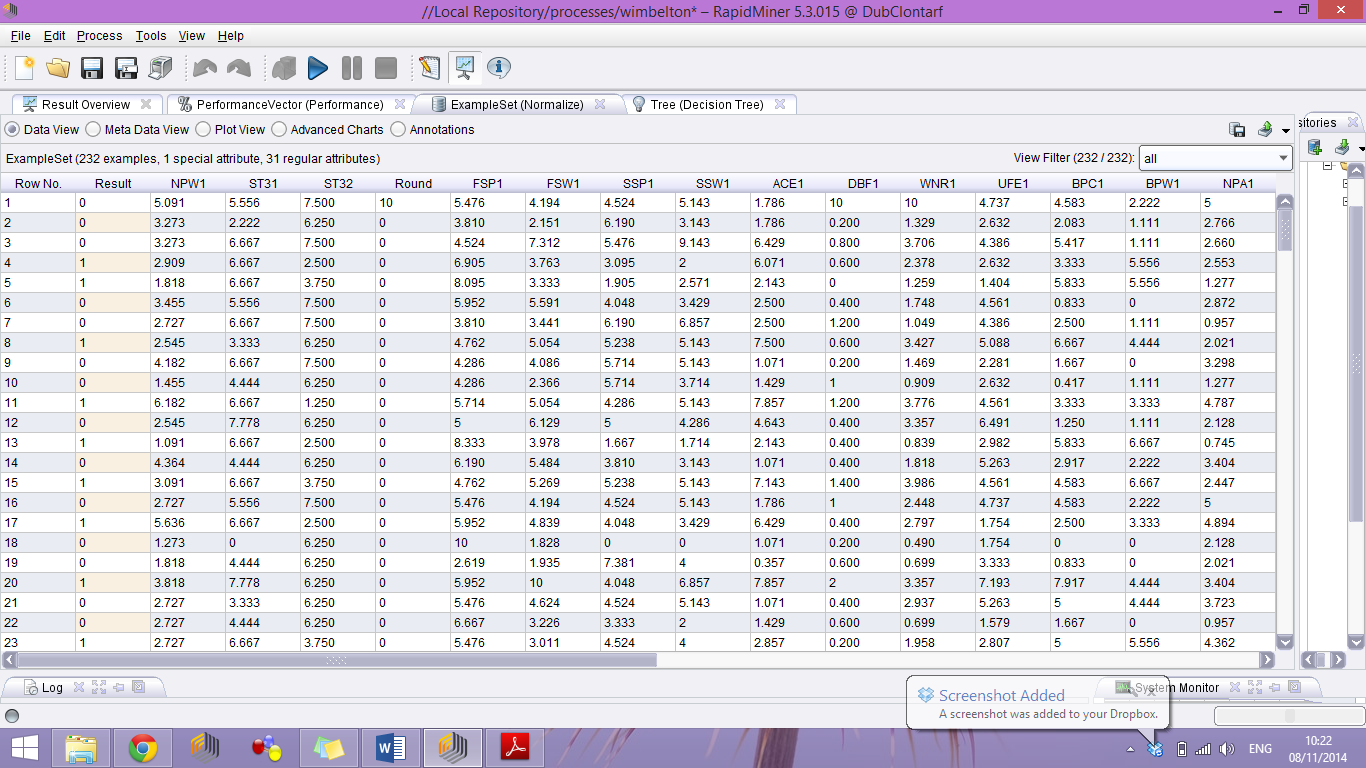
I found that tweaking the parameter “number of variations” to 12 from the default 10 increased accuracy by 2%.

**Normalising**

I noticed in part 1 that some attributes have very large ranges. This can skew data for some numeric based algorithms like kNN so I have decided to normalise them. I set the range for all attributes to 0-10. This step is important for kNN as they uses a distance calculation as its algorithm.



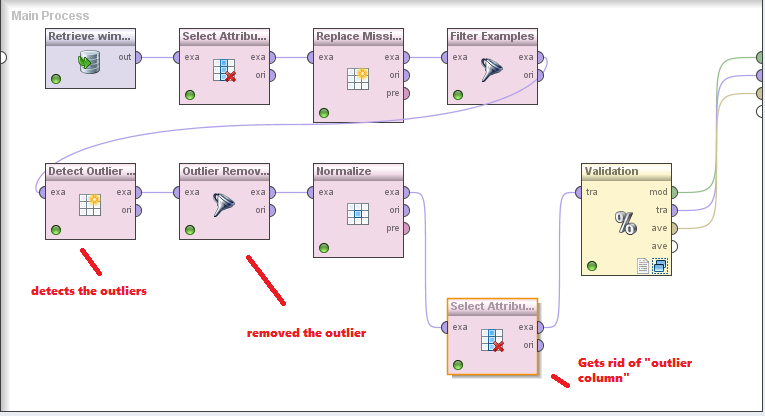
This figure shows the normalise operator. Set to range transformation between 0 and 10.



You can see the attributes are now all between 0-10. The data has been normalised.

**Detect and remove outliers**

I noticed there was a few main outliers. These were too large to leave in the dataset. To remove, I ran the “remove outlier distance” operator with the “number of outliers” parameter as 9 and the “distance” as 15. This shows up the 9 worst outliers (this gave the greatest accuracy). I then added in a “filter attributes” operator and set the condition class to “attribute value filter”. Setting all outliers with a value of “true” to be filtered from the database. I then had an extra column that I did not need. Everything left has an outlier value as “false” as we got rid of all the “true”. I add a “select attributes” operator and use a “regular expression” to get rid of the column under the given parameter “outlier” It is now gone, along with the outliers I detected!

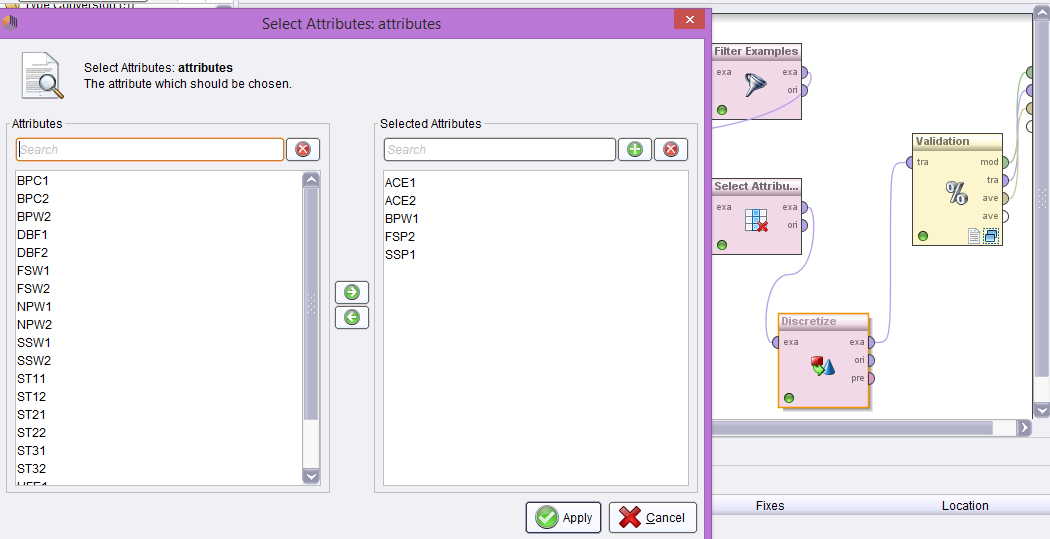


**A check on accuracy**

I now switch back to kNN as our model and our accuracy increases to 95.54%. I deduce this is due to its efficiency in using datasets with all numeric values, which has been well normalised and prepared. Of course there are some outliers left, but I feel this are not anomalies but actual if unusual data.

**Discretize by Binning**

I decided that ACE1, ACE2 and the stats for first and second serve speed should be binned into groups of “low”, “medium” and “high”. For example a 9.0 on the scale would give a value of high for ACE2 as it would mean a lot of aces. I did this by adding the “discretize” operator. I chose the “discretize by user specifications” example. I then created 3 groups as I said. I applied this and the result was as following. In the end I also decided that BPW1 had a lot of values as 0, so I added it into my binning also. This process increased accuracy to 96.44%, the highest so far.



**Optimising before final model**

I noticed a few things from my analysis so far. I decided I would make some final tweaks before I applied my model/models.

**Reduced sample size**

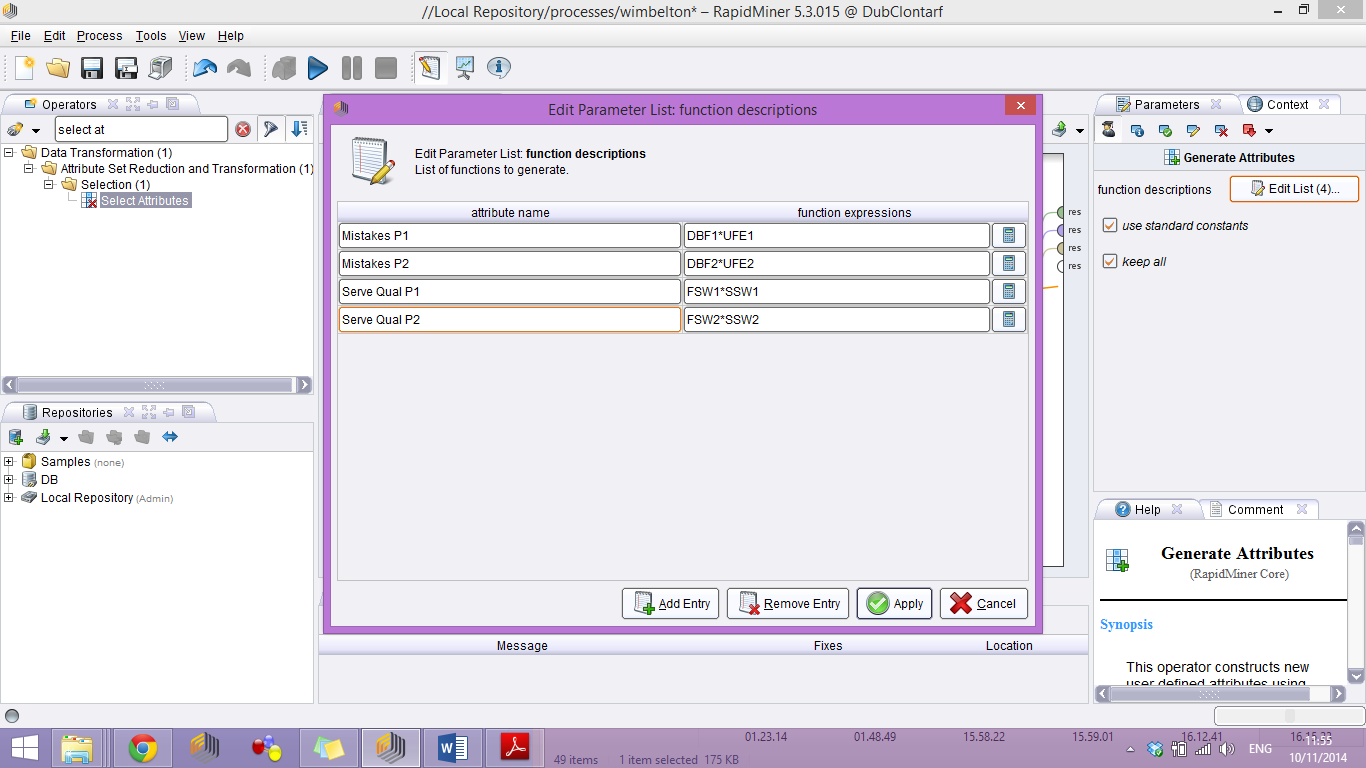
I found reducing the sample size to 90 had no effect on accuracy, anything below that and it started to decrease so I lowered it to 90.

**Change my binning range**

I also noticed that my binning range was causing a little bit of a decrease in accuracy as opposed to other values. I had 0-10 but I decided I would change it to 0-1. This increased accuracy by 1%.

**Generated new attributes**

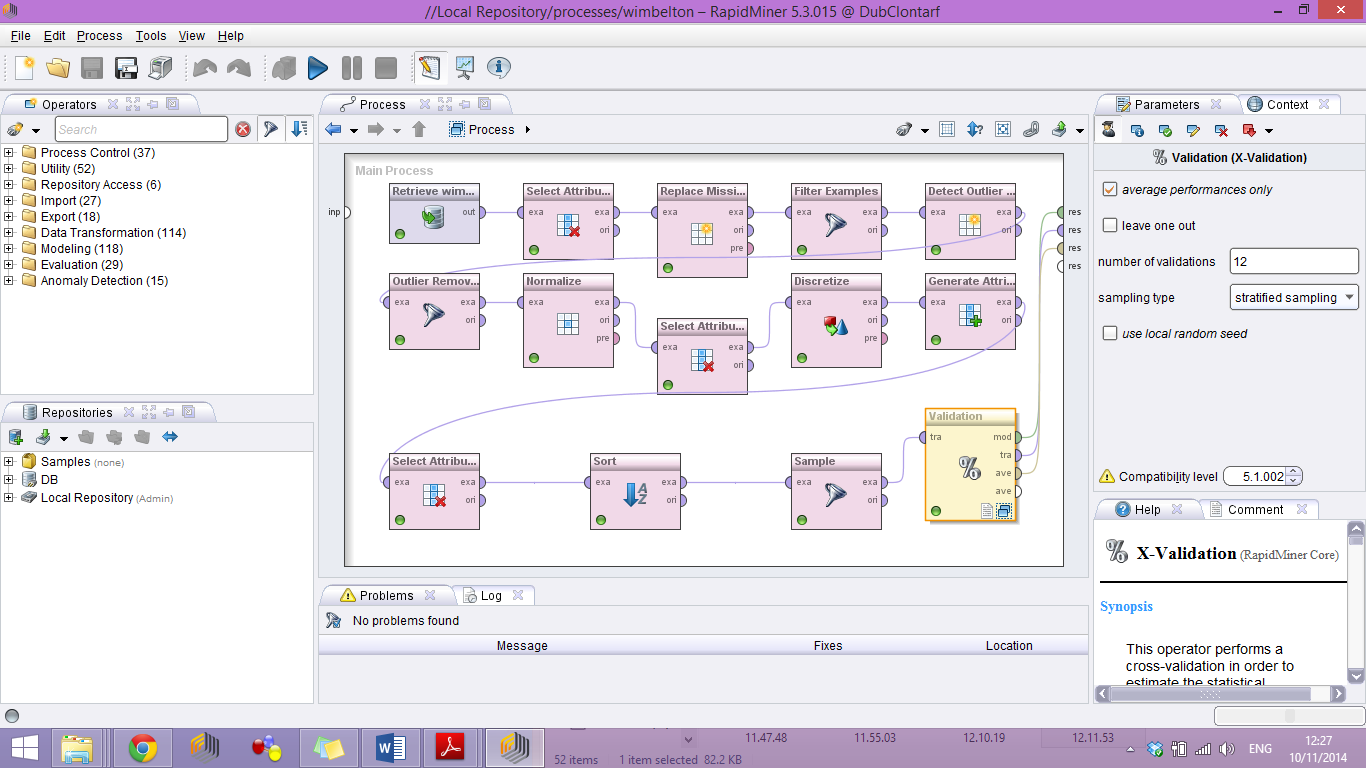
Having reviewed the dataset I decided I would generate some new attributes to improve my overall model. I decided that I would take the attributes for DBF1 and UFE1 and make a column called mistakes for player 1. Then the same for player 2. I could then delete those 4 columns having created 2 new ones, decreased my amount of attributes but not affecting accuracy. I also decided that serve quality could be more efficient. I added a new attribute called serve quality, and took the attributes for first server won and second serve won multiplied by each other as its value. I did this for both players so I had a new column for each, but could delete 4 columns in total. The accuracy did not go down. I think these steps make reading the output and dataset much easier. Fewer rows without decreasing the accuracy.



Above are my functions for this.

## CRISP-DM Phase 4 (Modelling)

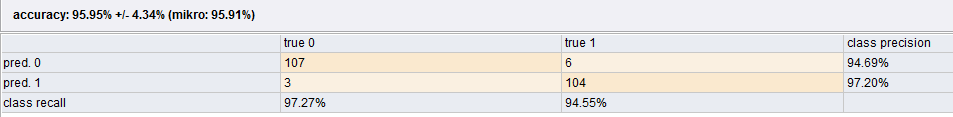
Having gone through the first 3 steps of the CRISP-DM process, I am now ready to model. I carefully analysed each technique, and came to the conclusion that first techniques I would use would be Neural Networks and Naïve Bayes. I first added in the X-Validation Block Operator. I set the number of validations to 12. Sampling type I left at stratified sampling.



I could then move on to applying the modelling technique for these two particular models.

### Neural Network Model

The reason I chose this was firstly that it works well with numeric data. It works well with normalised data, and I felt my data was fairly well normalised at this stage. It can also predict complex relationships which in this dataset may be of use. A big plus is that they are also great with numeric continuous data which I have. I first had to do some conversion. I converted all my data to numerical to get Neural Networks to work.



I then connected up the Neural Networks model operator and used all the default parameters. It ran to accuracy of 94.79%.

**Neural Networks what did I learn?**

It works well with numerical data I found. It is slow to train if not normalised properly however there is a parameter you can tick to normalise for you. It ran at 95.95% accuracy but was slightly less accurate when it predicted a win.

**Improving Neural Networks**

I optimised it by increasing the learning rate from 0.3 to 0.4 and accuracy went up by nearly a per cent. For efficiency and faster training times I decreased the training cycles from 500 to 400 which did not affect accuracy but it ran faster. Accuracy increased to 95.49% however class precision and recall lowed a very small amount.



### Naïve Bayes Model

Although my model worked to a high level of accuracy for Neural Networks, over 95% the modelling technique does tend to take quite a long time to train. As a beginner to data mining I also felt that I would like a model that can handle noisy data very well, as it was always very possible that I had not done the data cleaning phase perfectly and the model may struggle. Naïve Bayes offered me a good alternative. It is very quick to train and handle noisy data very well. It prefers categorical data but can work with numerical or mixed also without too much issue. The performance can be degraded if there are correlated attributes but I was very confident that I had eliminated all of these so did not feel it would affect model in that regard.

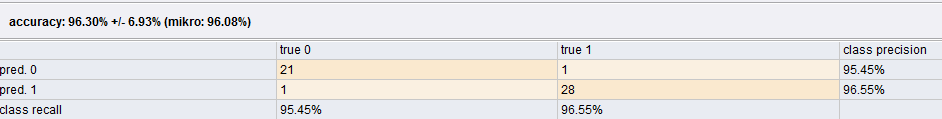


**Naïve Bayes what did I learn?**

The accuracy of this model was 93.65%. I had to increase the sample size to 200, any less the accuracy rapidly decreased. It prefers categorical data but seems to work quite well with numerical anyway. In contrast to neural networks it seems to predict a loss more accurately than a win.

**Improving Naïve Bayes**

I first lowed the sample size to optimise it. It could go as low as 65 without being affected but I decided on a conservative 75. I then changed sample probability to 0.2 which essentially just changed the sample probability for each class, a minor tweak but it did increase accuracy by half a per cent. I then changed the number of validations on the X-Validation to 9. This increased accuracy to over 96%. I was wary of doing anything else as I did not want the model to become over-fitted.



**Some overall Improvements**

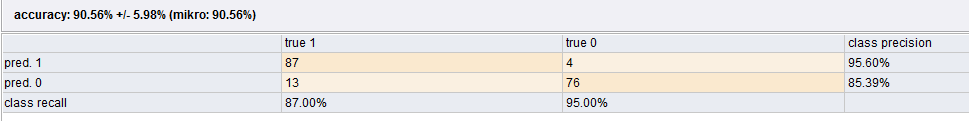
Accuracy levels for Naive Bayes and with Neural Networks were about as high as you can get without risking the model becoming over-fitted. I do however feel that there I still some noisy data that could be removed. It is difficult in relation to tennis scores as I feel that although something may seem to be an outlier or noisy data, sometimes it is just a particularly spectacular performance from a player that creates this unusual data. Should this be removed? I do not believe it should. I do however feel the model could be optimised in this way.

* Do not remove certain attributes about performance at the beginning even if they are correlated
* Use these attributes to generate new attributes
* Then remove the original attributes leaving only the newly generated one, which not be correlated with any other.
* Essentially the correlated one goes anyway but not before being used to help generate a new attribute which would be useful.
* You could also in regards Naïve Bayes change the data type to categorical which it tends to prefer.

## CRISP-DM Phase 5 (Evaluation)

In evaluating both this dataset and project I have come to learn numerous things. The most important attributes the models seem to regard are what I would have presumed myself. The number of aces, errors and break points won are what it sees as vital to the result. Of course this is logical, a player with lots of errors is unlikely to have won, and the model reflects this reality. The new attributes I generated were used in calculation, I confirmed this by running a decision tree model and viewing the tree itself, showing my new attributes vital to deciding the win or loss. The model tended to think that break point winners for each player were the starting point. I achieved very high levels of accuracy with both models however the most usable was probably Neural Networks, despite its slow training time it did give very accurate results constantly. It did not require many tweaks unlike Naïve Bayes it seemed to run efficiency under all parameters. From this project I have learned that all datasets will have noisy or unusable data. To build an efficient model you must deal with these. You should pretty much always then look for ways to tweak the model using the various operators available to Increase the models accuracy, while always being careful not to have an over-fitted model, which is essentially useless.

I ran an interesting test when finished. I applied my model to one of the other datasets. I was wary of over fitting of my model having achieved a high level of accuracy. On running the neural networks I attained an accuracy of 80%. I ran it with Naïve Bayes and attained an accuracy of 90%.



So, My final conclusion is that of the two models I tested, Naïve Bayes although has an accuracy of 1% less, it still is very high, and seems to be around 90% when applied to other datasets too, I feel this is an acceptable conclusion to the project, and although some changes can be made to the model to make it even more flexible it has worked out reasonably well. (Sample size had to be reduced due to dataset specifics)

**Final Model:**

* Naïve Bayes. 96.30% accuracy
* Class precision 95.5% average
* Fast to train
* Not over fitted, 90%+ accuracy on all data sets tried