



2130 – Data Management and Business Intelligence

- *Name:* Agata
- *Surname:* GABARA

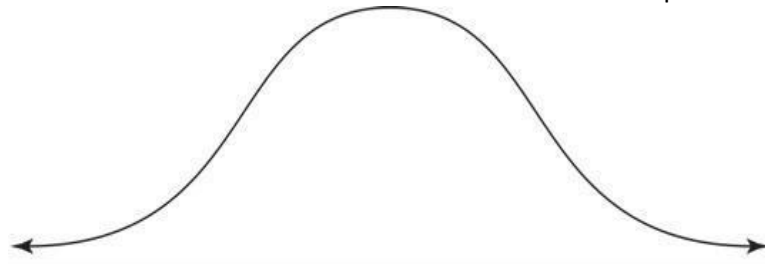
Task 1.2 Familiarise with the dataset and comment the frequency distribution of your data.

All data are numeric.

For calories:

Minimum and maximum are low. Mean is high.

It shows normal distribution because there is a bell shape.



Name: calories	Distinct: 10	Type: Numeric
Missing: 0 (0%)		Unique: 4 (7%)
Statistic	Value	
Minimum	50	
Maximum	160	
Mean	106.842	
StdDev	18.913	

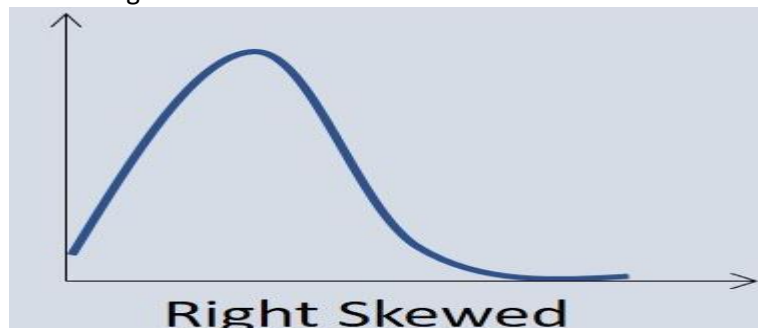
Calories are numeric value; you can see min, max, means, and standard deviation in 'Selected Attribute' window. Missing = 0 means that the attribute is specified for all instances (no missing values) Distinct = 10 means that calories have ten different values, and Unique = 4 means that other attributes or instances have the same 10 value as calories have.

Minimum 50 means it is the lowest value, maximum 160 means it is the highest value, mean = 106.842 means it is average value and St. dev (18.913) shows how spread out numbers are.

For protein:

Minimum is high and maximum is low. Mean is medium.

This is a right skewed.



For Sodium:

Minimum is medium, maximum is low and Mean is high.

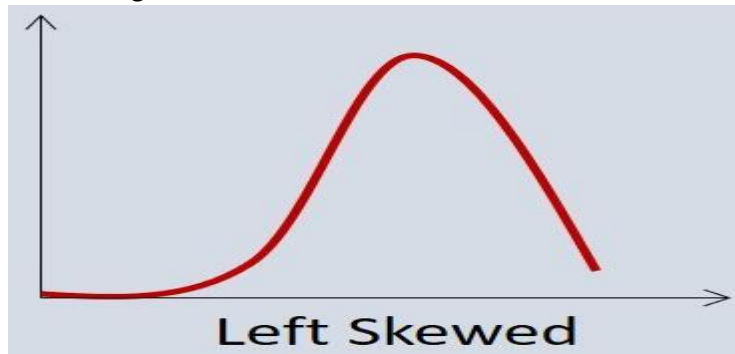
There is no distribution.

For Fibre:

Minimum is high. Maximum and mean are both low.
There is lack of data (lack of continuation)

For Carbo:

Minimum is low and maximum is medium.
Mean is high. This is a left skewed.



For Sugars:

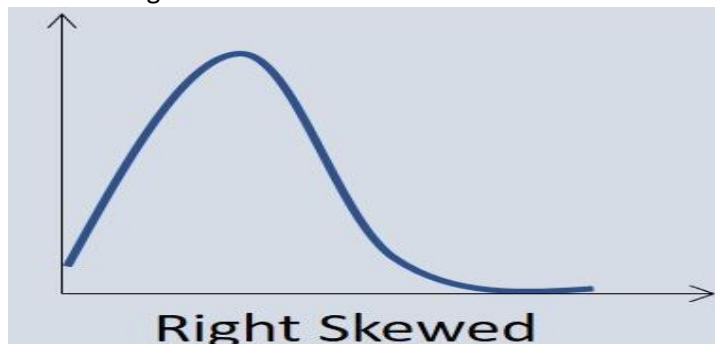
Minimum is high. Maximum is average and mean is medium.
There is no distribution.

For Vitamins:

Minimum and maximum are average. Below mean is high.
There is lack of continuity of data, no distribution.

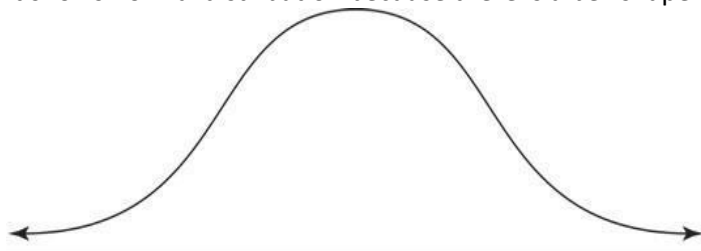
For Rating:

Minimum is average and maximum is low. Before Mean is high.
There is a right skewed distribution.



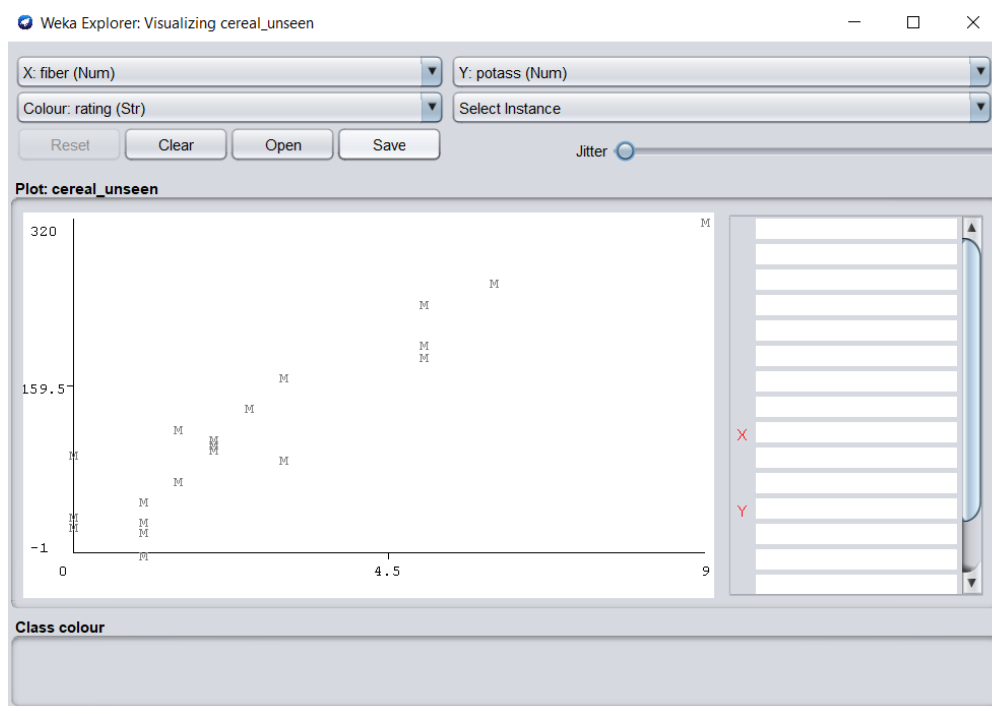
For cups:

Minimum and maximum are low. Mean is high.
It shows normal distribution because there is a bell shape.



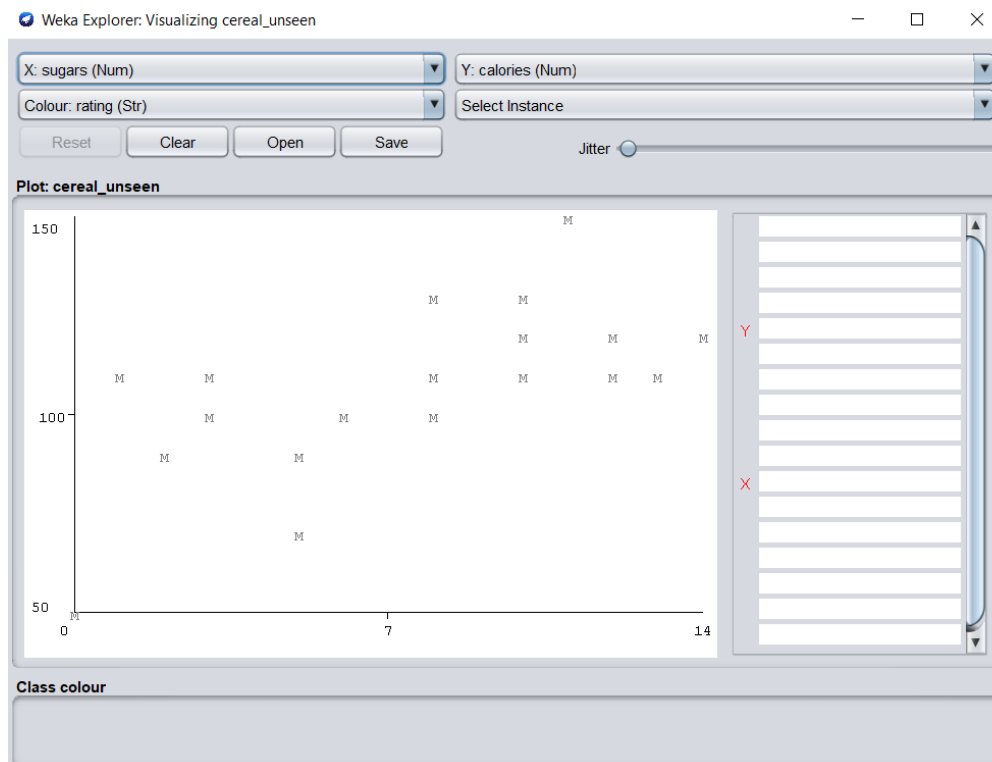
Task 1.4 Visualise the data and check for visual patterns (e.g., positive/negative, linear/nonlinear, strong/weak patterns)

Visualisation for fibre with potassium



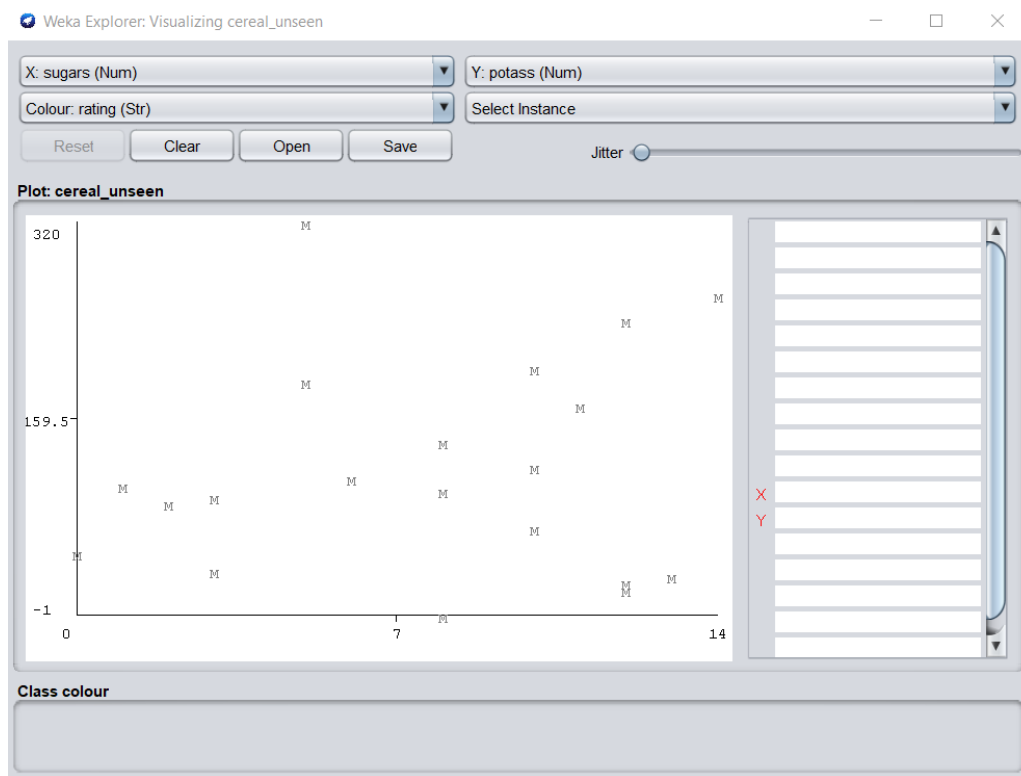
Direction: Positive
Form: Non-Linear
Strength: Weak
Outliers: None

Visualisation for sugars with calories



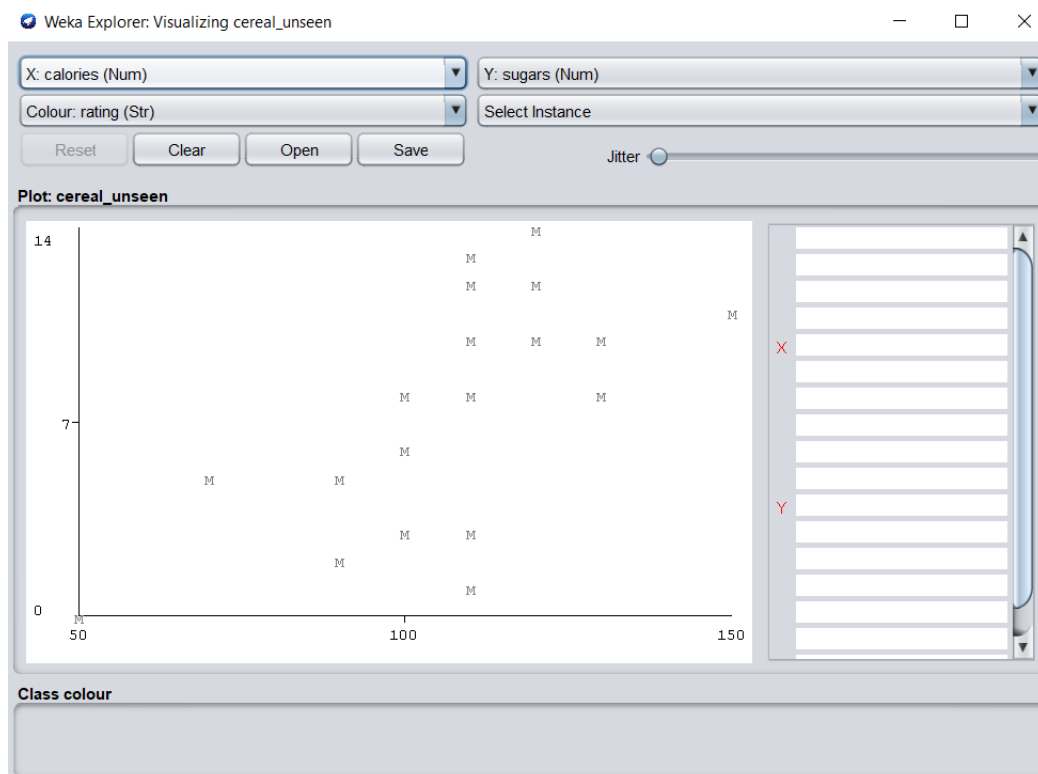
Direction: Positive
Form: Non-Linear
Strength: Weak
Outliers: 1

Visualisation for sugars with potassium



Direction: Positive
Form: Non-Linear
Strength: Weak
Outliers: 1

Visualisation for calories with sugars



Direction: Positive
Form: Non-Linear
Strength: Weak
Outliers: 1

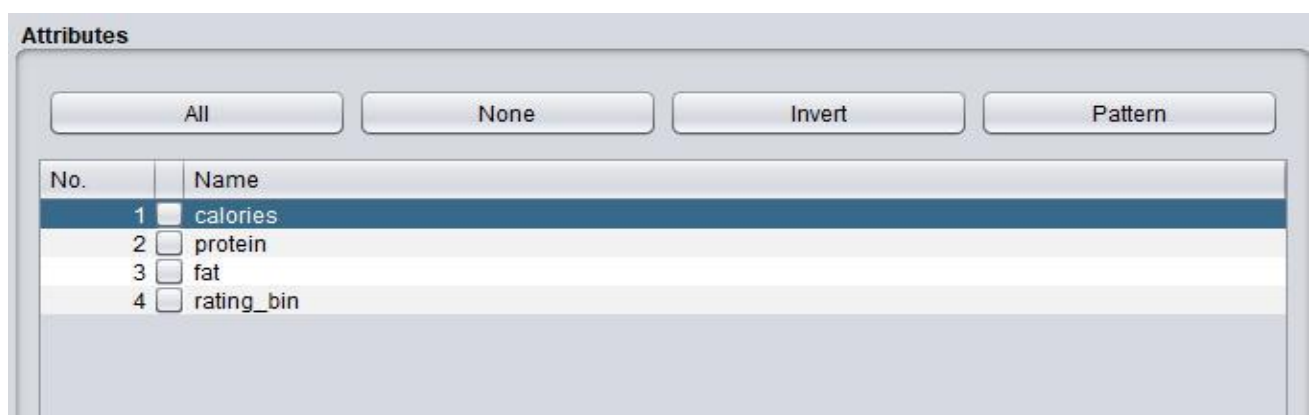
Task 2 Finding hidden patterns. Use any unsupervised machine learning technique (e.g., clustering, association rules) to find hidden patterns on your data. While doing so, focus on unexpected and interesting patterns. Motivate your analysis with clear and explanatory text.

Finding associations:

Data are nominal.

Apriori algorithm.

Firstly, it is needed to change data from numeric to nominal, and String to nominal. Get rid of all attributes which have missing values, not normal distribution and not left or right skew. We keep 4 attributes (3 not changed from database and 1 called rating_bin which have nominal data (high, low, medium) which data was divided in R) and also we discretise the data



```
Attributes:  4
             calories
             protein
             fat
             rating_bin
=== Associator model (full training set) ===
```

```
Apriori
=====
```

```
Minimum support: 0.1 (6 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 18
```

Minimum support: 0.1 (6 instances) Minimum metric <confidence>: 0.9 Number of cycles performed: 18

The program generated the sets of large item sets found for each support size considered. By default, Apriori tries to generate ten rules. The minimum confidence is set 0.9 (90%). As we can see, the

minimum support decreased to 0.1 (10%), before the required number of rules can be generated. Generation of the required number of rules involved a total of 18 iterations.

Associate rules:

We have found 8 rules.

```
Minimum support: 0.1 (6 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 18

Generated sets of large itemsets:

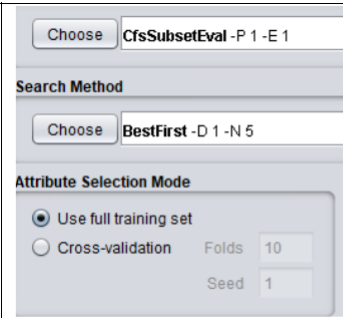
Size of set of large itemsets L(1): 12
Size of set of large itemsets L(2): 21
Size of set of large itemsets L(3): 7
Size of set of large itemsets L(4): 1

Best rules found:

1. calories=110 fat=1 11 ==> rating_bin=low 11    <conf:(1)> lift:(1.36) lev:(0.05) [2] conv:(2.89)
2. calories=110 protein=2 10 ==> rating_bin=low 10  <conf:(1)> lift:(1.36) lev:(0.05) [2] conv:(2.63)
3. protein=2 fat=1 10 ==> rating_bin=low 10    <conf:(1)> lift:(1.36) lev:(0.05) [2] conv:(2.63)
4. calories=120 7 ==> rating_bin=low 7    <conf:(1)> lift:(1.36) lev:(0.03) [1] conv:(1.84)
5. calories=110 protein=1 7 ==> rating_bin=low 7    <conf:(1)> lift:(1.36) lev:(0.03) [1] conv:(1.84)
6. calories=110 protein=2 fat=1 6 ==> rating_bin=low 6    <conf:(1)> lift:(1.36) lev:(0.03) [1] conv:(1.58)
7. calories=110 22 ==> rating_bin=low 20    <conf:(0.91)> lift:(1.23) lev:(0.07) [3] conv:(1.93)
8. protein=1 11 ==> rating_bin=low 10    <conf:(0.91)> lift:(1.23) lev:(0.03) [1] conv:(1.45)
```

The number preceding ==> symbol indicates the rule's support, that is, the number of items covered by its premise. Following the rule is the number of those items for which the rule's consequent holds as well. In the parentheses there is a confidence of the rule.

Select attributes:¹



Choose CfsSubsetEval -P 1 -E 1

Search Method

Choose BestFirst -D 1 -N 5

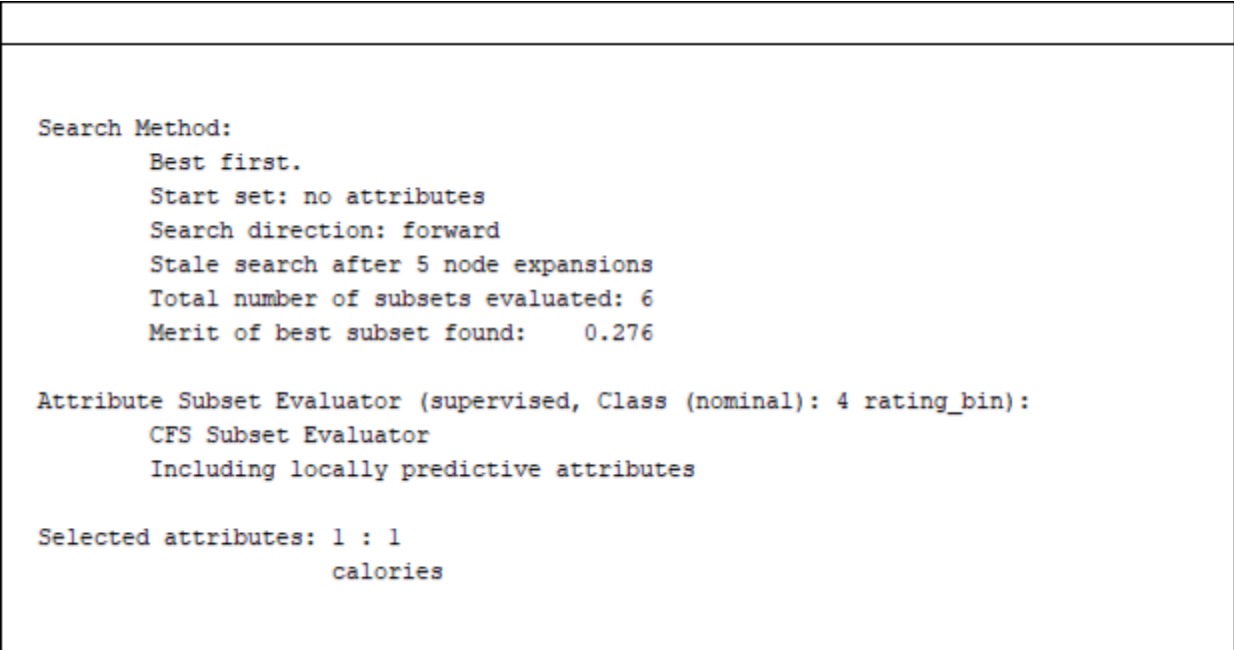
Attribute Selection Mode

☒ Use full training set

☐ Cross-validation Folds 10

Seed 1

Full training set – analyses



```
Search Method:
  Best first.
  Start set: no attributes
  Search direction: forward
  Stale search after 5 node expansions
  Total number of subsets evaluated: 6
  Merit of best subset found:    0.276

Attribute Subset Evaluator (supervised, Class (nominal): 4 rating_bin):
  CFS Subset Evaluator
  Including locally predictive attributes

Selected attributes: 1 : 1
                    calories
```

The search method selected is the Best Fit. The software started search with no attributes, and it is forward search. We evaluated 6 subsets and the merit of the best subset is 0.276. The attribute evaluator used is CFS Subset Evaluator. We used supervised learning with labels in the attribute rating_bin. The selected attribute for prediction is 'calories'.

¹(To search through all possible combinations of attributes in the data and find which subset of attributes works best for prediction, make sure that you set up attribute evaluator to 'CfsSubsetEval' and a search method to 'BestFirst'. The evaluator will determine what method to use to assign a worth to each subset of attributes. The search method will determine what style of search to perform. The options that you can set for selection in the 'Attribute Selection Mode' box are [2]: 1. Use full training set. The worth of the attribute subset is determined using the full set of training data. 2. Cross-validation. The worth of the attribute subset is determined by a process of cross-validation. The 'Fold' and 'Seed' fields set the number of folds to use and the random seed used when shuffling the data)

Clustering:²

Simple k-Means.

57 observations, 4 attributes.

```
kMeans
=====

Number of iterations: 3
Within cluster sum of squared errors: 91.0

Initial starting points (random):

Cluster 0: 100,3,0,medium
Cluster 1: 80,2,0,high

Missing values globally replaced with mean/mode

Final cluster centroids:

Attribute      Full Data      Cluster#
                (57.0)      (37.0)      (20.0)
=====
calories        110          100          110
protein          2           3           2
fat              0           0           1
rating_bin      low          low          low

Time taken to build model (full training data) : 0.02 seconds

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

Clustered Instances

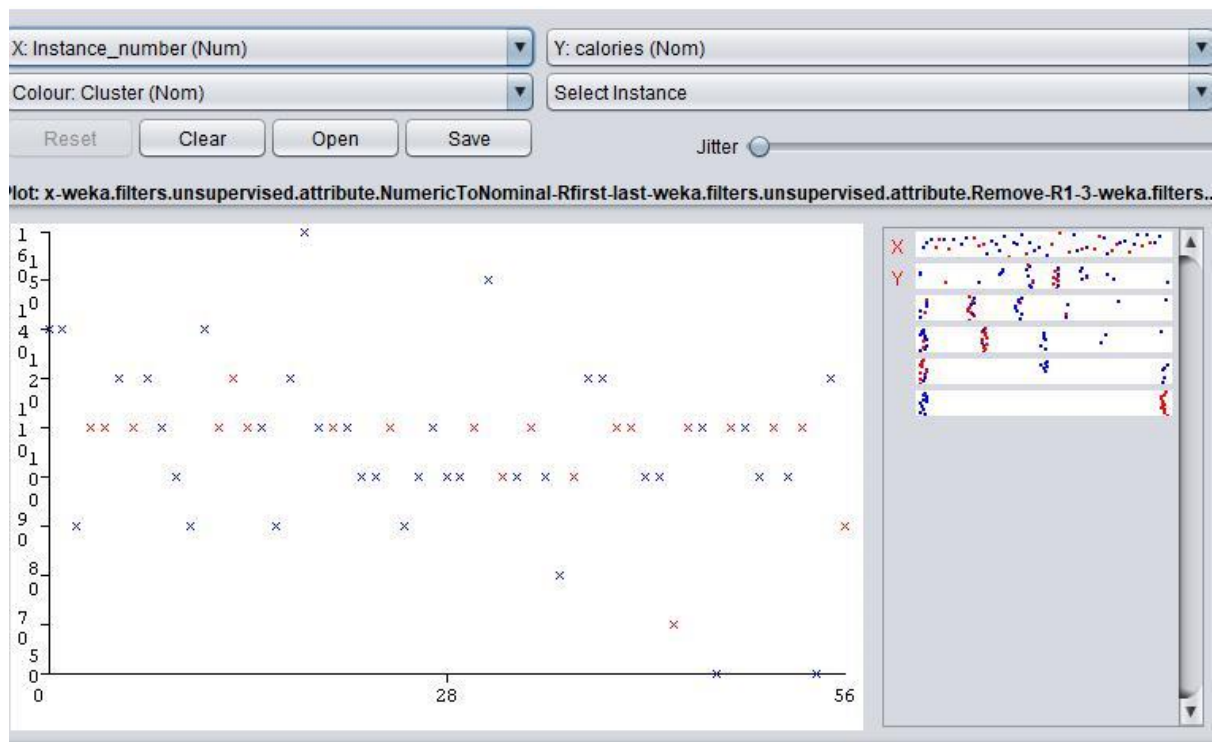
0      37 ( 65%)
1      20 ( 35%)
```

There are 2 clusters, WEKA replaced the missing or incorrect values with mode, incorrectly clustered instances 37.0 which constitutes circa 65%. It is not good for estimation when this replacement percentage is so high.

The clustering model shows the centroid of each cluster and statistics on the number and percentage of instances assigned to different clusters. Cluster centroids are the mean vectors for each cluster; so, each dimension value and the centroid represents the mean value for that dimension in the cluster. Thus, centroids can be used to characterize the clusters.

²The clustering schemes available in WEKA are k-Means, EM, Cobweb, X-means, Farthest First. Evaluation is based on log likelihood if clustering scheme produces a probability distribution.

Visualising cluster assignments:



Task 3 Build a set of supervised models to predict the field rating. You are welcome to build many prediction models by using different classifiers (e.g., Logistic regression, J48, Random Forest) and adopting different variations of them (this is done by changing the model parameters in Weka). While doing so, motivate your choices with clear and explanatory text.

Logistic regression:

We created database. We have used “resample” from WEKA and we have applied.

We did 2 bins: low and high. All what is below the average is low and above is high.

Data have been changed from String to Nominal. (rating_bin)

Selected attribute				
Name: rating_bin		Distinct: 2		Type: Nominal
Missing: 0 (0%)				Unique: 0 (0%)
No.	Label	Count	Weight	
1	low	22	22.0	
2	high	17	17.0	

Visualisation:



Seed: 153879

70%: 30%

Logistic regression

Time taken to test model on training data: 0 seconds

=== Summary ===

Correctly Classified Instances	39	100	%
Incorrectly Classified Instances	0	0	%
Kappa statistic	1		
Mean absolute error	0		
Root mean squared error	0		
Relative absolute error	0.0002	%	
Root relative squared error	0.0005	%	
Total Number of Instances	39		

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	low
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	high
Weighted Avg.	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	

=== Confusion Matrix ===

```
a b <-- classified as
22 0 | a = low
0 17 | b = high
```

The result is the same like J.48

Interpretation – the same like J.48.

J.48³

```
=== Summary ===

Correctly Classified Instances      39          100    %
Incorrectly Classified Instances    0           0    %
Kappa statistic                     1
Mean absolute error                 0
Root mean squared error             0
Relative absolute error              0    %
Root relative squared error          0    %
Total Number of Instances          39

=== Detailed Accuracy By Class ===

                TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                1.000    0.000    1.000     1.000    1.000     1.000    1.000    1.000    low
                1.000    0.000    1.000     1.000    1.000     1.000    1.000    1.000    high
Weighted Avg.   1.000    0.000    1.000     1.000    1.000     1.000    1.000    1.000

=== Confusion Matrix ===

  a  b  <-- classified as
22  0  |  a = low
 0 17  |  b = high
```

Total: 39 instances

100 % correctly adjusted. 0 % incorrectly adjusted

Mean absolute error explains 0% variance of independent variable. (not need to interpretate as it is perfect adjust, 100% correctly classified instances) Others errors also not needed to interpret.

Confusion matrix:

We have 2 parameters: a and b. 22 have been identified as low (a), 17 have been identified as high (b) Both correctly.

Simple Logistic:⁴

Attributes:

³ J48 is an algorithm used to generate a decision tree developed by Ross Quinlan

⁴ Simple Logistic is a symmetric model (see <http://list.waikato.ac.nz/pipermail/wekalist/2013-October/059347.html>) whereas Logistic is not. That means for logistic, if A_j is set of coefficients for class j , the probability instance x_i is in class j is $\exp(A_j x_i) / (\sum_{k=1}^{k-1} \exp(A_k x_i) + 1)$ for classes $1, \dots, \{k-1\}$ and $1 - (\sum_{k=1}^{k-1} \exp(A_k x_i) + 1)$ for the last class.

No.	Name
1	<input type="checkbox"/> calories
2	<input type="checkbox"/> protein
3	<input type="checkbox"/> fat
4	<input type="checkbox"/> sodium
5	<input type="checkbox"/> fiber
6	<input type="checkbox"/> carbo
7	<input type="checkbox"/> sugars
8	<input type="checkbox"/> potass
9	<input type="checkbox"/> vitamins
10	<input type="checkbox"/> shelf
11	<input type="checkbox"/> weight
12	<input type="checkbox"/> cups
13	<input type="checkbox"/> rating_bin

All data must be numeric.

Classify – simple logistics.

Training set.

```

Correctly Classified Instances      38          97.4359 %
Incorrectly Classified Instances    1           2.5641 %
Kappa statistic                    0.9482
Mean absolute error                 0.1606
Root mean squared error             0.2169
Relative absolute error             32.6319 %
Root relative squared error         43.7371 %
Total Number of Instances          39

```

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.955	0.000	1.000	0.955	0.977	0.949	1.000	1.000	low
	1.000	0.045	0.944	1.000	0.971	0.949	1.000	1.000	high
Weighted Avg.	0.974	0.020	0.976	0.974	0.974	0.949	1.000	1.000	

=== Confusion Matrix ===

```

a b  <-- classified as
21 1 | a = low
0 17 | b = high

```

97.4359 % of data have been correctly classified.

2.564 1% of data have been incorrectly classified.

Mean absolute error explains 16% variance of independent variable.

The number of instances 39.

Confusion matrix:

Categories: a, b had been predicted.

18 observations had been done incorrectly (1+17)

21 observations had been predicted correctly.

Task 4:

Use k-fold cross validation to test the performance of your models. Comment the confusion matrices and the accuracy measures you get. Among your models, choose the one showing the best accuracy.

k=3, We had 3 types: low, medium, high.

We removed rating and rating-bin changes from String to Nominal.

We removed String attributes.

No.		Name
1	<input type="checkbox"/>	calories
2	<input type="checkbox"/>	protein
3	<input type="checkbox"/>	fat
4	<input type="checkbox"/>	sodium
5	<input type="checkbox"/>	fiber
6	<input type="checkbox"/>	carbo
7	<input type="checkbox"/>	sugars
8	<input type="checkbox"/>	potass
9	<input type="checkbox"/>	vitamins
10	<input type="checkbox"/>	shelf
11	<input type="checkbox"/>	weight
12	<input type="checkbox"/>	cups
13	<input type="checkbox"/>	rating_bin

3-fold cross validation: (It is based on 3 averages)

- Divide dataset into 3 parts,
- Hold out each part in turn
- Average the results
- Each data point used once for testing, 2 times for training.

Number of iterations: 4
 Within cluster sum of squared errors: 43.0437629045781

Initial starting points (random):

Cluster 0: 100,3,0,0,3,14,7,100,25,2,1,0.8,medium
 Cluster 1: 80,2,0,0,3,16,0,95,0,1,0.83,1,high

Missing values globally replaced with mean/mode

Final cluster centroids:

jAttribute	Cluster#		
	Full Data	0	1
	(57.0)	(36.0)	(21.0)
calories	106.8421	110	101.4286
protein	2.4211	2.3889	2.4762
fat	0.9298	1.2222	0.4286
sodium	158.8596	156.8056	162.381
fiber	2.0035	2.2222	1.6286
carbo	14.9386	14.0556	16.4524
sugars	6.6667	7.9722	4.4286
potass	88.2281	99.0278	69.7143
vitamins	28.5088	34.0278	19.0476
shelf	2.0877	2.6389	1.1429
weight	1.0226	1.0406	0.9919
cups	0.8268	0.7692	0.9257
cups	0.8268	0.7692	0.9257
rating_bin	low	low	low

Time taken to build model (full training data) : 0.01 seconds

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

Clustered Instances

0 36 (63%)
 1 21 (37%)

Confusion matrix – interpretation.

We have 2 levels only significant in this model it is a medium and high. Low level is abnormal in this modelling. We have 57 instances. 63% have been identified as high and 37% as fitting model.

Task 5

Use your best model to make predictions on unseen data. Generate your predictions and save them to file.

1R

“The purpose of the boundary visualizer is to show the predictions of a given model for every possible combination of attribute values—that is, for every point in the two-dimensional space”⁵

```
Time taken to test model on training data: 0 seconds

=== Summary ===

Correctly Classified Instances      35           89.7436 %
Incorrectly Classified Instances     4           10.2564 %
Kappa statistic                     0.7886
Mean absolute error                  0.1026
Root mean squared error              0.3203
Relative absolute error              20.8386 %
Root relative squared error          64.5829 %
Total Number of Instances           39

=== Detailed Accuracy By Class ===
```

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.955	0.176	0.875	0.955	0.913	0.793	0.889	0.861	low
	0.824	0.045	0.933	0.824	0.875	0.793	0.889	0.846	high
Weighted Avg.	0.897	0.119	0.900	0.897	0.896	0.793	0.889	0.854	

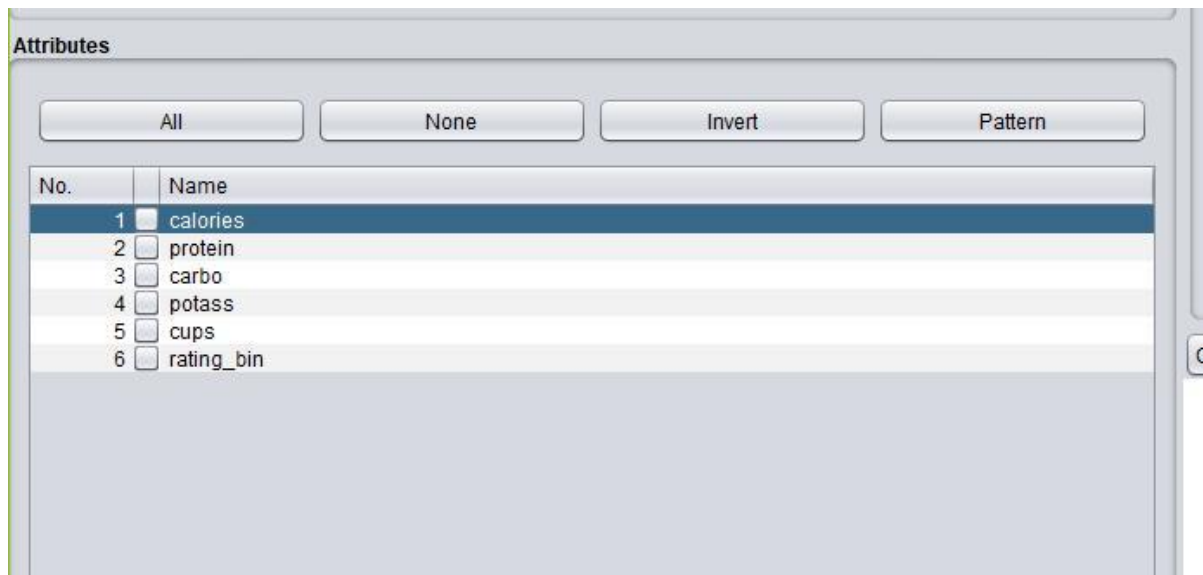
```
=== Confusion Matrix ===

 a b  <-- classified as
21  1 | a = low
 3 14 | b = high
```

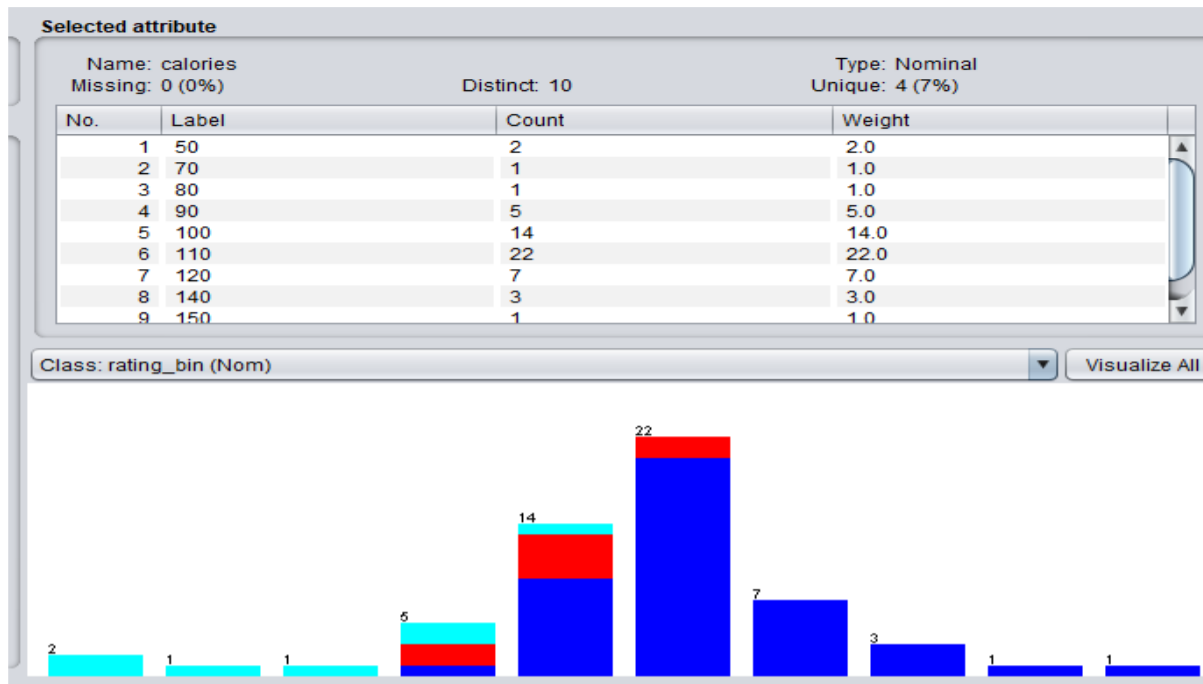
In this case it is close 90% correctly assess instances. We have 2 classes: low and high.

Mean absolute error explains 10% variance of independent variable.

NAÏVE BAYES



⁵ Weka tutorial, page: 15



=== Summary ===

Correctly Classified Instances	52	91.2281 %
Incorrectly Classified Instances	5	8.7719 %
Kappa statistic	0.7752	
Mean absolute error	0.1084	
Root mean squared error	0.1962	
Relative absolute error	37.4152 %	
Root relative squared error	52.2612 %	
Total Number of Instances	57	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.976	0.267	0.911	0.976	0.943	0.766	0.981	0.994	low
	0.750	0.020	0.857	0.750	0.800	0.772	0.995	0.970	medium
	0.714	0.000	1.000	0.714	0.833	0.829	0.994	0.962	high
Weighted Avg.	0.912	0.199	0.914	0.912	0.909	0.775	0.985	0.986	

=== Confusion Matrix ===

```

a b c <-- classified as
41 1 0 | a = low
 2 6 0 | b = medium
 2 0 5 | c = high

```

Close 92% correctly classified instances. Mean absolute error explains 11 % variance of independent variable.

NaïveBayes - theory:

Its assumption that attributes are conditionally independent given a particular class value means that the overall class probability is obtained by simply multiplying the per-attribute conditional probabilities together.

By default, Weka's NaiveBayes classifier assumes that the attributes are normally distributed given the class. This will cause NaïveBayes to discretize the numeric attributes in the data with a supervised discretization technique. In most practical applications of NaïveBayes, supervised discretization works better than the default method.