

David J. Aumueller¹
MSc in Communications Studies
May 08, 2003

13 pages

Table of Contents

Data Mining in the School of Computing.....	1
Task 2.2 Describe data (30 mins) (5 marks)	1
Task 2.3 Explore data (1 hour) (5 marks)	2
Task 2.4 Verify Data Quality (5 marks)	3
Task 3.1 Select Data (5 marks)	3
Task 3.2 Clean Data (5 marks).....	4
Tasks 3.3/3.4 Construct Data and Integrate Data (2 hours) (10 marks)	5
Task 3.5 Format Data (1 hour) (5 marks)	6
Task 4.1 Select modelling technique (10 min) (0 marks: possible bonus marks)	7
Task 4.2 Generate Test Design (20 min) (5 marks)	7
Task 4.3 Build Model (2 hours) (5 marks)	7
Task 4.4 Assess Model (15 mins) (5 marks).....	7
Task 5.1 Evaluate Results (20 minutes) (10 marks)	8
Tasks 5.2/5.3 Review Process/Determine Next Steps (20 minutes) (5 marks)	8
Tasks 6.1 - 6.3 Deployment (1 hour) (30 marks).....	9
References	10
Appendix.....	11
Tree using all columns but sid	10
Tree using average marks only	11
Rules using average marks only.....	12
Tree using SID only	12
Rules using COMP modules only.....	13

Data Mining in the School of Computing

Task 2.2 Describe data (30 mins) (5 marks)

Table degree_programmes

The table ‘degree_programmes’ (10 columns) lists all the available programmes of study across the departments of the university. Each programme has a distinctive ‘prog_code’, a title, an ‘scs_code’ (eg DB32), the ‘degree’ (eg BSc, BA), a ‘type’ (eg SS for single subject), the belonging department as ‘owner’ (eg COMP), and a field to indicate the ‘current’ status of the programme, i.e. whether it is running this year or

¹ ics2dja@leeds.ac.uk

not, or in the next year or not anymore at all. All fields are of type varchar; except 'prog_code' all columns may contain NULL-values. There are 295 tuples.

Table modules

The table 'modules' (10 columns) lists all modules with distinctive 'module_code', the 'scs_code', the 'title', the amount of 'credits' (integer), the 'semester' (integer) of the year in which it is taught, to which 'level' it is assigned (eg 1-5), whether it is a 'project' or not. If not stated otherwise, all fields are varchar. Except 'module_code' all columns may contain NULL-values. There are 607 tuples, 159 COMP-modules.

Table students

The table 'students' (8 columns) contains the anonymised students' sur-/fore-/usernames, their student ID (sid) which can be used to reference a student in the table 'student_modules' and some personal background: gender, age and country of origin. All fields are varchar; there are 2437 tuples.

Table student_modules

The table 'student_modules' (4 columns) contains the mark a student received for each module, including the final year project. Except the field 'year' which is of type integer, the datatypes for SID, the 'module_code' and the 'result' (mark) are varchar. There are 39252 tuples.

Tables edass_assignment, edass_mark

Table edass_assignment lists all single courseworks/exams for each module with their individual weight (383 tuples). Table edass_mark lists all the marks achieved in each single assignment by all the individual students (53895 tuples). These two tables are not considered too deeply since with the table student_modules an enough detailed data of student marks is available. Thus, they are not described further.

Task 2.3 Explore data (1 hour) (5 marks)

Generally, I will only look at modules offered by the School of Computing; they can easily identified by their module_code starting with 'COMP', followed by a digit which indicates the year the module is taught (1st, 2nd, ...).

To see how many students sat each module each year the following query was used:

```
select year, module_code, count(*) as ppl
from student_modules
where module_code like 'COMP%'
group by module_code, year
order by ppl, year, module_code
```

This result will be of help when selecting the data since it will not be very useful to look at modules with only few students. On the contrary, it is anticipated to use subjects/modules with many students AND modules which are 'consistent' over some years.

To see what percentage of final year projects are awarded a first class or upper second mark year-by-year the following query was used to get the number of students. Since the 'result'-field contains not only integer-values I used a for-loop in Perl to help me build the query-string.

```

select year, count(*) from student_modules where module_code in
(
select module_code from modules where module_code like 'COMP%' and
project like 'Y' and level = 3
)
and result in ('60', '61', '62', '63', '64', '65', '66', '67',
'68', '69', '70', '71', '72', '73', '74', '75', '76', '77', '78',
'79', '80', '81', '82', '83', '84', '85', '86', '87', '88', '89',
'90', '91', '92', '93', '94', '95', '96', '97', '98', '99', '100')
group by year

```

Table 1

year	students	percentage
1995	00 / 001	only one student on a project (but result=0).
1996	70 / 108	64.81481
1997	75 / 128	58.59375
1998	73 / 124	58.87096
1999	79 / 143	55.24475
2000	63 / 120	52.5
2001	95 / 160	59.375
2002	00 / 174	only NULL values in results (not finished project yet).

The overall percentage is 47.495% (455 of 958 students who have done a final year project).

This task made me aware of some issues of data quality. Quality will be examined further in the next step, listing some issues.

Task 2.4 Verify Data Quality (5 marks)

- There is for example a referential inconsistency, in that respect that some (188) students are not in any modules:

```

select count(sid)
from students
where sid not in
(
select distinct sid from student_modules
)

```

- There are 3 students that do not have a proper gender assigned. From a total of 2437 students, 1958 male, 476 female and 3 '?'.
- There are 434 students with no age (NULL) assigned, one is supposedly 0 years old, another one 100.
- Concerning country_of_orign there are 3 students having an empty string "" assigned, 3 with '???' and 1854 students with '000' (is it UK?).
- As seen in task 2.3 the Marks..student_modules.result is not numeric but will be cleaned later using the Marks..grade_map table. (Some numbers are followed by a character, other values are pure alphas such as NSAX or ABS – those will be changed.) Similar cleaning may have to be done for the edass_mark table. Here it seems that even some other characters such as '-' has to be changed manually.

Task 3.1 Select Data (5 marks)

Using some temporarily created tables in the database I filtered the available modules to the ones having more than 29 students *and* being stable over the highest amount of

years. This was achieved² by a very wide inner table join over the temporal tables³ consisting of the modules for one year each. Thus I was able to reduce the total of 159 COMP-modules to 26 modules that I consider relevant being stable over the years 1999 – 2001 (see Table 2). Year 2002 cannot be used since no project results are available.

Indicators of whether a student will do a good final year project might be:

Table 2

Field	Motivation
Age	Younger students might be less serious with their studies since usually they do not pay themselves for their study and prefer 'hanging out' with friends than studying hard.
Country of Origin	Overseas students pay more and thus might be more serious (code <> 000 ?). Very vague.
Gender	Female students might be more serious about their study since they are said to be mature earlier and boys get more easily distracted...
SID	This is very vague / hypothetic, but SID might be an indicator insofar greater numbers could stand for later matriculation which again might point to students being not too serious about their study at all (deciding late for a course, applying late, matriculate late).
Average mark over all modules taken by one student	This is considered a very strong indicator. Students being good overall should do a good project as well.
COMP modules	COMP1 and COMP2 will be considered generally. Since I do not know anything about their content I cannot distinguish specific suitable modules. But as mentioned above, I looked at quantity of students and consistency of modules to limit the amount of modules to the following ones: COMPx with x in 1300, 1360, 1400, 1450, 1500, 1510, 1600, 1610, 1620, 1650, 1660, 2300, 2350, 2360, 2370, 2380, 2400, 2450, 2470, 2550, 2560, 2600, 2610, 2620, 2650, 2660.

I also thought about using the edass-tables to look at the marks of each assignment by each student. Since these marks are not weighted they would have to be normalised first (after cleaning the data e.g. with the grade_map table, of course). This was considered not worth the effort since with the module marks an enough detailed indicator exists.

Task 3.2 Clean Data (5 marks)

- Incorporating the mapped values from the grade_map table a new student_modules table is created having numeric result values instead of the varchar result field, using the following SQL:

² As another output-example of this approach, the following are the COMP-modules stable over the years 1998 -- 2001 AND having more than 99 students: 1300, 1360, 1400, 1450, 1500, 1510, 1600, 1610, 1620, 1650, 1660, 5010, 5060, and 5200.

³ If interested, the tables still exist in the database DB32_ics2jda.

```

select sid, module_code, year, numeric_grade
into student_modules_clean
from Marks..student_modules SM, Marks..grades_map GM
where SM.result = GM.result

```

- In the field for country of origin, there is one instance having '???' which has to be changed manually into a numeric value like '-1' or '000'.
- If student names would not be anonymous one could try to guess the gender of the unassigned values in this field.

Tasks 3.3/3.4 Construct Data and Integrate Data (2 hours) (10 marks)

As mentioned above, I will look at the average of marks by each individual student. This is a derived attribute constructed with the following SQL-bit:

```

select sid, AVG(numeric_grade) AS average
into student_modules_averages
from student_modules_clean
where module_code not like 'COMP3510'
group by sid

```

Since the group by-statement does not take into account NULL values for calculation this operation results in proper averages. Note that the project has to be excluded. Looking at students with high average marks (e.g. ≥ 60) this attribute seems already to be a very good indicator: only very few (in this example only two) students have a low project mark, which is significantly more (nearly double percentage) than the ~48% having a good mark in the project as discovered in task 2.3.

After constructing, merging, integrating the data pivoting the data (using modifications of the SQL provided) results in one table that is going to be exported as comma-separated values for use with the mining-/modelling-tool, here WEKA.

The following lists the SQL-statements used for this task – adopted from the appendix of the coursework brief.

```

select sm.*, gender, country_of_orign, age, scs_code
into selected_results_final from student_modules_final sm
inner join Marks..students s on s.sid = sm.sid
inner join Marks..degree_programmes p on p.prog_code =
s.degree_programme
where sm.sid in (
  select sid from student_modules_final
  where year in (1999, 2000, 2001, 2002)
  and year between sm.year+2 and sm.year+3
  and module_code = 'COMP3510'
  and numeric_grade is not null
)
and type = 'SS' and numeric_grade is not null

```

This embedded select-statement of the SQL above (originally appendix A1) selects all students having a valid result in the project (COMP3510) but only if data for the two to three years before his/her final year of this particular student (of the ones selected by the inner select) is available.

The following SQL (originally appendix A2) adds in the result for the final year project:

```

insert into selected_results_final
select distinct sm.*, gender, country_of_orign, age, scs_code
from student_modules_final sm
inner join Marks..students s on s.sid = sm.sid
inner join Marks..degree_programmes p on p.prog_code =
s.degree_programme
where module_code = 'COMP3510' and year in (1999, 2000, 2001,
2002) and type = 'SS' and numeric_grade is not null

```

Adopted from appendix B:

```

SELECT sid, gender, country_of_orign, age, average,
max(CASE module_code WHEN 'COMP1300' THEN numeric_grade ELSE '?'
END) AS COMP1300,

```

[...]

```

max(CASE module_code WHEN 'COMP3510' THEN numeric_grade ELSE '?'
END) AS COMP3510
INTO for_export_final
FROM selected_results_final r group by sid,
gender, country_of_orign, age, scs_code, average

```

This pivots the selected data in order to get a dataset of one row for each student. That way data is summarized and can be easily analysed. Fields can be easily rearranged changing the view of the table.

Finally, the numeric result values for the project are translated into two class labels, here 'Good' and 'Bad', depending on a chosen threshold of 60:

```

update for_export_final set COMP3510 = 'Good' where COMP3510 > 59
update for_export_final set COMP3510 = 'Bad' where COMP3510 <>
'Good' and COMP3510 <> '?'

```

Task 3.5 Format Data (1 hour) (5 marks)

The data is exported into a plain/text-file containing the comma-separated values. To conform to the syntax needed by WEKA an appropriate ARFF-header is added to the file, listing all the attribute names and their data type, e.g. numeric or nominal.

By default WEKA uses the last attribute in each line as ending leaf in the J48-tree. Surprisingly, the order of attributes affects the shape of the resulting tree! Thus, I decided to order the attributes conforming to my current understanding of relevance. Probably some random-order (or various runs with different random-orders) would be most suitable.

A glimpse into the exported data:

```

average, age, sid, gender, country_of_orign, COMP1300, COMP1360,
COMP1400, COMP1450, COMP1500, COMP1510, COMP1600, COMP1610,
COMP1620, COMP1650, COMP1660, COMP2300, COMP2350, COMP2360,
COMP2370, COMP2380, COMP2400, COMP2450, COMP2470, COMP2550,
COMP2560, COMP2600, COMP2610, COMP2620, COMP2650, COMP2660,
COMP3510

71.33333333,21,7000466,M,0,64,52,88,?,84,83,76,75,66,78,69,?,72,73
,74,79,82,?,?,54,?,81,66,67,68,72,Good

57.97142857,21,7000469,M,0,65,53,60,?,44,51,59,73,64,62,60,?,?,?,4
7,?,63,52,50,?,?,64,53,55,62,62,Bad

74.05714286,21,7000488,M,0,90,68,60,?,76,70,82,91,72,68,89,75,68,8
1,84,69,84,?,?,?,?,84,69,75,68,75,Good

```

Task 4.1 Select modelling technique (10 min) (0 marks: possible bonus marks)

As modelling tool WEKA will be used, using the J48 classifiers j48.J48 for decision trees and j48.PART for rules. Trees will be looked at pruned and unpruned. To help to find correlations clusters will be looked at, as well.

To use association rules data ought to be nominal. Since most data (module marks, averages) are numeric these tools cannot be used without converting the data. The grade_map table would help in converting marks to classifications.

Task 4.2 Generate Test Design (20 min) (5 marks)

The data will not be divided into training data, test data and validation data by hand but I will use the n-fold cross-validation algorithm in WEKA with n = 10 folds (default).

WEKA will report error-statistics.

Task 4.3 Build Model (2 hours) (5 marks)

For parameters all default values will be kept. Only the confidence value which I set up to 0.9 in order to get more accurate trees.

See the appendix for some models.

A first tree gives an idea of which attributes one might look at closer. Attribute average can be seen as good index already. Also, e.g. COMP1600 and COMP1510 seem to be stronger indicators.

To test some hypothesis from above, the classifier will be run with less attributes selected, e.g. with average only the j48.PART returns these rules:

```
average > 53.69697: Good (82.0/21.0)
: Bad (59.0/21.0)
```

Running j48.J48 on SID only returns the following tree/rule, indicating that SID indeed can be an indicator for the project result:

```
sid <= 7000598: Good (97.0/15.0)
sid > 7000598: Bad (114.0/41.0)
```

Interesting seems to be the following rule resulting from a j48.PART run with the COMP modules only:

```
COMP1600 > 64 AND COMP1400 > 65: Good (48.58/6.91)
```

Most of the other rules/leaves do not seem to be very accurate, thus it is hard to find useful or true correlations in this data set. Overall an accuracy of around 65% has to be seen not as the highest that can be reached but that one has to be satisfied in average with. So far it can be concluded that the made assumptions partly are true. Nothing really new discovered.

Task 4.4 Assess Model (15 mins) (5 marks)

Ranking the models concerning the business objective, the rule/tree using the attribute average marks has to be seen most important (68% correctly assigned instances), followed by the SID (73%) which is hard to interpret though. The models showing

specific modules as indicator seem to be of minor importance (57% - 60%) but still show higher correlation than gender, age, or country of origin.

Task 5.1 Evaluate Results (20 minutes) (10 marks)

As total output of this data mining project the result has to be seen as the sum of models plus other findings around them.

The best indicator found is as expected the average overall marks which is very plausible and would not really need to be discovered using DM. Two modules which seem to have a high influence are COMP1600 and COMP1510. Their content should be looked at to interpret the reason for this out-standing of the other modules.

Figure 1 (at the very end) shows two relationships. Firstly, looking at the coded colours (blue for good project, red for bad project) and the average of marks (y-axis) it can be seen that these two fields (average marks and project mark) correlate.

Secondly, an interesting grouping/clustering of the SID (x-axis) takes place. Roughly, there are two groups of SID-ranges and very surprisingly the students in the lower SID-range/cluster have higher mark averages and a good project whereas the higher SID values correlate with a bad project and low average marks.

As hypothesised earlier higher SID values may point to less decided students.

Whether these numbers are relevant has to be checked with the original dataset; the data selecting approach might have an influence (e.g. blend out a whole group of students). Also, anonymising these values might have distorted something here. More likely though is that with a new year of matriculation a new block of numbers is introduced. Interestingly though, the six students with the very high SID are male and did badly in the project. Apart from that, no real indicators could be found in the students' personal backgrounds like age, gender and country of origin.

Looking at some diagrams or visualised clusters in WEKA some modules seem perfectly correlate with the average mark attribute, but a few do not at all, e.g. COMP1450 or COMP1620 are not correlating.

Tasks 5.2/5.3 Review Process/Determine Next Steps (20 minutes) (5 marks)

Most striking has to be seen the fact, that a higher SID correlates with better marks. Reasons for this can only be guessed as aforementioned. Secondly, it is quite satisfying that the derived attribute average marks correlates with good project results as predicted. Disappointing might be that no significant correlation between gender and marks was found.

Further investigations are necessary. Some attributes might have been overlooked a bit, e.g. country of origin did not prove useful in any models. With not knowing the exact code of this attribute it is also hard to investigate further. From the point of view of the SoC itself more attributes might be allowed to be used since no anonymisation is necessary. Also, more data could be collected in the future, such as final marks from former schools for example.

Another idea might be to examine the lecturer/tutor-side, e.g. whether `taught_by.workload` has any influence on specific modules as indicator. If workload is high, the project might be not as good because students could be less motivated being poorly supervised – or contrary – the project might be good because of students being already used to do more unstructured work themselves.

Moreover it would be interesting to look at specific correlations of gender, age to individual modules; basically to do some association rules.

Tasks 6.1 - 6.3 Deployment (1 hour) (30 marks)

Business understanding. The aim of this data-mining project was to find new means to predict whether a student from the School of Computing is likely to be successful in her/his (external) final year project. This project was organised in the framework provided by the CRISP-DM methodology consisting of the six phases summarised in this last phase.

After determining the above business objectives the current situation was assessed, i.e. checking available tools, data, and projected time of completeness. As more exact data mining goal it was decided to examine data on a student's personal background coupled with results in some first or second year modules in order to discover if this data forms a good indicator for the student's final year project.

Data understanding. Any available data was collected and resulted in some tables on a relational database containing data such as student-names, IDs, gender, age, country of origin, modules, student's marks and information about the degree programmes and other. Describing this data helped to understand it and led to explore it more. Thus it was discovered how many students sat each module each year and what percentage of students did have a good result in their final year project, which proved useful in narrowing down the final data to be examined and made aware of some data quality issues. Some fields in the database were assigned varchar although they should be numeric like module results, but since these fields also contained non-numeric characters the data needed to be cleaned.

Data preparation. To finally decide for data to be used some further tests in the form of SQL-queries were run on the database. It was decided to use the personal background information and some specific COMP modules that are stable over the years 1999 to 2002 and always had many (30+) listening. Not only to be able to identify a student but also because of a vague but possible correlation with results, the student-ID was selected, too. As possibly quite strong indicator the average of one student's marks were calculated as derived attribute.

As mentioned above some data had to be cleaned, e.g. the non-numeric results had to be mapped to numeric ones before any calculations on them were possible.

The selected data was merged into one table by pivoting the data into one row with all the attributes for each student. The chosen modelling tool WEKA needs its data to be input from a special text/plain-file, thus the final table was exported to comma-separated values and final adjustments were made manually.

Modelling. Data was classified using the decision-tree finder J48 and its rule finder j48.PART. WEKA supports randomised division between test- and training-set using the n-fold cross validation algorithm. Some trees were modelled, and re-modelled having changed confidence parameter. A first interpretation of results did not show much correlation except for average marks and (of lower importance) student ID.

Evaluation. As top-most indicator the average overall marks by one student has to be seen which sounds quite logical. Why should an overall good student be bad in an (external) project? The most astonishing finding has to be seen in the student ID correlating with average marks. The reasons for this correlation have to be investigated further. As most indicative modules I want to mention COMP1600 and COMP1510. Of further interest should be to examine why some modules do not correlate with the average marks. Unfortunately, no significant indicators could be found in the students' personal backgrounds like age, gender and country of origin. A more detailed view on these might reveal some indicators, though.

References

- Baeza-Yates, R. and Ribeiro, B. d. A. j. N. (1999) *Modern information retrieval*, Addison-Wesley Longman, Harlow.
- Chapman, P., Clinton, J. and others, a. (2000) *CRISP-DM 1.0 : Step-by-step data mining guide*, <<http://www.crisp-dm.org/CRISPWP-0800.pdf>>, Accessed 08/04/2003.
- Chen, M.-S., Yu, P. S. and Liu, B. (2002) *Advances in knowledge discovery and data mining : 6th Pacific-Asia conference, PAKDD 2002, Taipei, Taiwan, May 6-8, 2002 : proceedings*, Springer, Berlin ; London.
- Fayyad, U. M. (1996) *Advances in knowledge discovery and data mining*, AAAI Press; MIT Press, Menlo Park Cambridge, Mass. ; London.
- Witten, I. H. and Eibe, F. (2000) *WEKA : Machine Learning Algorithms in Java*. In *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations* Morgan Kaufmann, pp. 265-320.
- Zighed, D. A., Komorowski, J. and *Zytkow, J. M. (2000) *Principles of data mining and knowledge discovery : 4th European Conference, PKDD 2000, Lyon, France, September 13-16, 2000 : proceedings*, Springer, New York.

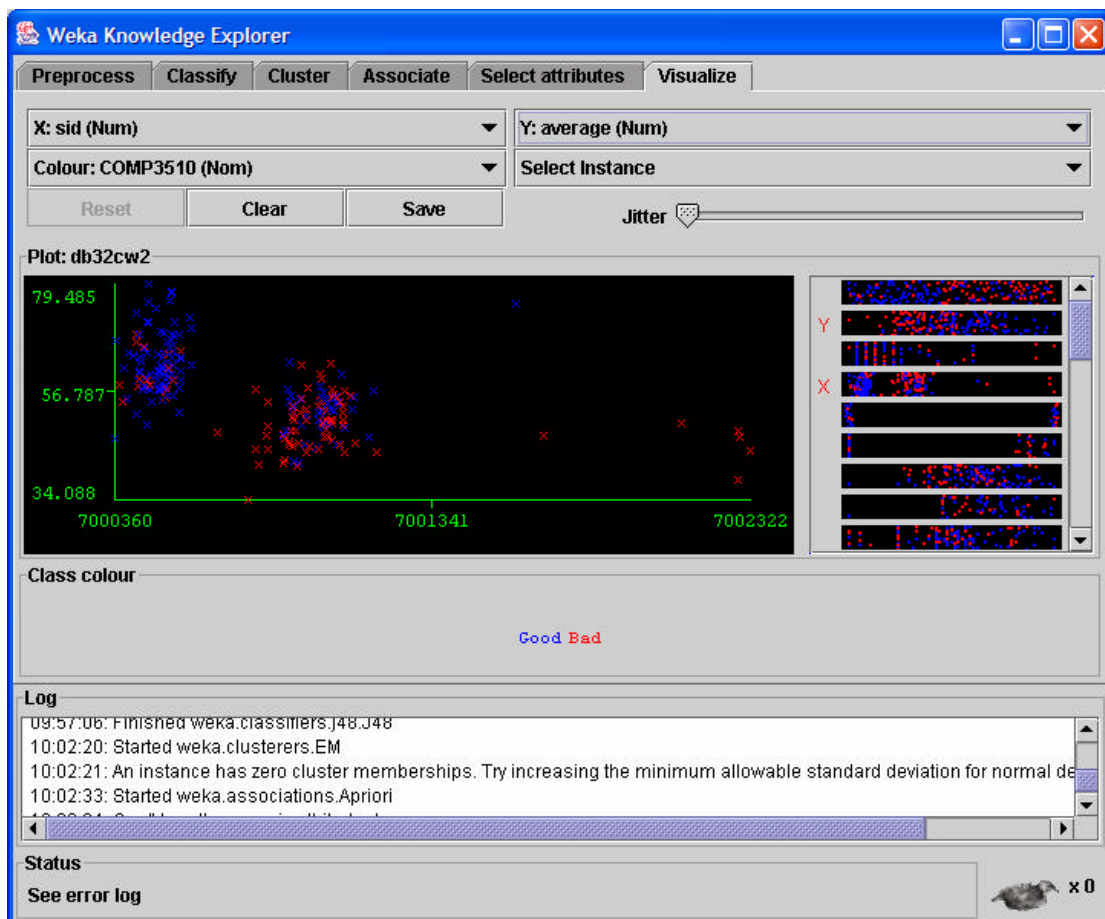


Figure 1⁴

⁴ Figure available in colour as last page

Appendix

Tree using all columns but sid

Scheme: weka.classifiers.j48.J48 -C 0.9 -M 2
Relation: db32cw2-weka.filters.AttributeFilter-V-R1-2,4-32
Instances: 211
Attributes: 31
Test mode: 10-fold cross-validation

J48 pruned tree

```
-----  
average <= 59.333333: Bad (141.0/62.0)  
average > 59.333333  
|   COMP1600 <= 62  
|   |   COMP1510 <= 76  
|   |   |   age <= 21  
|   |   |   |   average <= 60.969697: Good (5.0)  
|   |   |   |   average > 60.969697: Bad (4.38/1.38)  
|   |   |   |   age > 21: Good (3.75)  
|   |   |   COMP1510 > 76: Bad (3.28/0.28)  
|   |   COMP1600 > 62: Good (53.59/3.0)
```

Number of Leaves : 6
Size of the tree : 11

Correctly Classified Instances	127	60.1896 %
Incorrectly Classified Instances	84	39.8104 %
Kappa statistic	0.2143	
Mean absolute error	0.4411	
Root mean squared error	0.5073	
Relative absolute error	90.6935 %	
Root relative squared error	102.8777 %	
Total Number of Instances	211	

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	Class
0.553	0.33	0.701	0.553	0.618	Good
0.67	0.447	0.518	0.67	0.584	Bad

=== Confusion Matrix ===

```
a b <-- classified as  
68 55 | a = Good  
29 59 | b = Bad
```

Tree using average marks only

Scheme: weka.classifiers.j48.J48 -C 0.9 -M 2
Relation: db32cw2-weka.filters.AttributeFilter-V-R1,32
Instances: 211
Attributes: 2
 average
 COMP3510
Test mode: 10-fold cross-validation

J48 pruned tree

```
-----  
average <= 51.428571: Bad (63.0/17.0)  
average > 51.428571: Good (148.0/42.0)
```

Number of Leaves : 2

Size of the tree : 3

Correctly Classified Instances	144	68.2464 %
Incorrectly Classified Instances	67	31.7536 %
Kappa statistic	0.3416	
Mean absolute error	0.4156	
Root mean squared error	0.4655	
Relative absolute error	85.4405 %	
Root relative squared error	94.4158 %	
Total Number of Instances	211	

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	Class
0.748	0.409	0.719	0.748	0.733	Good
0.591	0.252	0.627	0.591	0.608	Bad

=== Confusion Matrix ===

```
a b <-- classified as
92 31 | a = Good
36 52 | b = Bad
```

Rules using average marks only

PART decision list

```
-----
average <= 59.333333 AND
average > 51.428571: Good (78.0/33.0)
```

```
average > 55.151515: Good (70.0/9.0)
: Bad (63.0/17.0)
```

Number of Rules : 3

PART decision list

```
-----
average > 53.69697: Good (82.0/21.0)
: Bad (59.0/21.0)
```

Number of Rules : 2

Tree using SID only

Scheme: weka.classifiers.j48.J48 -U -M 2
Instances: 211
Attributes: 2
Test mode: 10-fold cross-validation

J48 unpruned tree

```
-----
sid <= 7000598: Good (97.0/15.0)
sid > 7000598: Bad (114.0/41.0)
```

Number of Leaves : 2

Size of the tree : 3

Correctly Classified Instances	154	72.9858 %
Incorrectly Classified Instances	57	27.0142 %
Kappa statistic	0.4677	
Mean absolute error	0.3726	
Root mean squared error	0.4337	
Relative absolute error	76.6032 %	
Root relative squared error	87.9596 %	

Total Number of Instances 211

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	Class
0.659	0.17	0.844	0.659	0.74	Good
0.83	0.341	0.635	0.83	0.719	Bad

=== Confusion Matrix ===

```
a b <-- classified as
81 42 | a = Good
15 73 | b = Bad
```

Rules using COMP modules only

Scheme: weka.classifiers.j48.PART -C 0.9 -M 2
Instances: 211
Attributes: 27
Test mode: 10-fold cross-validation

PART decision list

```
-----
COMP1600 > 64 AND
COMP1400 > 65: Good (48.58/6.91)

COMP2380 > 52 AND
COMP2470 > 50: Good (32.48/15.42)

COMP2370 > 47 AND
COMP2470 > 43: Good (51.75/24.7)

COMP2660 <= 66: Bad (54.08/24.4)
: Good (24.1/11.29)
```

Number of Rules : 5

Correctly Classified Instances	120	56.872 %
Incorrectly Classified Instances	91	43.128 %
Kappa statistic	0.134	
Mean absolute error	0.449	
Root mean squared error	0.474	
Relative absolute error	92.3085 %	
Root relative squared error	96.134 %	
Total Number of Instances	211	

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	Class
0.569	0.432	0.648	0.569	0.606	Good
0.568	0.431	0.485	0.568	0.524	Bad

=== Confusion Matrix ===

```
a b <-- classified as
70 53 | a = Good
38 50 | b = Bad
```