

17.1.1

The values that the attribute *temperature* have are nominal values. It includes “hot”, “mild” and “cool”, as is shown below.

Name: temperature Missing: 0 (0%)		Distinct: 3	Type: Nominal Unique: 0 (0%)
No.	Label	Count	Weight
1	hot	4	4.0
2	mild	6	6.0
3	cool	4	4.0

17.1.2

Load a new dataset. Click the *Open file* button and select the file *iris.arff*, which corresponds to the iris dataset in Table 1.4. How many instances does this dataset have? How many attributes? What is the range of possible values of the attribute *petallength*?

Current relation	
Relation: iris Instances: 150	Attributes: 5 Sum of weights: 150

As is shown above, the dataset has 150 instances and 5 attributes. The first four attributes *sepalength*, *sepalwidth*, *petallength*, and *petalwidth* are properties (real attributes), and the last attribute *class* is a class label attribute.

Selected attribute	
Name: petallength Missing: 0 (0%)	
Distinct: 43	
Type: Numeric Unique: 10 (7%)	
Statistic	Value
Minimum	1
Maximum	6.9
Mean	3.759
StdDev	1.764

The range of values of the attribute *petallength* is $\text{Range}(\text{petallength}) = (6.9 - 1) = 5.9$

17.1.3

What is the function of the first column in the Viewer window?

The first column is called No. It can show how many instances there are in the dataset. It also assigns each instance a unique ID number to be identified.

17.1.4

What is the class value of instance number 8 in the weather data?

The class label is the *play* column. The class value of instance No.8 is no.



Viewer

Relation: weather.symbolic					
No.	1: outlook	2: temperature	3: humidity	4: windy	5: play
	Nominal	Nominal	Nominal	Nominal	Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

17.1.5

Load the iris data and open it in the editor. How many numeric and how many nominal attributes does this dataset have?



Viewer

Relation: iris					
No.	1: sepalength	2: sepalwidth	3: petallength	4: petalwidth	5: class
	Numeric	Numeric	Numeric	Numeric	Nominal
1	5.1	3.5	1.4	0.2	Iris-setosa
2	4.9	3.0	1.4	0.2	Iris-setosa
3	4.7	3.2	1.3	0.2	Iris-setosa

The iris dataset has 4 numeric attributes and 1 nominal attribute. The one nominal attribute is the class label attribute.

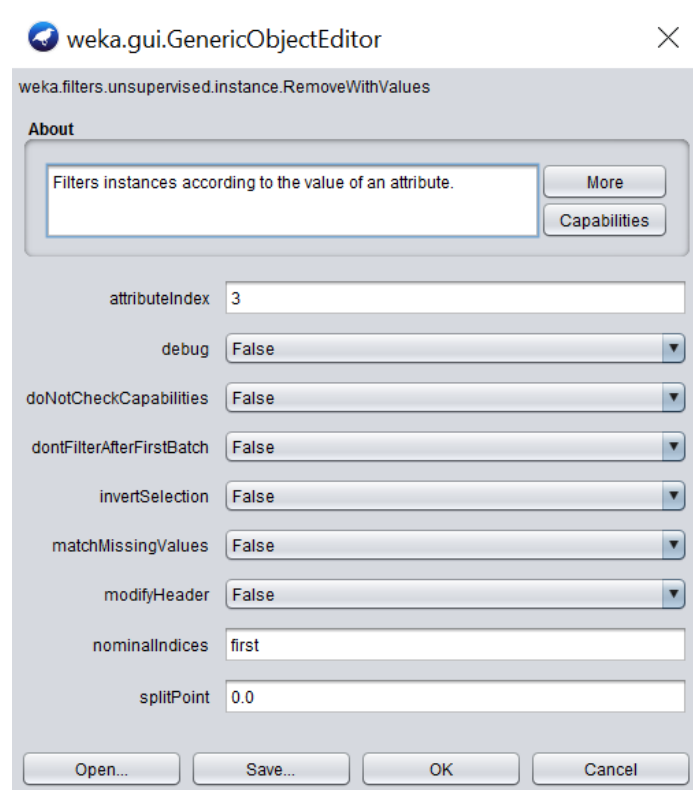
17.1.6

Load the *weather.nominal* dataset. Use the filter *weka.unsupervised.instance.RemoveWithValues* to remove all instances in which the *humidity* attribute has the value *high*.

When following the path and choosing *RemoveWithValues*, I click on it again and set values to configure the function.

Selected attribute	
Name: humidity	
Missing: 0 (0%)	
No.	Label
1	high
2	normal

As can be seen above, the *high* label is the first label of attribute *humidity*, so I need to remove the instances whose *humidity* attribute has first label values. As is shown below, I set attributeIndex to 3 since the *humidity* attribute is the third one in the dataset. Then I enter “first” in the nominalIndices row in order to remove instances in which the *humidity* attribute has the value *high*.



Then, it shows on the right of the choose button.



When I click “Apply”, the filter works. Consequently, 7 instances have been removed.

Selected attribute			
Name: humidity Missing: 0 (0%)		Distinct: 1	Type: Nominal Unique: 0 (0%)
No.	Label	Count	Weight
1	high	0	0.0
2	normal	7	7.0

17.1.7

Undo the change to the dataset that you just performed, and verify that the data has reverted to its original state.

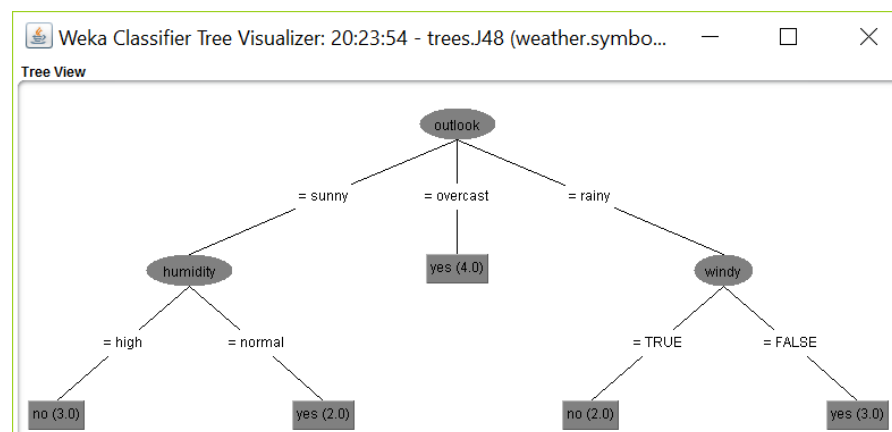
Just click “Undo” button, and then it has reverted to the original state. I verify it by taking a look at the *humidity* attribute again.

Selected attribute			
Name: humidity Missing: 0 (0%)		Distinct: 2	Type: Nominal Unique: 0 (0%)
No.	Label	Count	Weight
1	high	7	7.0
2	normal	7	7.0

17.1.8

How would this instance be classified using the decision tree?

outlook = sunny, temperature = cool, humidity = high, windy = TRUE



Using the decision tree above, the first attribute that should be considered is outlook. Since outlook = sunny, following the left branch, the humidity is the second attribute that needs to be considered. Since humidity = high, the instance should be classified as *no* class. That is to say, the instance would be classified as not to play.

17.1.9

Load the iris data using the Preprocess panel. Evaluate C4.5 on this data using (a) the training set and (b) cross-validation. What is the estimated percentage of correct classifications for (a) and (b)? Which estimate is more realistic?

(a) Training Dataset: Only to be used when you have all of the data and you are interested in creating a descriptive rather than a predictive model. You are interested in creating a model to better understand the problem (Brownlee, 2016).

For using the training set method, this evaluation is highly optimistic. It may still be useful because it generally represents an upper bound to the model's performance on fresh data. The ideal percentage of correct classification for using the training set is 100%.

(b) Cross Validation: The default. To be used when you are unsure. Generally, it provides a more accurate estimate of the performance than the other techniques. Not to be used when you have a very large data. Common values for k are 5 and 10, depending on the size of the dataset.

Brownlee, J. (2016, July 18). How To Estimate The Performance of Machine Learning Algorithms in Weka. Retrieved from <http://machinelearningmastery.com/estimate-performance-machine-learning-algorithms-weka/>

(a) Test mode: evaluate on training data

```

=== Run information ===

Scheme:      weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:    iris
Instances:   150
Attributes:  5
              sepallength
              sepalwidth
              petallength
              petalwidth
              class
Test mode:   evaluate on training data

```

Correctly Classified Instances 147 **98 %**

Incorrectly Classified Instances 3 2 %

(b) Test mode: 10-fold cross-validation

(may be better when the amount of data for training and testing is limited)

```

=== Run information ===

Scheme:      weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:    iris
Instances:   150
Attributes:  5
              sepallength
              sepalwidth
              petallength
              petalwidth
              class
Test mode:   10-fold cross-validation

```

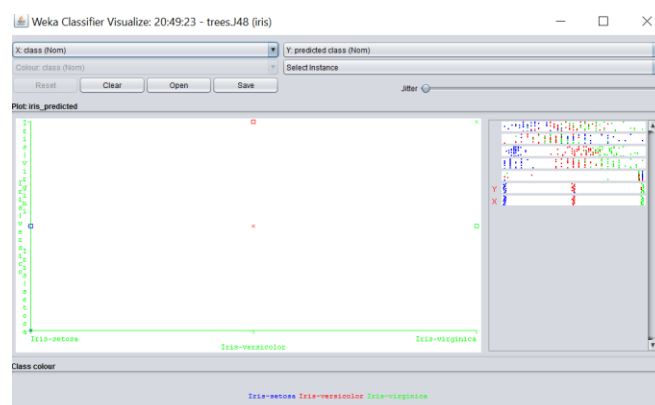
Correctly Classified Instances 144 **96 %**

Incorrectly Classified Instances 6 4 %

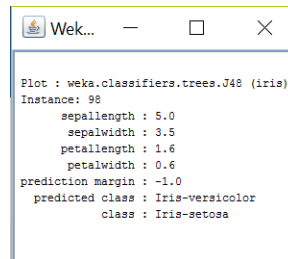
The second estimate (96%) which adopting the 10-fold cross-validation is more realistic. Reasons are illustrated above.

17.1.10

Use the *Visualize classifier errors* function to find the wrongly classified test instances for the cross-validation performed in Exercise 17.1.9. What can you say about the location of the errors?



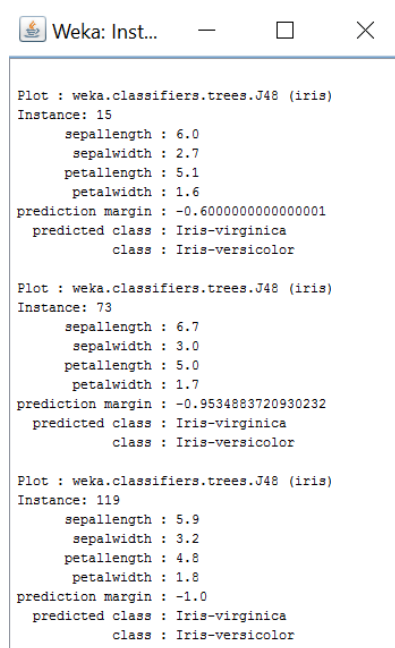
There are three locations of the errors. The first blue square box contains one wrongly classified instance, which is shown below. It belongs to class setosa but is classified as class versicolor.



```

Plot : weka.classifiers.trees.J48 (iris)
Instance: 98
    sepalength : 5.0
    sepalwidth : 3.5
    petallength : 1.6
    petalwidth : 0.6
prediction margin : -1.0
predicted class : Iris-versicolor
class : Iris-setosa
  
```

The red square box contains three wrongly classified instances, which are shown below. All of them are from class versicolor but are classified as class virginica.



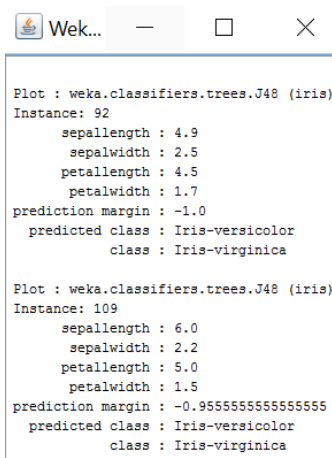
```

Plot : weka.classifiers.trees.J48 (iris)
Instance: 15
    sepalength : 6.0
    sepalwidth : 2.7
    petallength : 5.1
    petalwidth : 1.6
prediction margin : -0.6000000000000001
predicted class : Iris-virginica
class : Iris-versicolor

Plot : weka.classifiers.trees.J48 (iris)
Instance: 73
    sepalength : 6.7
    sepalwidth : 3.0
    petallength : 5.0
    petalwidth : 1.7
prediction margin : -0.9534883720930232
predicted class : Iris-virginica
class : Iris-versicolor

Plot : weka.classifiers.trees.J48 (iris)
Instance: 119
    sepalength : 5.9
    sepalwidth : 3.2
    petallength : 4.8
    petalwidth : 1.8
prediction margin : -1.0
predicted class : Iris-virginica
class : Iris-versicolor
  
```

The green square box contains two wrongly classified instances, which are shown below. Both of them are from class virginica but are classified as class versicolor.



```

Plot : weka.classifiers.trees.J48 (iris)
Instance: 92
    sepalength : 4.9
    sepalwidth : 2.5
    petallength : 4.5
    petalwidth : 1.7
prediction margin : -1.0
predicted class : Iris-versicolor
class : Iris-virginica

Plot : weka.classifiers.trees.J48 (iris)
Instance: 109
    sepalength : 6.0
    sepalwidth : 2.2
    petallength : 5.0
    petalwidth : 1.5
prediction margin : -0.9555555555555555
predicted class : Iris-versicolor
class : Iris-virginica
  
```

X-axis represents real class; y-axis represents predicted class. The errors are located at the intersections of two different classes. All of the three square boxes are one unit far from the corrected classified crosses.