

by : Agung Mulyo Widodo, ST. M.Sc.
Ir. Nizirwan Anwar, MT



Penggunaan WEKA sebagai alat bantu Data Mining



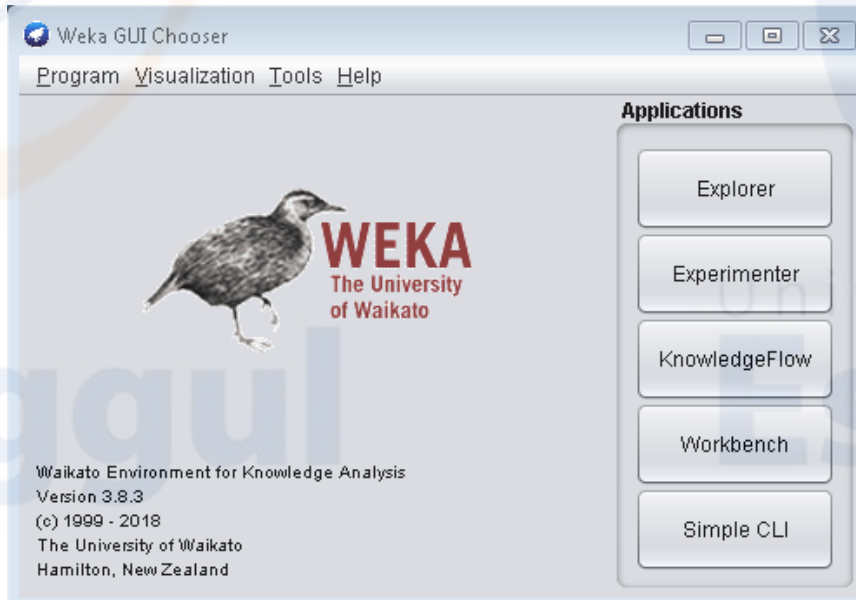
INTRODUCTION TO WEKA

Universitas

Esa Unggul



What is WEKA



- WEKA (Waikato Environment for Knowledge Analysis) is a free and open source (licensed GPL) tool that contains a collection of machine learning and preprocessing algorithms.
- Weka can be used starting from preprocessing, process, evaluation until visualization.

Weka provides interfaces for processing datasets

- **Explorer** : Explorer is used to visualize data and look for the most appropriate algorithm. All data is loaded into memory so that it can be quickly processed, but can only be used for a limited amount of data. Explorer can be used for preprocessing, association rule, classification, clustering, selecting attributes and data visualization.
- **Experimentary** : Used to find suitable parameters. Similar to explorer but the process can be automated. Large-scale experiments (multi-machine) can be done with this interface.
- **KnowledgeFlow** : Used to process data streams. Process configuration can be arranged and can handle large data. Supports incremental learning.
- **Workbench** : an environment that combines all of the GUI interfaces into a single interface. It is useful if you find yourself jumping a lot between two or more different interfaces, such as between the Explorer and the Experiment Environment. This can happen if you try out a lot of what's in the Explorer and quickly take what you learn and put it into controlled experiments.
- **Simple CLI (Command Line Interface)** : provides a simple command-line interface and allows direct execution of Weka commands



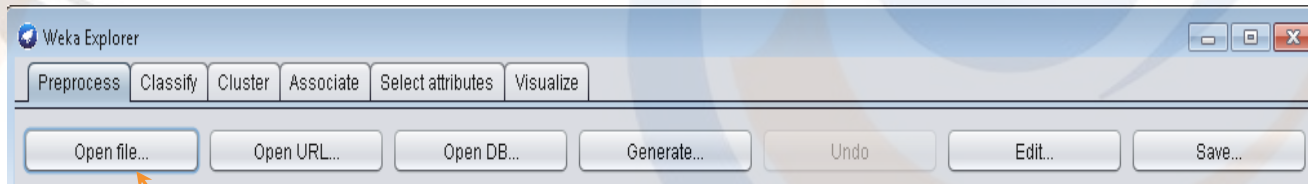
Explorer Module

Universitas

Esa Unggul

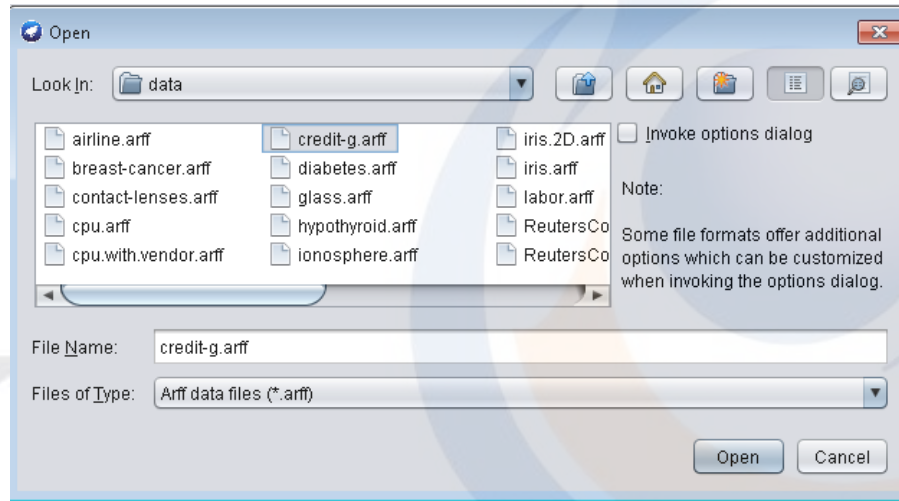


Selecting the Explorer menu the following display will appear, there are **tabs for preprocess, classify, cluster, associate, select attributes and visualize**



- Select **open file**, then select the location where Weka is uploaded and enter the data directory.
- Try to load *credit-g.arff* data that contains bank customer data related to credit.

Preprocessing



The preprocess tab will display a summary of this dataset. Important information is the number of instances (1000 rows) and the number of attributes (21 columns). Weka said dataset as "relation". In this dataset each instances has the same weight so that "Sum of weight" equals the number of instances

Current relation	
Relation: german_credit	Attributes: 21
Instances: 1000	Sum of weights: 1000

Preprocessing

Attributes

All None Invert Pattern

No.	Name
1	<input checked="" type="checkbox"/> checking_status
2	<input type="checkbox"/> duration
3	<input type="checkbox"/> credit_history
4	<input type="checkbox"/> purpose
5	<input type="checkbox"/> credit_amount
6	<input type="checkbox"/> savings_status
7	<input type="checkbox"/> employment
8	<input type="checkbox"/> installment_commitment
9	<input type="checkbox"/> personal_status
10	<input type="checkbox"/> other_parties
11	<input type="checkbox"/> residence_since
12	<input type="checkbox"/> property_magnitude
13	<input type="checkbox"/> age
14	<input type="checkbox"/> other_payment_plans
15	<input type="checkbox"/> housing
16	<input type="checkbox"/> existing_credits
17	<input type="checkbox"/> job
18	<input type="checkbox"/> ...

Remove

- Try clicking on one of the attribute names, for example: *checking_status* attribute.
- Then the panel on the right, "selected attribute" will be updated.

Selected attribute

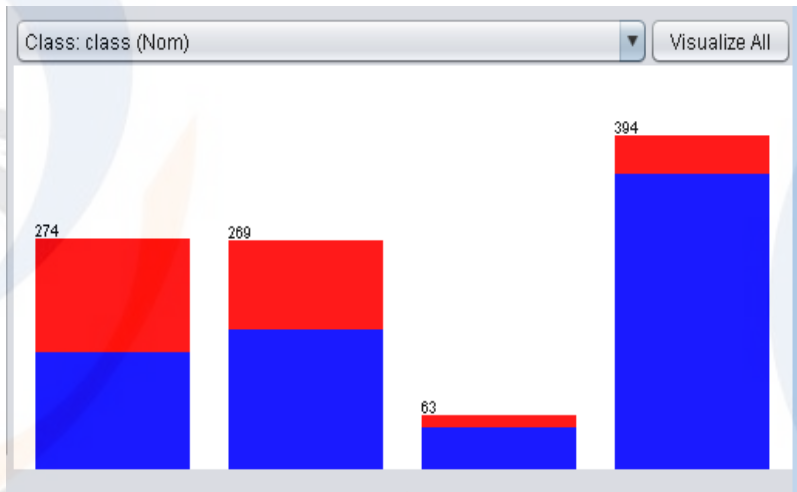
Name: checking_status Type: Nominal
Missing: 0 (0%) Distinct: 4 Unique: 0 (0%)

No.	Label	Count	Weight
1	<0	274	274.0
2	0<=X<200	269	269.0
3	>=200	63	63.0
4	no checking	394	394.0

- Nominal data type

- The table above shows the count for each category

Preprocessing



The color indicates the class to be targeted

The blue color represents good credit ("Good") and red bad credit ("Bad").

The bottom panel shows this information in the form of a histogram

Preprocessing

Now, we try selecting the "duration" attribute. The "selected attribute" panel will contain the minimum, maximum, mean and standard deviation

The screenshot shows a software interface for data analysis. On the left, the 'Current relation' panel shows 'Relation: german_credit' with 21 attributes and 1000 instances. Below it, the 'Attributes' panel lists various attributes, with 'duration' selected. On the right, the 'Selected attribute' panel displays statistics for 'duration': Name: duration, Type: Numeric, Missing: 0 (0%), Distinct: 33, Unique: 5 (1%). A table below shows the Minimum (4), Maximum (72), Mean (20.903), and StdDev (12.059). At the bottom, a histogram shows the distribution of 'duration' values, with a red bar at the far right (value 72) highlighted by an orange arrow.

Statistic	Value
Minimum	4
Maximum	72
Mean	20.903
StdDev	12.059

Class: class (Nom) Visualize All

Numeric data types

Preprocessing

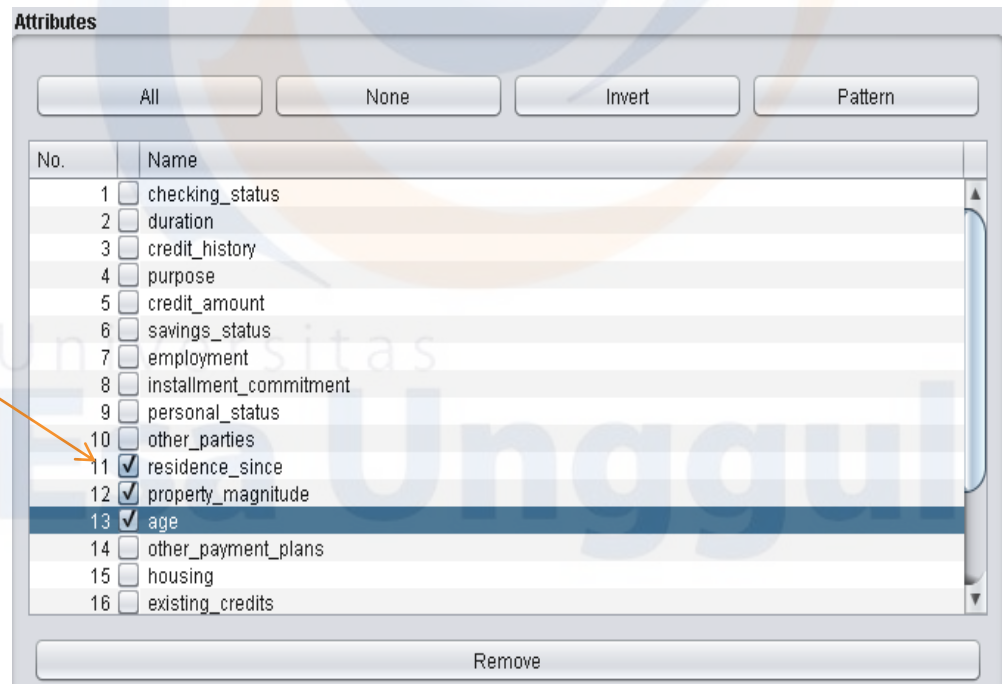
Some preprocessing that can be done are :

- **Add or remove attributes.**
- Discretization.
- Handling of missing values.
- Sampling.
- Normalization.

We will delete the attribute, select one or several attributes and click the "Remove" button (don't worry, it will be lost later)

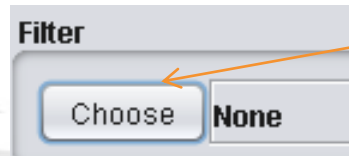
Note :

To undelete an attribute, press the "Undo" button, but if you want to save the modification, press "Save"



Preprocessing

For other pre-processing is done through filters. **Select Choose**



- Many filters available. Filters are divided into two categories : supervised and unsupervised.
- Filters also apply at the instance (row) and attribute (column) level


Preprocessing

- Add or remove attributes.
- **Discretization.**
- Handling of missing values.
- Sampling.
- Normalization.

We will try one of the filters.
For example : we will change the discrete data from the numeric attribute "Age".

Selected attribute

Name: age	Distinct: 53	Type: Numeric
Missing: 0 (0%)		Unique: 1 (0%)
Statistic	Value	
Minimum	19	
Maximum	75	
Mean	35.546	
StdDev	11.375	



The minimum value of the age attribute is 19 and a maximum of 75.

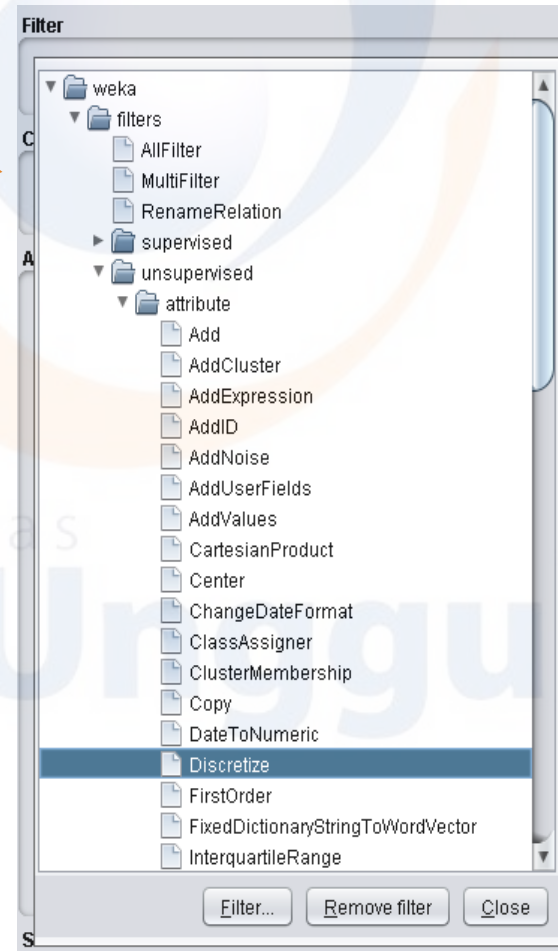
Preprocessing

We will make it discrete into 5 groups.

Select **Choose** → **Filters** → **Unsupervised**
→ **Attribute** → **Discretize**



The number after the word "Discretize" is the default filter parameter. For example -B 10 means the value will be grouped with a number of "bin" as many as 10. By default Discretize will be done for all attributes (first-last). Because we only want the "Age" attribute to be discretized, the default value needs to be changed. Click on the words "Discretize"



Preprocessing

Click on the words "Discretize"

Parameter details will appear

Edit the Indices attribute section with 13 because the Age serial number is 13 (only the age attribute will be transformed)

RangePrecision with 0 (integer) and change the value of the bin to 3 (age divided into 3 groups)

You can see the Discretize parameter change

- 11 residence_since
- 12 property_magnitude
- 13 age
- 14 other_payment_plans
- 15 housing

weka.gui.GenericObjectEditor
weka.filters.unsupervised.attribute.Discretize

About

An instance filter that discretizes a range of numeric attributes in the dataset into nominal attributes.

More
Capabilities

attributeIndices 13
binRangePrecision 0
bins 3

Filter

Choose Discretize -B 3 -M -1.0 -R 13 -precision 0

Preprocessing

Check again the attribute "Age", then it has been changed to 3 categories (<38), (38-56) and (> 56).

The screenshot shows a software interface for data preprocessing. The main window is titled "Filter" and contains a "Discretize" tool. The "Current relation" section shows the relation name and attributes. The "Attributes" section lists various attributes, with "age" selected. The "Selected attribute" section displays a table with the following data:

No.	Label	Count	Weight
1	'(-inf-38]'	656	656.0
2	'(38-56]'	276	276.0
3	'(56-inf]'	68	68.0

Below the table, a bar chart visualizes the distribution of the 'age' attribute. The x-axis represents the age categories, and the y-axis represents the count. The bars are stacked with blue at the bottom and red at the top. The counts for each category are 656, 276, and 68.

Discretization is useful for reducing the amount of data so as to speed up the process of making a model (imagine the reduction obtained if applied to millions of rows of numeric numbers). But discretization also results in missing information, for example people aged 42 and 51 will both fall into the 38-56 age category

Preprocessing

- Add or remove attributes.
- Discretization.
- **Handling of missing values.**
- Sampling.
- Normalization.

Open the *soybean.arff* file
in the [weka] / data
directory

This dataset contains data about soybean plants affected by the disease. There are 683 instances (rows) and 36.

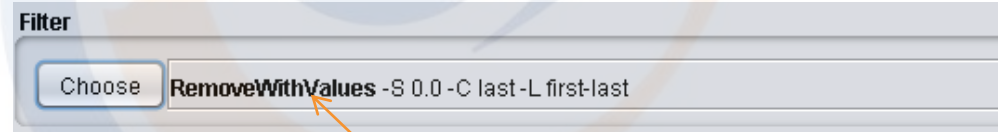
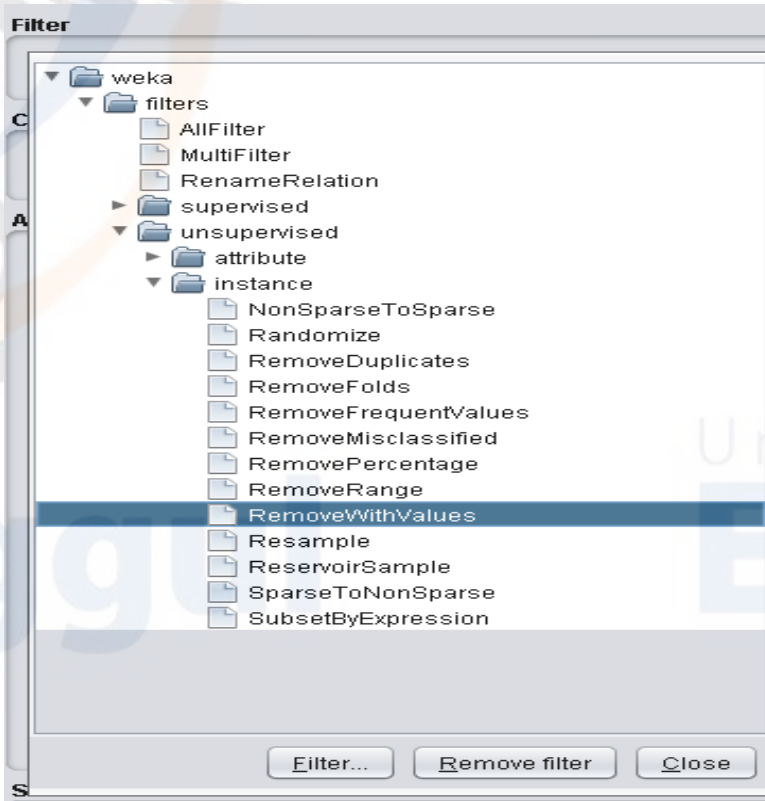
Selected attribute

Name: germination	Distinct: 3	Type: Nominal	
Missing: 112 (16%)		Unique: 0 (0%)	
No.	Label	Count	Weight
1	90-100	165	165.0
2	80-89	213	213.0
3	lt-80	193	193.0

For the missing one-line data attribute, try looking at the other attributes, there are attributes that have a missing value of 16%.

Preprocessing

In the filter section, click **Choose dan filter Filters** → **Unsupervised** → **Instance** → **RemoveWithValues**.



Click the name of the filter to configure the filter

Preprocessing

What needs to be changed is **attributeIndex** to 1 (**attribute date**), **matchMissingValues** = **True** and **invertSelection** = **True**

Selected attribute

Name: date	Distinct: 7	Type: Nominal	
Missing: 1 (0%)		Unique: 0 (0%)	
No.	Label	Count	Weight
1	april	26	26.0
2	may	75	75.0
3	june	93	93.0
4	july	118	118.0
5	august	131	131.0
6	september	149	149.0
7	october	90	90.0

Result

Filter

Choose	RemoveWithValues -S 0.0 -C 1 -L first-last -V -M
--------	---

The number of instances decreases by one and the number of missing values for the date attribute will be 0.

weka.gui.GenericObjectEditor
weka.filters.unsupervised.instance.RemoveWithValues

About

Filters instances according to the value of an attribute. More Capabilities

attributeIndex: 1

debug: False

doNotCheckCapabilities: False

don'tFilterAfterFirstBatch: False

invertSelection: True

matchMissingValues: True

modifyHeader: False

nominalIndices: first-last

splitPoint: 0.0

Open... Save... OK Cancel

Preprocessing

Result, there is no data missing

Filter

Choose RemoveWithValues - S 0.0 - C 1 - L first-last - V - M Apply Stop

Current relation

Relation: soybean-weka.filters.unsupervised.instance.RemoveWithValues-S0.0-C1-Lfirs... Attributes: 36
Instances: 682 Sum of weights: 682

Selected attribute

Name: date
Missing: 0 (0%) Distinct: 7 Type: Nominal
Unique: 0 (0%)

No.	Label	Count	Weight
1	april	26	26.0
2	may	75	75.0
3	june	93	93.0
4	july	118	118.0
5	august	131	131.0
6	september	149	149.0
7	october	90	90.0

Attributes

All None Invert Pattern

No.	Name
1	<input checked="" type="checkbox"/> date
2	<input type="checkbox"/> plant-stand
3	<input type="checkbox"/> precip
4	<input type="checkbox"/> temp
5	<input type="checkbox"/> ...

Preprocessing

Another alternative is to replace the missing value with the mode or median value in the dataset

choose **filters** → **unsupervised** → **attribute** → **ReplaceMissingValues**

The filter will replace all missing values with a **median for numeric type attributes** and **mode for nominal type attributes**

The screenshot shows the Weka Explorer interface with the 'ReplaceMissingValues' filter selected. The 'Current relation' is 'soybean-weka.filters.unsupervised.attribute R...' with 38 attributes and 683 instances. The 'Selected attribute' is 'germination', which is a nominal type with 3 distinct values and 0 missing values. The 'Attributes' list on the left includes 'date', 'plant-stand', 'precip', 'temp', 'hail', 'crop-hist', 'area-damaged', 'severity', 'seed-Int', 'germination', 'plant-growth', 'leaves', 'leafspots-halo', 'leafspots-marg', 'leafspot-size', 'leaf-shread', 'leaf-malf', and 'leaf-rot'. The 'Class' is set to 'class (Nom)'. Below the attribute table, there are three stacked bar charts representing the distribution of the 'germination' attribute.

No.	Label	Count	Weight
1	90-100	165	165.0
2	80-89	325	325.0
3	lt-80	193	193.0

Preprocessing

If we want to replace **the missing value with a certain value**

It can be used **filters** → **unsupervised** → **attribute** → **ReplaceMissingWithUserConstant**

For example : we want to replace the empty value in **the "plant-stand" attribute**.

The screenshot shows the Weka software interface. On the left, the 'Current relation' panel displays 'Relation: soybean-weka.filters.unsupervised.instance.RemoveWithValues-S0.0-C1-Lfirs...' and 'Instances: 682'. Below it, the 'Attributes' panel shows a list of attributes: 'date', 'plant-stand', 'precip', and 'temp'. The 'plant-stand' attribute is selected. On the right, the 'Selected attribute' panel shows 'Name: plant-stand', 'Missing: 35 (5%)', 'Distinct: 2', and 'Type: Nominal'. Below this, a table displays the distribution of values for the selected attribute.

No.	Label	Count	Weight
1	normal	354	354.0
2	lt-normal	293	293.0

There are 35 missing values

Preprocessing

We will replace the missing value with a new value that is "not-available"

Select the **ReplaceMissingWithUserConstant** filter

There are 36 missing values

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Open file... Open URL... Open DB... Generate... Undo Edit... Save...

Filter: Choose **ReplaceMissingWithUserConstant** -A first-last -R 0 -F %%%-MM-dd(T)H-H:mm:ss* Apply Stop

Current relation: Relation: soybean Instances: 683 Attributes: 36 Sum of weights: 683

Selected attribute: Name: plant-stand Missing: 36 (5%) Distinct: 2 Type: Nominal Unique: 0 (0%)

No.	Label	Count	Weight
1	normal	354	354.0
2	il-normal	293	293.0

Attributes: All None Invert Pattern

No.	Name
1	date
2	plant-stand
3	precip
4	temp
5	day
6	crop-hist
7	area-damaged
8	severity
9	seed-Int
10	germination
11	plant-growth
12	leaves
13	leafspots-halo
14	leafspots-marg
15	leafspot-size
16	leaf-shread
17	leaf-malf
18	leaf-rot

Class: class (Nom) Visualize All

354 293

Preprocessing

Set the parameters as follows : **Attributes = 2; NominalStringValue = "not-available"**

Result

There are **no more missing values** and **there are additional new values that are not available as many as 35.**

Selected attribute

No.	Label	Count	Weight
1	not-available	35	35.0
2	normal	354	354.0
3	lt-normal	293	293.0

Name: plant-stand
Missing: 0 (0%)
Distinct: 3
Type: Nominal
Unique: 0 (0%)

weka.gui.GenericObjectEditor
weka.filters.unsupervised.attribute.ReplaceMissingWithUserConstant

About

Replaces all missing values for nominal, string, numeric and date attributes in the dataset with user-supplied constant values.

attributes: 2

dateFormat: yyyy-MM-ddTTHH:mm:ss

dateReplacementValue:

debug: False

doNotCheckCapabilities: False

ignoreClass: False

nominalStringValue: not-available

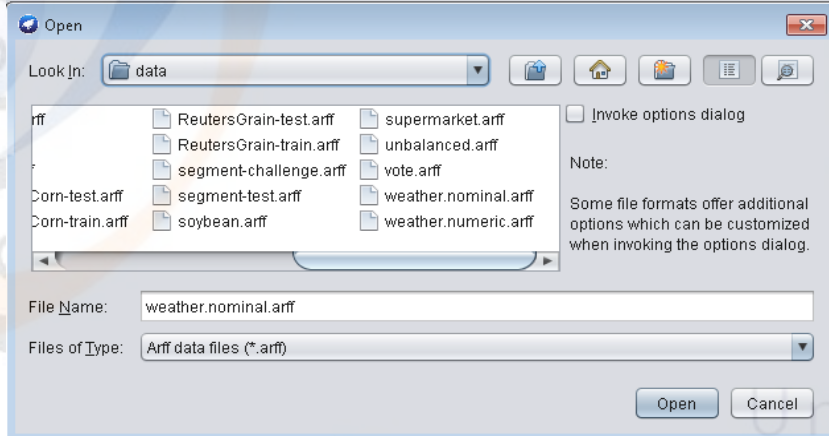
numericReplacementValue: 0

Open... Save... OK Cancel

Preprocessing

Load Data

On the **Preprocess** tab, select the **Open file button** and navigate to the data folder in the weka directory then **select *weather.nominal.arff***.



Result

Filter

Choose None Apply Stop

Current relation
Relation: weather.symbolic Attributes: 5
Instances: 14 Sum of weights: 14

Selected attribute
Name: play Type: Nominal
Missing: 0 (0%) Distinct: 2 Unique: 0 (0%)

No.	Label	Count	Weight
1	yes	9	9.0
2	no	5	5.0

Attributes
All None Invert Pattern

No.	Name
1	outlook
2	temperature
3	humidity
4	windy
5	play

Class: play (Nom) Visualize All

Remove

The *weather.nominal.arff* dataset has 4 attributes

Classification

- This dataset has 14 instances (or lines or examples) labeled Yes (9 instances), and No (5 instances).
- Each instance states weather conditions for one day, while the **Yes** label indicates that the day is **suitable for playing tennis**, while the **No** label states that the day is **not suitable for playing tennis**.

To be clearer, click the Edit button (next to Undo) to see this dataset in tabular form

We will make a model that can predict whether it is suitable to play tennis (play = Yes) or not (play = No) based on the weather

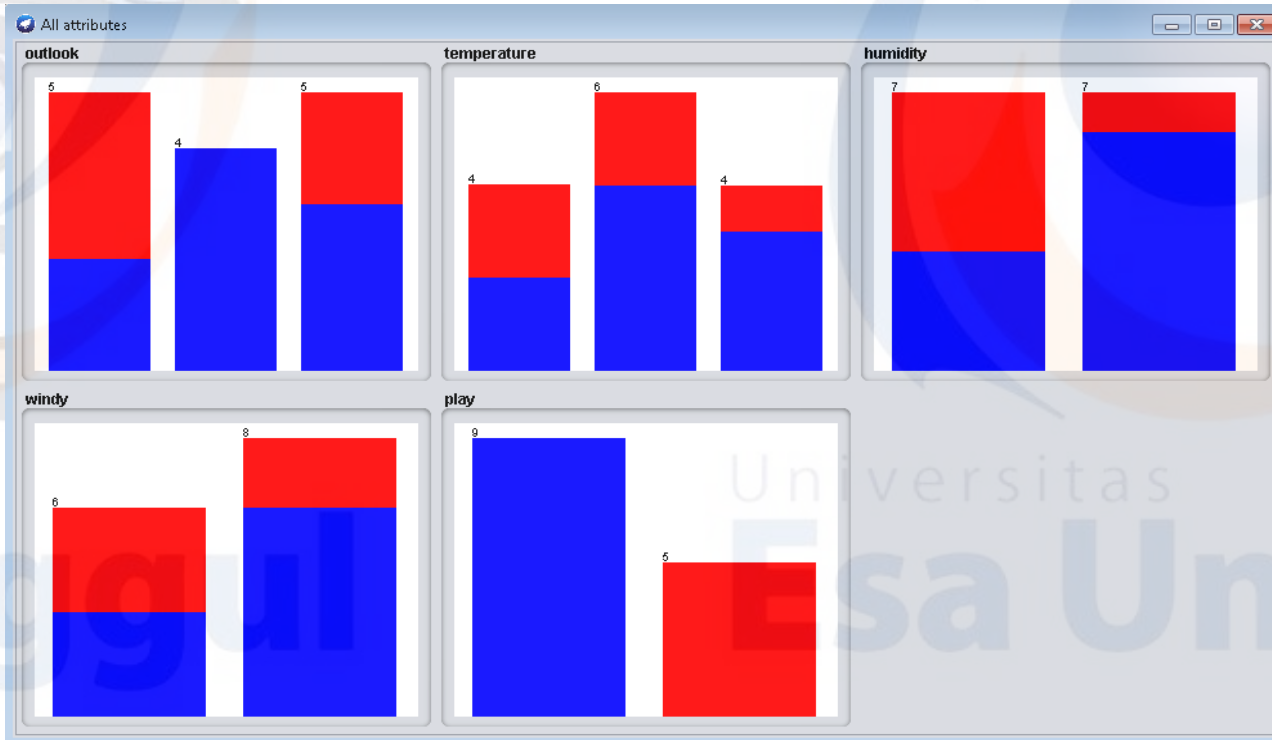
It is interesting to see that there are 4 instances with the value outlook =overcast, all of which are labeled play = Yes.

Relation: weather.symbolic

No.	1: outlook Nominal	2: temperature Nominal	3: humidity Nominal	4: windy Nominal	5: play 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

Classification

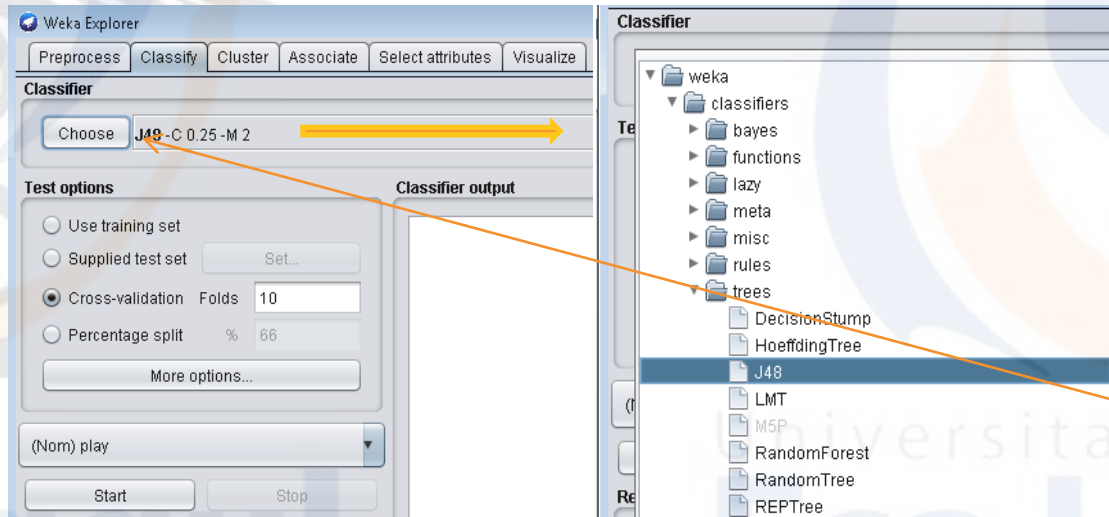
If the **Visualize All** button is selected on the bottom right, a visualization of the proportion of classes will appear for each attribute value



It is interesting to see that there are 4 instances with the value outlook = overcast, all of which are labeled play = Yes.

Classification

Model Making

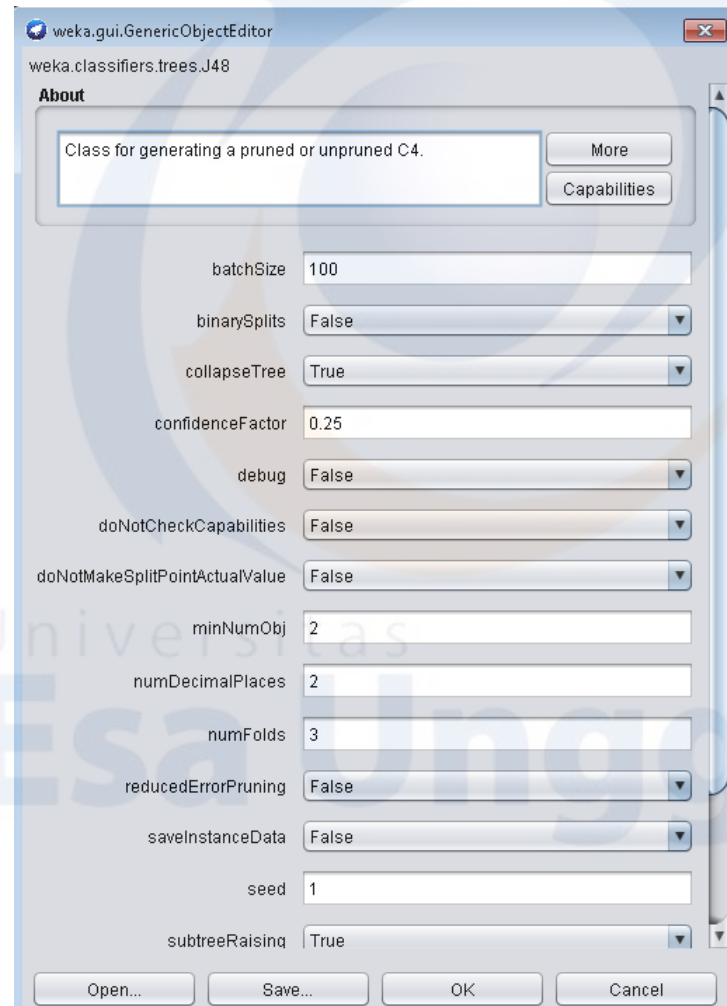


- Select the **Classify** tab.
- In the Classifier section, there is the **Choose** button and the name of the machine learning algorithm that has parameters.
- Select the **Choose** button, a dropdown menu will appear
- Click **trees**, and choose **J48 (not ID3, because my Weka didn't provide ID3 algorithm)**
- In the text, it says "**J48 -C 0.25 -M 2**".
- If **the text is clicked**, the parameter dialog will come out whose value can be adjusted.

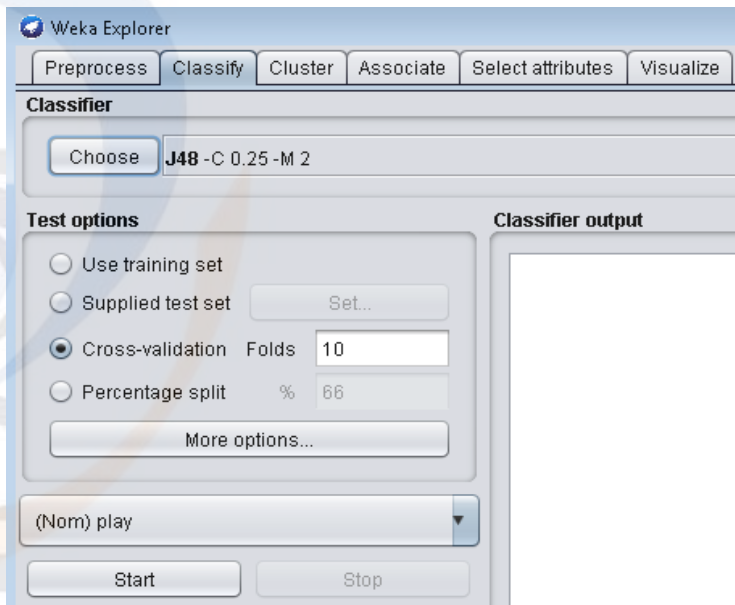
Classification

The parameter dialog

- Before starting the learning process (or learning) by **pressing the Start button**, make sure the class attributes are appropriate.
- In our dataset, the class attribute is play. In addition, choose the appropriate test options.



Classification



Here are the details of the test options :

- **Use training set:** the model evaluates how well it predicts the class of all data instances. This is rather dangerous because it can cause accuracy to be artificially high.
- **Supplied test set:** the model evaluates how well it predicts the class of test data instances that are loaded from separate files. This test data is in the form of ARFF and of course it must be the same attribute as the training data.
- **Cross validation:** the model is evaluated by cross-validation, with the number of folds according to user input (default: 10). For example if the training data have 1000 rows and 10 folds are set (ten cross validation), then for the first batch, rows 1-100 are used for testing and rows 101-1000 are used for training. For the second batch 101-200 for testing and 1-100 plus 201-1000 for training. And so on in turns until batch 10. Accuracy of each batch is calculated and the accuracy of the model is the average of all batches. Cross validation is more commonly used.
- **Percentage split:** the model evaluates how well it predicts the class of instances which are certain% of all data (hold-out). Large% according to user input (default: 66). Training data is broken up into two according to percentage, the first part is testing data, the second part is training data.

Classification

After all settings are correct, click **Start**.

Results in the **Classifier output**, There are 5 parts that are displayed as output, i.e :

1. **Run information:** the list of learning information includes the scheme (learning algorithm and its parameters), the name of the relationship (defined in the arff file), number of instances, number and list of attributes, and test options.

Classifier output

```
=== Run information ===  
  
Scheme:      weka.classifiers.trees.J48 -C 0.25 -M 2  
Relation:    weather.symbolic  
Instances:   14  
Attributes:  5  
              outlook  
              temperature  
              humidity  
              windy  
              play  
Test mode:   10-fold cross-validation
```

Classification

2. **Classifier model (full training set):** learning model that results from all training data in text representation. The following is a decision tree model written in text representation. **note:** whatever test options are chosen, the classifier model that is output is the model that is built from all training data (full training sets).

```
Classifier output

=== Classifier model (full training set) ===

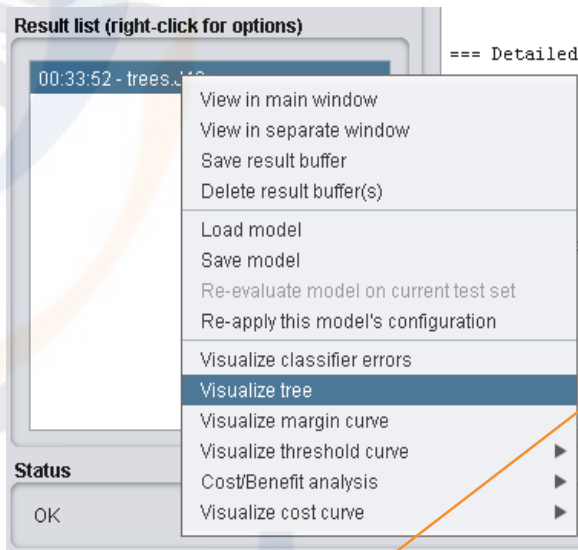
J48 pruned tree
-----

outlook = sunny
|  humidity = high: no (3.0)
|  humidity = normal: yes (2.0)
outlook = overcast: yes (4.0)
outlook = rainy
|  windy = TRUE: no (2.0)
|  windy = FALSE: yes (3.0)

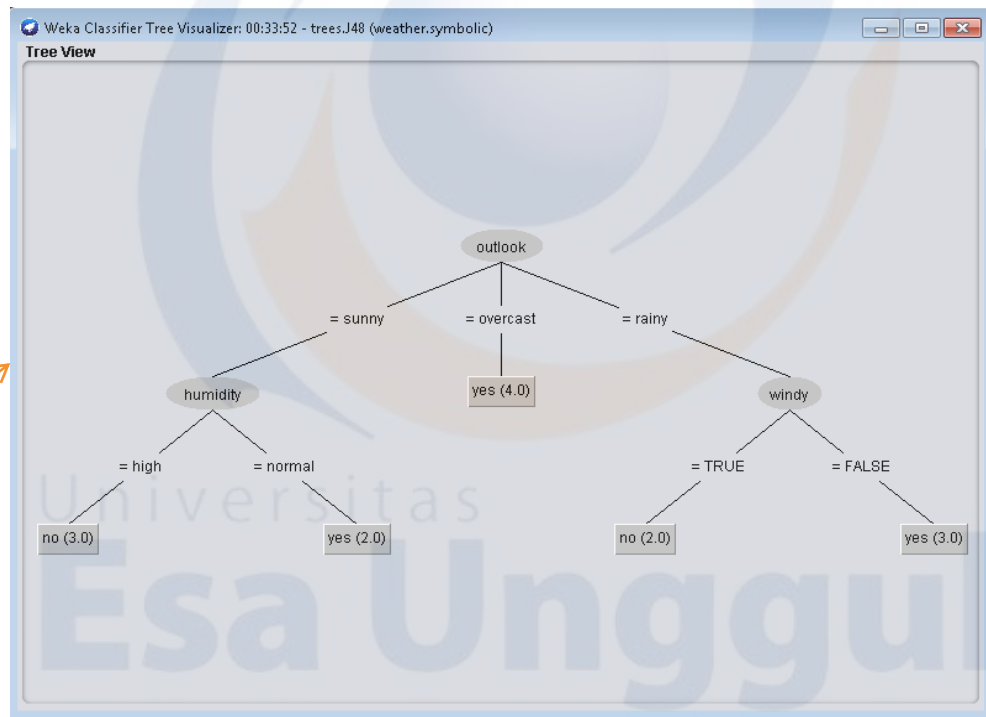
Number of Leaves :    5
Size of the tree :    8
```

Classification

To see the visualization tree, right-click the result list (bottom image) then select visualize tree.



Result



Based on this decision tree, the model can be seen clearly. If there is an Outlook input: Sunny; Humidity: Normal, then to predict the tree trace from the outlook attribute, go down to the humidity branch on the left and get to the end (leaf) labeled "yes".

Classification

3. **Summary:** list of performance statistics that show how accurately the classifier model can predict the actual class of each instance according to test options. For the 10-fold cross validation test options selected, there are 14 instances tested in 10 iterations. In the results shown, there were 7 instances (50%) that were correctly predicted by the class, and 7 other instances (50%) were incorrectly predicted. The accuracy of this model is still poor, 50% means the accuracy of this model is the same as tossing a coin. Indeed the dataset used is still an experimental dataset

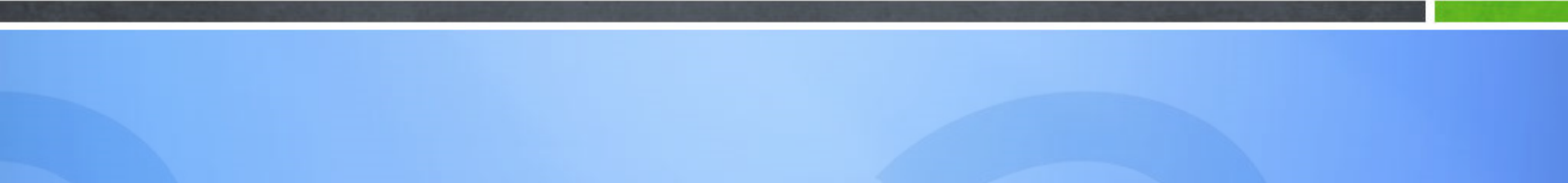
```
Classifier output
Time taken to build model: 0.06 seconds

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

Correctly Classified Instances      7          50   %
Incorrectly Classified Instances    7          50   %
Kappa statistic                    -0.0426
Mean absolute error                 0.4167
Root mean squared error             0.5984
Relative absolute error             87.5   %
Root relative squared error         121.2987 %
Total Number of Instances          14
```



Classification



4. **Detailed accuracy by class:** a more detailed measure of performance at the class level.

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

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,556	0,600	0,625	0,556	0,588	-0,043	0,633	0,758	yes
	0,400	0,444	0,333	0,400	0,364	-0,043	0,633	0,457	no
Weighted Avg.	0,500	0,544	0,521	0,500	0,508	-0,043	0,633	0,650	

5. **Confusion matrix:** a matrix that shows how many instances are predicted to each class (per column) and the number of instances that correspond to the actual label (per row). In this example there are 8 instances that are predicted or classified as Yes. Of these 8 instances, 5 instances are labeled Yes (also called True Positive), and 3 instances are labeled No (also called False Positive). The Correctly Classified Instances value in Summary 7 (50%) is a 5 + 2 value from this confusion matrix.

```
=== Confusion Matrix ===
 a b  <-- classified as
 5 4 | a = yes
 3 2 | b = no
```

Universitas

Esa Unggul

Classification

We can determine whether our ROC curve is good or not by looking at AUC (Area Under the Curve) and other parameters which are also called as Confusion Metrics

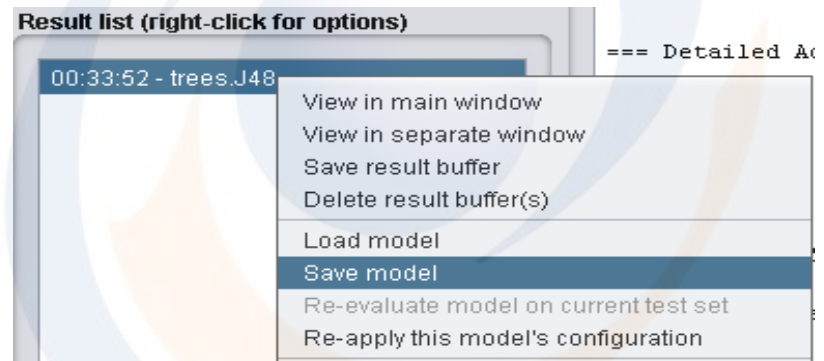
	Predicted class		
	Class = Yes	Class = No	
Actual Class	Class = Yes	True Positive	False Negative
	Class = No	False Positive	True Negative

- **True Positives (TP)** - These are the correctly predicted positive values which means that the value of actual class is yes and the value of predicted class is also yes.
- **True Negatives (TN)** - These are the correctly predicted negative values which means that the value of actual class is no and value of predicted class is also no.
- **False Positives (FP)** – When actual class is no and predicted class is yes.
- **False Negatives (FN)** – When actual class is yes but predicted class in no.

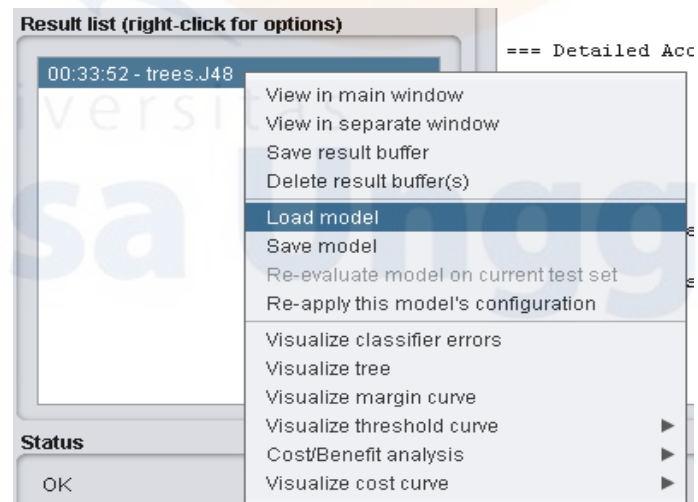
Matrix Confusion

Save the Model

Click the result list to be saved,
then right-click and "Save Model"

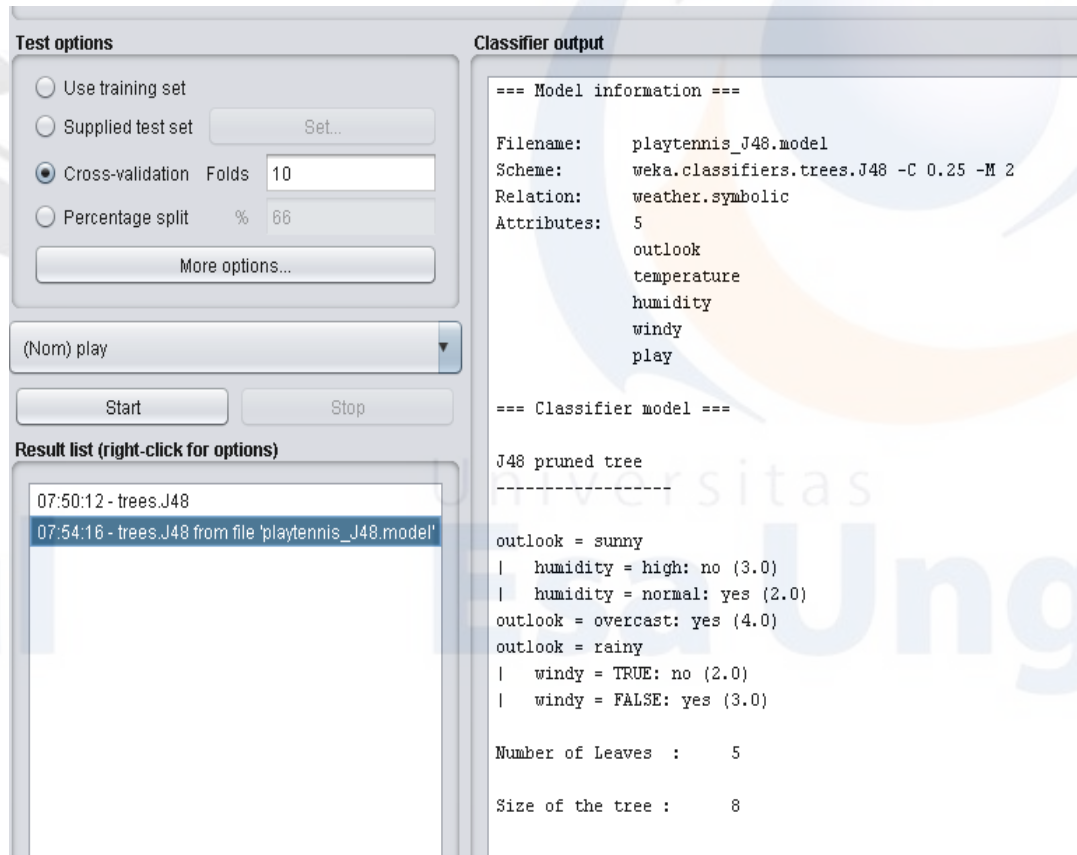


And after saving, if needed we
can load it



Classification

In the result list it will appear like this, meaning that the model was successfully loaded



The screenshot displays the Weka software interface. On the left, the 'Test options' panel is active, showing 'Cross-validation' selected with 10 folds. Below it, a dropdown menu shows '(Nom) play' and 'Start'/'Stop' buttons. The 'Result list' shows two entries: '07:50:12 - trees.J48' and '07:54:16 - trees.J48 from file 'playtennis_J48.model'', with the latter selected. On the right, the 'Classifier output' panel shows the following text:

```
=== Model information ===
Filename:    playtennis_J48.model
Scheme:     weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:   weather.symbolic
Attributes: 5
            outlook
            temperature
            humidity
            windy
            play

=== Classifier model ===

J48 pruned tree
-----
outlook = sunny
|  humidity = high: no (3.0)
|  humidity = normal: yes (2.0)
outlook = overcast: yes (4.0)
outlook = rainy
|  windy = TRUE: no (2.0)
|  windy = FALSE: yes (3.0)

Number of Leaves :    5
Size of the tree :    8
```

Classification

Universitas
Thank You Very Much