

Universidade Federal do Paraná (UFPR)
Especialização em Engenharia Industrial 4.0

Introdução ao Weka

Data Mining with Open Source Machine Learning
Software in Java

David Menotti

www.inf.ufpr.br/menotti/am-221

Hoje

- Weka
 - Introdução
 - Como instalar
 - Datasets
 - Usando algoritmos de:
 - Classificação
 - Clustering
 - Regressão

Introdução

- Weka é uma coleção de algoritmos de aprendizado de máquina para tarefas de mineração de dados. Ele contém ferramentas para preparação de dados, classificação, regressão, agrupamento, mineração de regras de associação e visualização.
- Encontrada apenas nas ilhas da Nova Zelândia, a **Weka** é uma ave que não voa e tem uma natureza inquisitiva. O nome é pronunciado assim, e o pássaro soa assim.

Introdução

- Weka é um software de código aberto emitido sob a ***GNU General Public License***.
- Sim, é possível aplicar a Weka para processar **big data** e realizar aprendizado profundo (**deep learning**)!

Introdução

- No [site](https://www.cs.waikato.ac.nz/ml/weka/courses.html), existem vários cursos on-line gratuitos que ensinam aprendizado de máquina e mineração de dados usando o Weka. Confira no [site](https://www.cs.waikato.ac.nz/ml/weka/courses.html) os cursos para detalhes sobre quando e como se inscrever. Os vídeos dos cursos estão disponíveis no Youtube.

<https://www.cs.waikato.ac.nz/ml/weka/courses.html>



Machine Learning Group at the University of Waikato

[Project](#)

[Software](#)

[Book](#)

[Courses](#)

[Publications](#)

[People](#)

[Related](#)

Free online courses on data mining with machine learning techniques in Weka

To help you explore the Weka software and learn about machine learning techniques for data mining and how to apply them, we have put together a series of three online courses that come with videos and plenty of exercises! They are hosted on the **FutureLearn** platform and are free of charge, but you can upgrade to receive an official FutureLearn Certificate of Achievement to use when applying for jobs or courses.

Data Mining with Weka

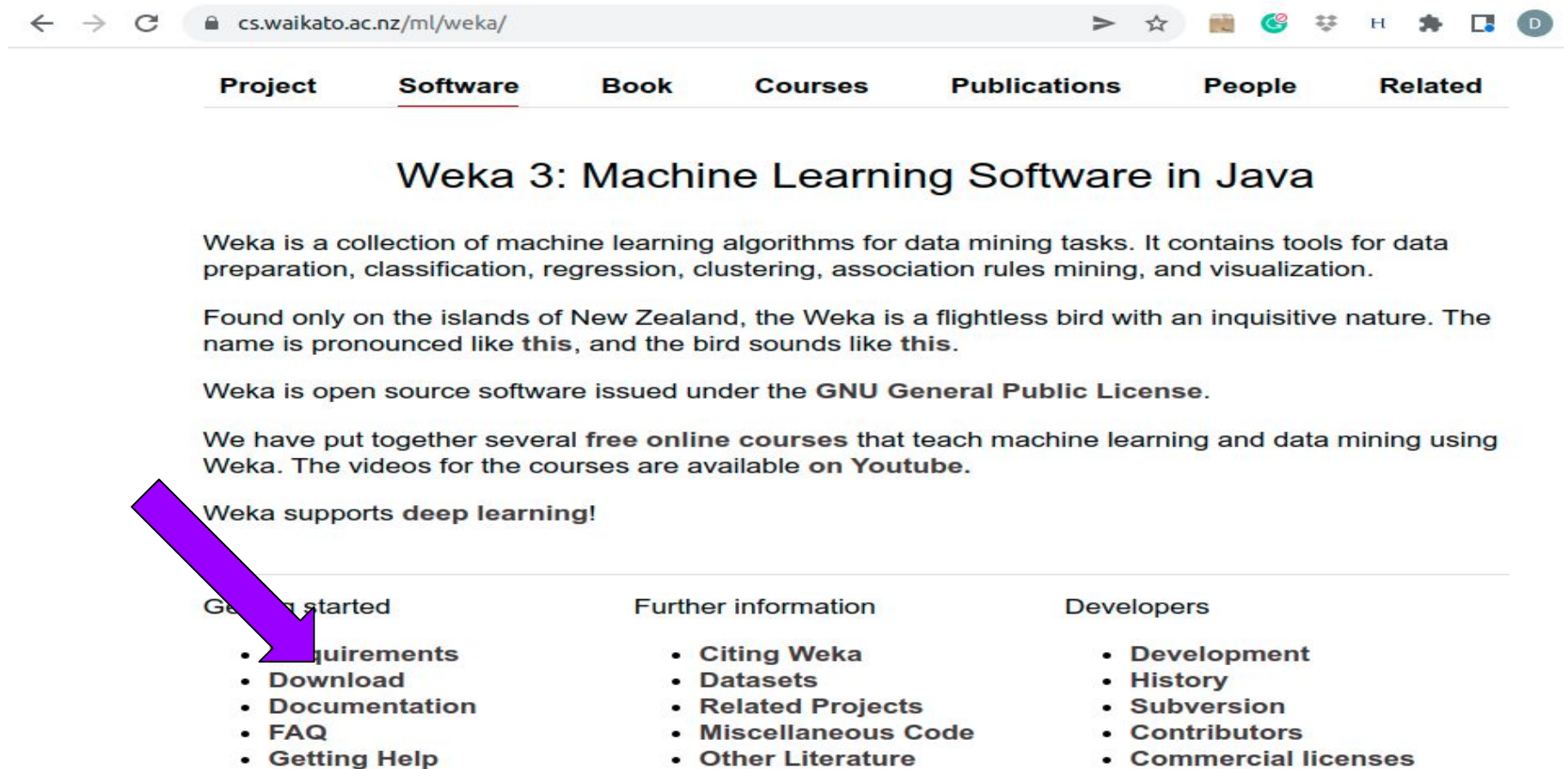
Everybody talks about Data Mining and Big Data nowadays. Weka is a powerful, yet easy to use tool for machine learning and data mining. **Data Mining with Weka** introduces you to practical data mining.

Manual

- Weka Manual
 - (v3-8-1) 19/12/2016
 - https://alvarofpinheiro.eti.br/_files/200000910-c2f11c3ec2/Weka%20Manual-3-8-1.pdf
 - (v3-7-8) 21/01/2013
 - https://statweb.stanford.edu/~lpekelis/13_datafest_cart/WekaManual-3-7-8.pdf
 - (v3-6-8) 03/05/2012
 - <http://www.nilc.icmc.usp.br/elc-ebralc2012/minicursos/WekaManual-3-6-8.pdf>

Como Instalar

- Weka website (latest version 3.8/3.9)
 - <https://www.cs.waikato.ac.nz/ml/weka/>



The screenshot shows the Weka website homepage. The browser address bar displays 'cs.waikato.ac.nz/ml/weka/'. The navigation menu includes 'Project', 'Software' (which is underlined), 'Book', 'Courses', 'Publications', 'People', and 'Related'. The main heading is 'Weka 3: Machine Learning Software in Java'. Below this, there is a paragraph describing Weka as a collection of machine learning algorithms for data mining tasks. Another paragraph explains the origin of the name 'Weka' from New Zealand. A third paragraph states that Weka is open source software under the GNU General Public License. A fourth paragraph mentions free online courses available on YouTube. A fifth paragraph states 'Weka supports deep learning!'. At the bottom, there are three columns of links: 'Getting started' (with a purple arrow pointing to 'Requirements'), 'Further information', and 'Developers'.

Project Software Book Courses Publications People Related

Weka 3: Machine Learning Software in Java

Weka is a collection of machine learning algorithms for data mining tasks. It contains tools for data preparation, classification, regression, clustering, association rules mining, and visualization.

Found only on the islands of New Zealand, the Weka is a flightless bird with an inquisitive nature. The name is pronounced like **this**, and the bird sounds like **this**.

Weka is open source software issued under the **GNU General Public License**.

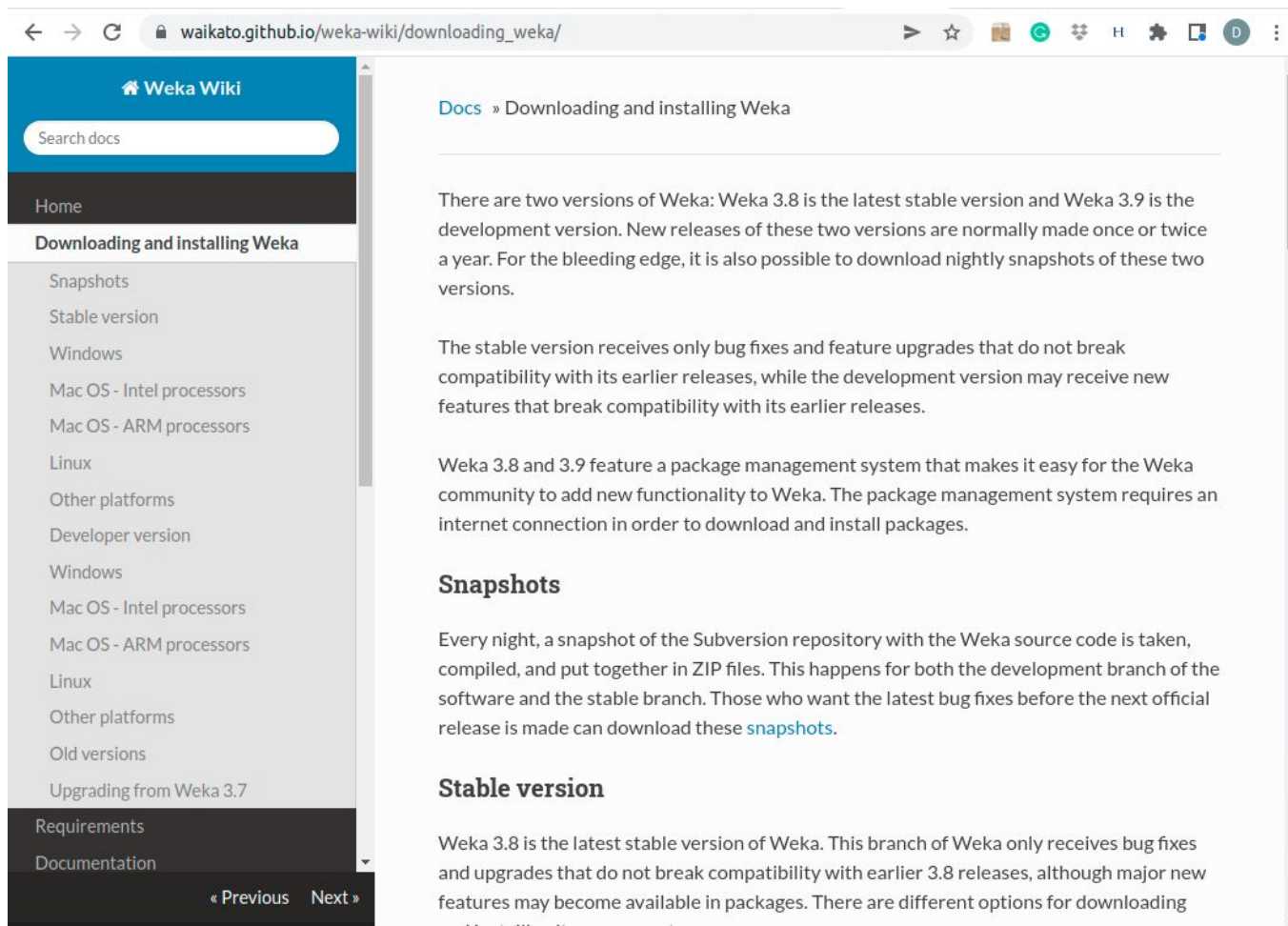
We have put together several **free online courses** that teach machine learning and data mining using Weka. The videos for the courses are available **on Youtube**.

Weka supports **deep learning**!

Getting started	Further information	Developers
• Requirements • Download • Documentation • FAQ • Getting Help	• Citing Weka • Datasets • Related Projects • Miscellaneous Code • Other Literature	• Development • History • Subversion • Contributors • Commercial licenses

Como Instalar

- Downloading: https://waikato.github.io/weka-wiki/downloading_weka/



Como Instalar

- Downloading: https://waikato.github.io/weka-wiki/downloading_weka/

The screenshot shows a web browser displaying the Weka Wiki page for downloading Weka. The page has a sidebar on the left with a search bar and a list of navigation links. The main content area is titled 'downloading_weka/' and contains instructions for Windows, Mac OS, and Linux. Three purple arrows point from the left margin to the download links for Windows, Mac OS - Intel processors, and Linux. A red arrow points from the right margin to the Mac OS - ARM processors section.

Weka Wiki

Search docs

Home

Downloading and installing Weka

- Snapshots
- Stable version
- Windows
- Mac OS - Intel processors
- Mac OS - ARM processors
- Linux
- Other platforms
- Developer version
- Windows
- Mac OS - Intel processors
- Mac OS - ARM processors
- Linux
- Other platforms
- Old versions
- Upgrading from Weka 3.7

Windows

- Click [here](#) to download a self-extracting executable for 64-bit Windows that includes Azul's 64-bit OpenJDK Java VM 17 (weka-3-8-6-azul-zulu-windows.exe; 133.2 MB)

This executable will install Weka in your Program Menu. Launching via the Program Menu or shortcuts will automatically use the included JVM to run Weka.

Mac OS - Intel processors

- Click [here](#) to download a disk image for Mac OS that contains a Mac application including Azul's 64-bit OpenJDK Java VM 17 for Intel Macs. (weka-3-8-6-azul-zulu-osx.dmg; 180 MB)

Mac OS - ARM processors

- Click [here](#) to download a disk image for Mac OS that contains a Mac application including Azul's 64-bit OpenJDK Java VM 17 for ARM Macs. (weka-3-8-6-azul-zulu-arm-osx.dmg; 166 MB)

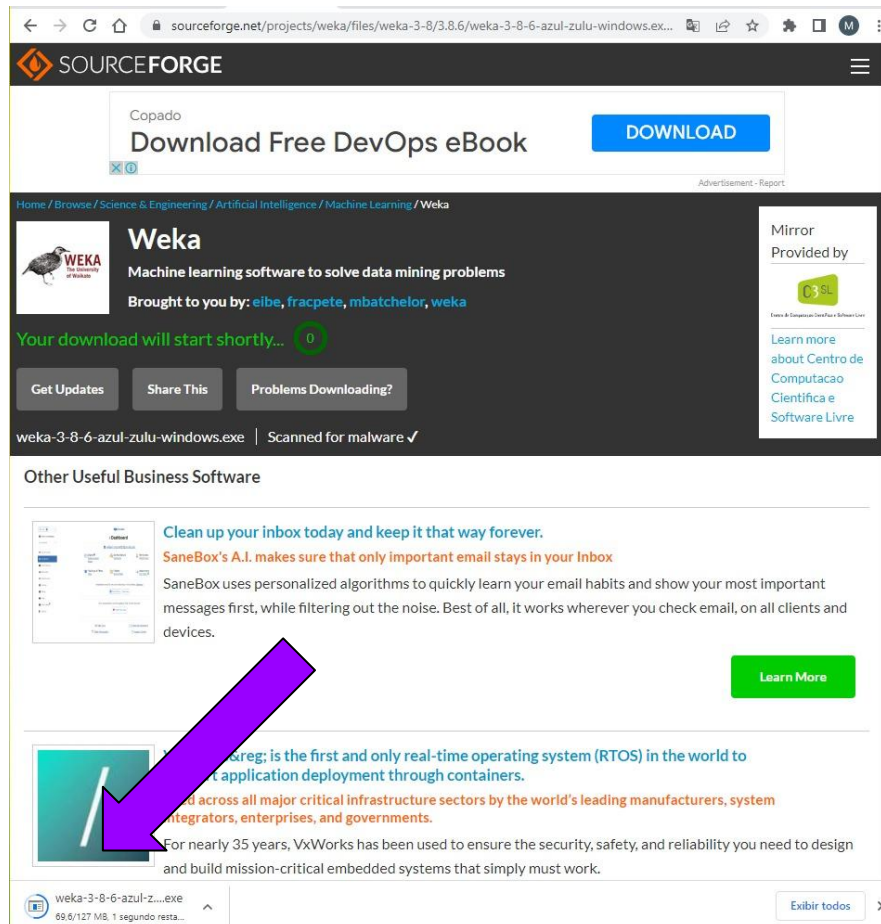
Linux

- Click [here](#) to download a zip archive for Linux that includes Azul's 64-bit OpenJDK Java VM 17 (weka-3-8-6-azul-zulu-linux.zip; 146.9 MB)

First unzip the zip file. This will create a new directory called weka-3-8-5. To run Weka, change into that directory and type

Como Instalar

- Downloading: https://waikato.github.io/weka-wiki/downloading_weka/



Como Instalar

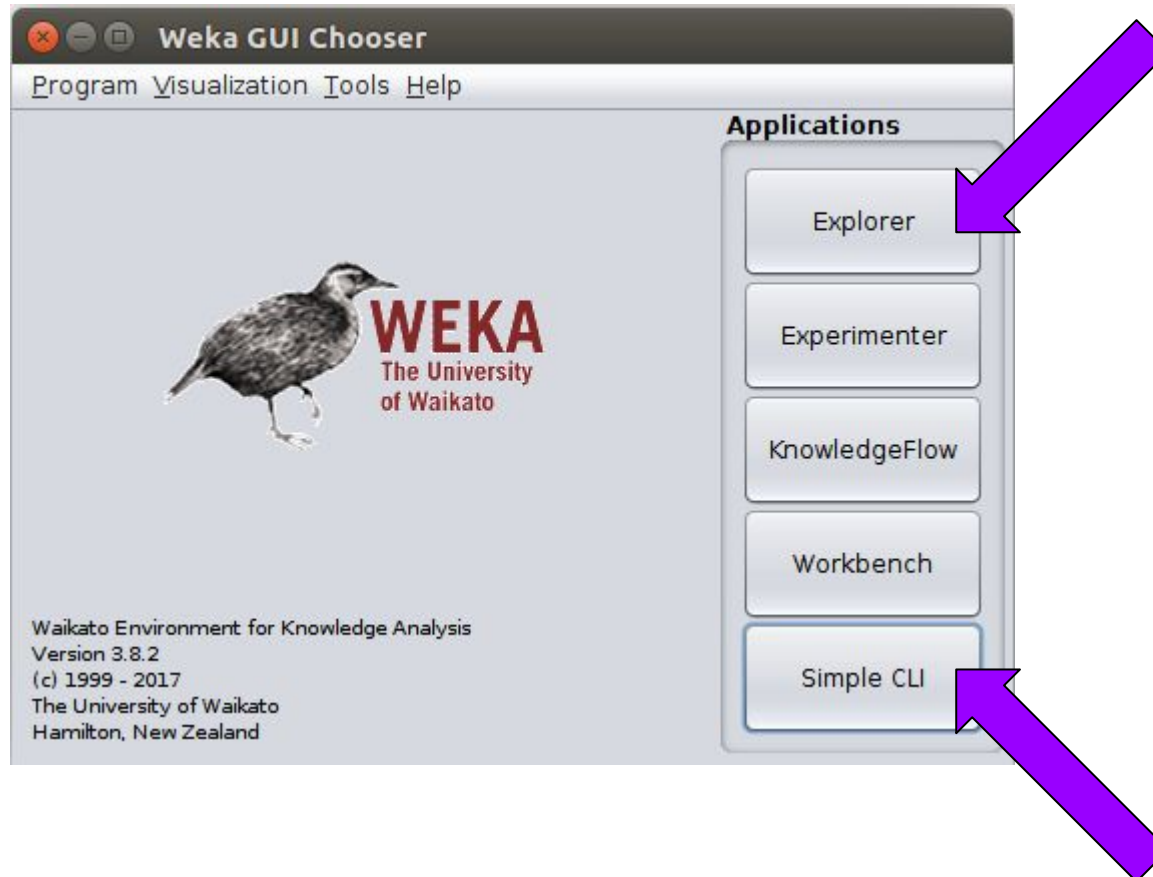
- Downloading: https://waikato.github.io/weka-wiki/downloading_weka/

Basta baixar (130Mb) e executar

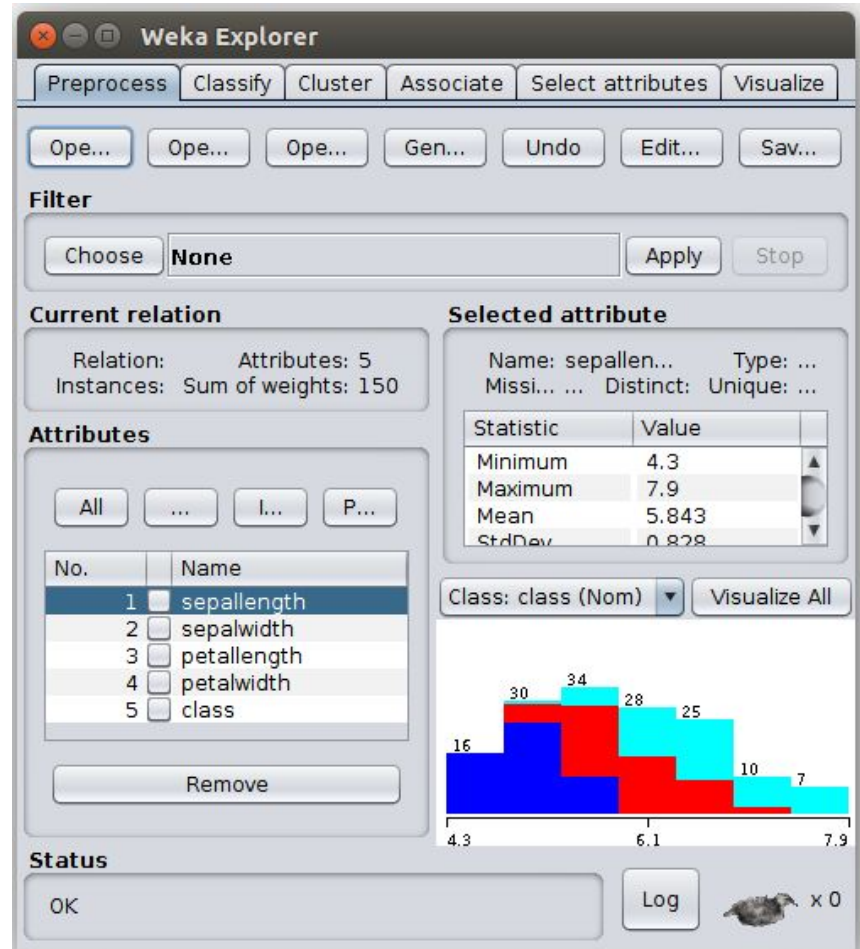
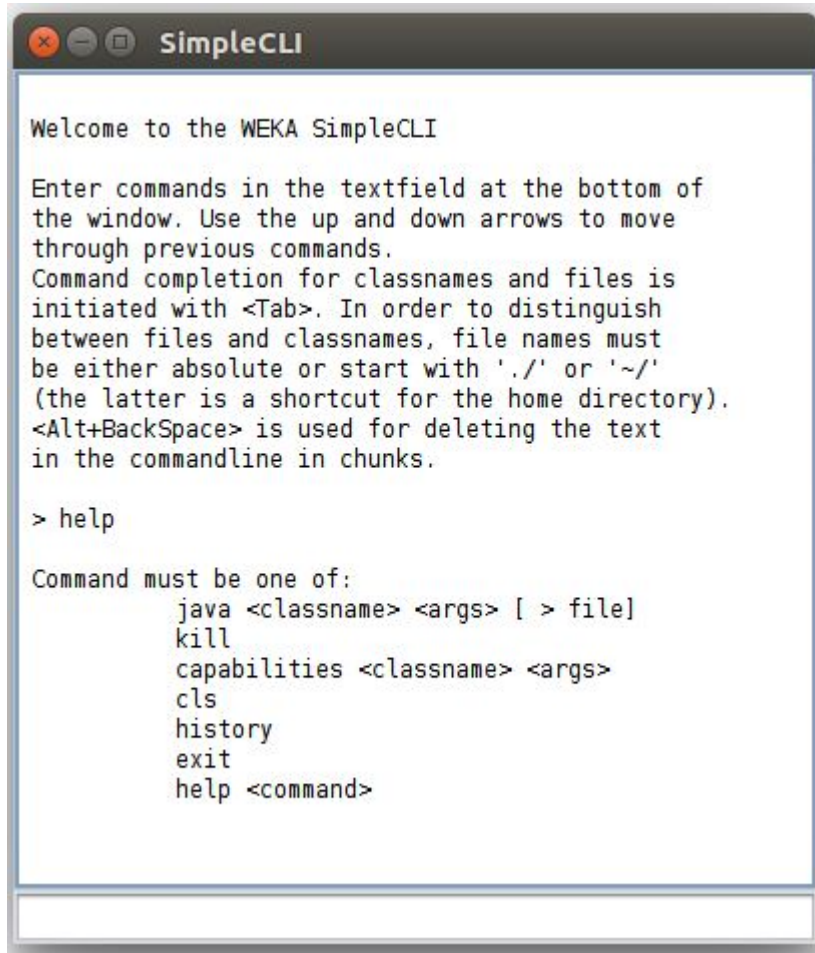
Weka-3-8-6-azul-zulu-windows.exe

CLI vs GUI

- Início



CLI vs GUI



Atributos

- *Nominal*: um de uma lista predefinida de valores
 - e.g. vermelho, azul, amarelo
- *Numérico*: Um número real ou inteiro
- *String*: delimitada por “aspas duplas”
- *Data*
- *Relational*

Arquivos ARFF

- A representação das instâncias
- Consiste em:
 - Um cabeçalho (header): Descreve os tipos de atributos e seus valores
 - Seção de dados: lista de dados separada por vírgula

Exemplo de Arquivo ARFF

Comment →

```
% This is a toy example, the UCI weather dataset
% Any relation to real weather is purely coincidental

@relation weather.symbolic ← Nome do Dataset

@attribute outlook {sunny, overcast, rainy}
@attribute temperature {hot, mild, cool}
@attribute humidity {high, normal}
@attribute windy {TRUE, FALSE}
@attribute play {yes, no} ← Atributos

@data
sunny,hot,high,FALSE,no
sunny,hot,high,TRUE,no
overcast,hot,high,FALSE,yes
rainy,mild,high,FALSE,yes
rainy,cool,normal,FALSE,yes
rainy,cool,normal,TRUE,no
overcast,cool,normal,TRUE,yes
sunny,mild,high,FALSE,no
sunny,cool,normal,FALSE,yes
rainy,mild,normal,FALSE,yes
sunny,mild,normal,TRUE,yes
overcast,mild,high,TRUE,yes
overcast,hot,normal,FALSE,yes
rainy,mild,high,TRUE,no ← Dados/Valores
```

Classe / Meta →

ARFF

```
% This is a toy example, the UCI weather dataset  
% Any relation to real weather is purely coincidental
```

```
@relation weather.symbolic
```

```
@attribute outlook {sunny, overcast, rainy}
```

```
@attribute temperature {hot, mild, cool}
```

```
@attribute humidity {high, normal}
```

```
@attribute windy {TRUE, FALSE}
```

```
@attribute play {yes, no}
```

```
@data
```

```
sunny,hot,high,FALSE,no
```

```
sunny,hot,high,TRUE,no
```

```
overcast,hot,high,FALSE,yes
```

```
rainy,mild,high,FALSE,yes
```

```
rainy,cool,normal,FALSE,yes
```

```
rainy,cool,normal,TRUE,no
```

```
overcast,cool,normal,TRUE,yes
```

```
sunny,mild,high,FALSE,no
```

```
sunny,cool,normal,FALSE,yes
```

```
rainy,mild,normal,FALSE,yes
```

```
sunny,mild,normal,TRUE,yes
```

```
overcast,mild,high,TRUE,yes
```

```
overcast,hot,normal,FALSE,yes
```

```
rainy,mild,high,TRUE,no
```

Comment

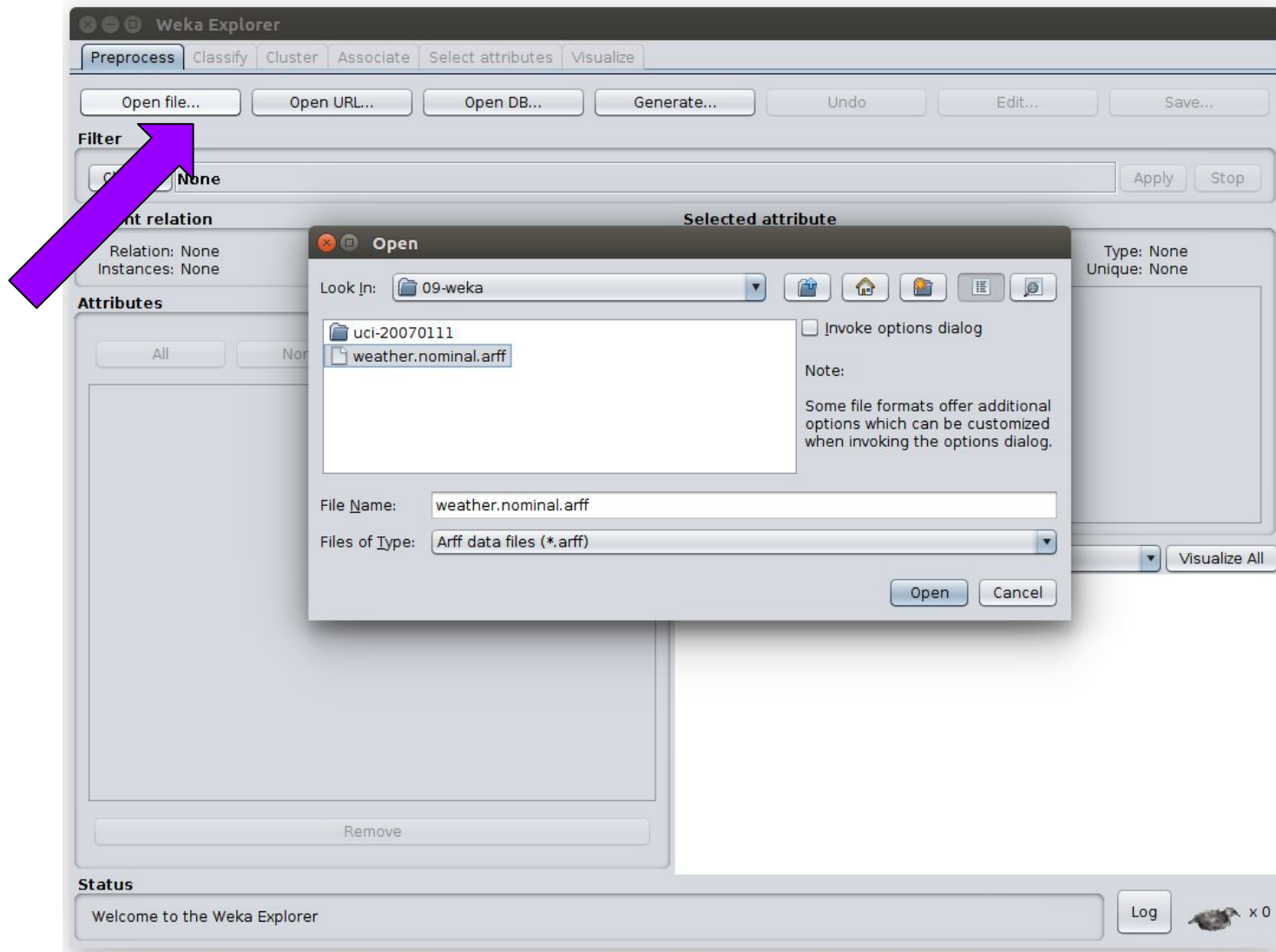
Nome do Dataset

Atributos

Classe / Meta

Dados/Valores

Abrindo um Dataset



Visualizando

Weka Explorer

Preprocess | **Classify** | Cluster | Associate | Select attributes | Visualize

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

Filter

Choose **None** Apply Stop

Current relation

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

Attributes

All None Invert Pattern

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

Remove

Selected attribute

Name: outlook Missing: 0 (0%) Distinct: 3 Type: Nominal Unique: 0 (0%)

No.	Label	Count	Weight
1	sunny	5	5.0
2	overcast	4	4.0
3	rainy	5	5.0

Class: play (Nom) Visualize All

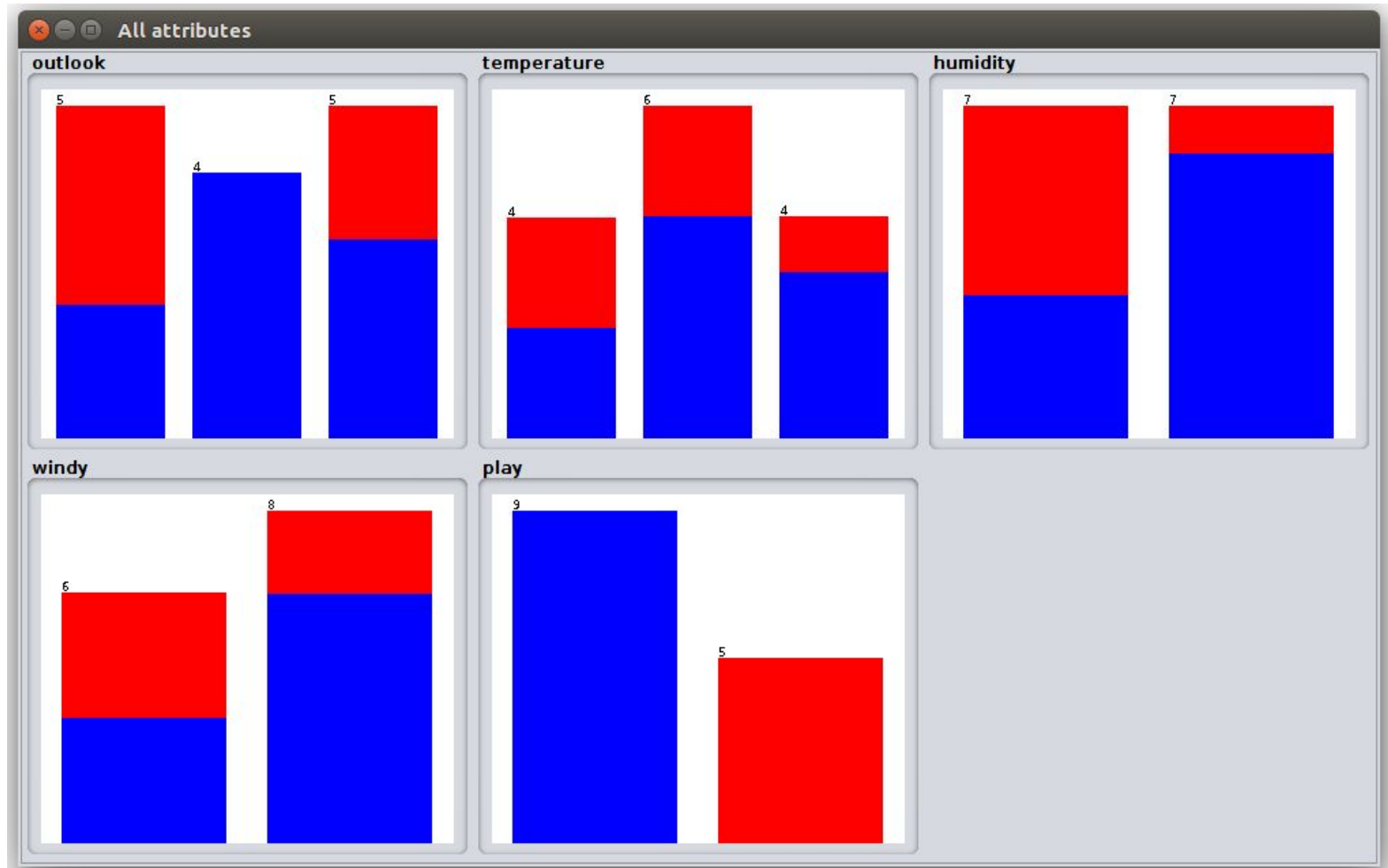
Status

OK Log x 0

Bar Chart Data:

Outlook	play = no (Blue)	play = yes (Red)
sunny	3	2
overcast	4	0
rainy	3	2

Visualizando

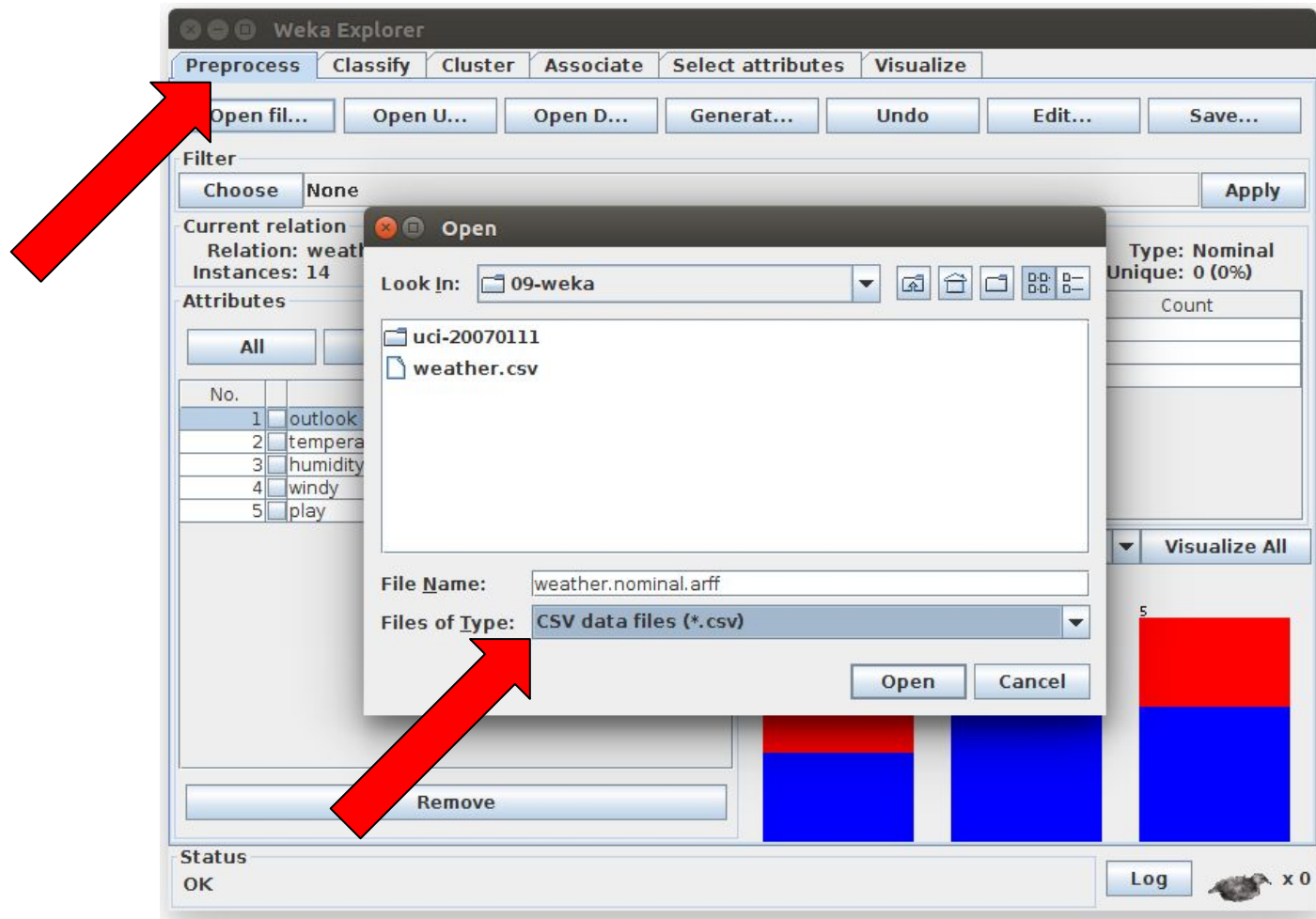


Excel => CSV

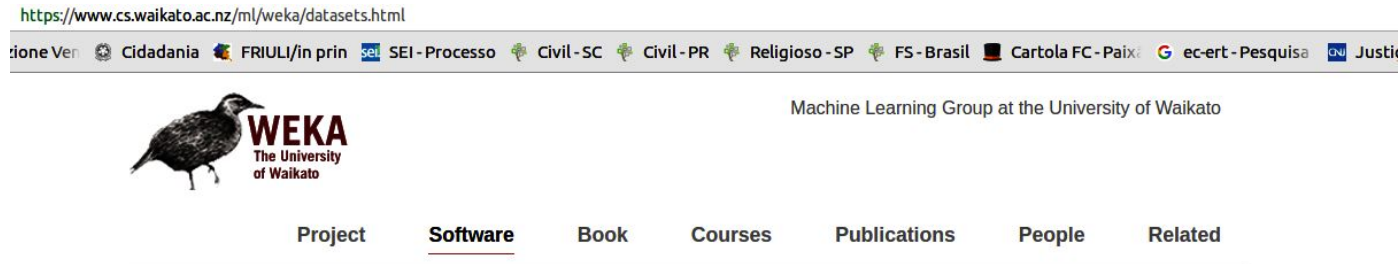
	A	B	C	D	E
1	outlook	temperature	humidity	windy	play
2	sunny	hot	high	FALSE	no
3	sunny	hot	high	TRUE	no
4	overcast	hot	high	FALSE	yes
5	rainy	mild	high	FALSE	yes
6	rainy	cool	normal	FALSE	yes
7	rainy	cool	normal	TRUE	no
8	overcast	cool	normal	TRUE	yes
9	sunny	mild	high	FALSE	no
10	sunny	cool	normal	FALSE	yes
11	rainy	mild	normal	FALSE	yes
12	sunny	mild	normal	TRUE	yes
13	overcast	mild	high	TRUE	yes
14	overcast	hot	normal	FALSE	yes
15	rainy	mild	high	TRUE	no
16					

weather.csv
outlook,temperature,humidity,windy,play
sunny,hot,high,FALSE,no
sunny,hot,high,TRUE,no
overcast,hot,high,FALSE,yes
rainy,mild,high,FALSE,yes
rainy,cool,normal,FALSE,yes
rainy,cool,normal,TRUE,no
overcast,cool,normal,TRUE,yes
sunny,mild,high,FALSE,no
sunny,cool,normal,FALSE,yes
rainy,mild,normal,FALSE,yes
sunny,mild,normal,TRUE,yes
overcast,mild,high,TRUE,yes
overcast,hot,normal,FALSE,yes
rainy,mild,high,TRUE,no

Excel => CSV



Software > Datasets



Some example datasets are included in the Weka distribution.

Available separately:

- A jarfile containing 37 classification problems, originally obtained from the **UCI repository** (**datasets-UCI.jar, 1,190,961 Bytes**).
- A jarfile containing 37 regression problems, obtained from various sources (**datasets-numeric.jar, 169,344 Bytes**).
- A jarfile containing 6 agricultural datasets obtained from agricultural researchers in New Zealand (**agridatasets.jar, 31,200 Bytes**).
- A jarfile containing 30 regression datasets collected by Luis Torgo (**regression-datasets.jar, 10,090,266 Bytes**).
- A gzip'ed tar containing **UCI** and **UCI KDD** datasets (**uci-20070111.tar.gz, 17,952,832 Bytes**)
- A gzip'ed tar containing **StatLib** datasets (**statlib-20050214.tar.gz, 12,785,582 Bytes**)
- A gzip'ed tar containing ordinal, real-world datasets donated by **Dr. Arie Ben David** (Holon Inst. of **Technology/Israel**) (**datasets-arie_ben_david.tar.gz, 11,348 Bytes**)
- A zip file containing 19 multi-class (1-of-n) text datasets donated by **George Forman/Hewlett-Packard Labs** (**19MclassTextWc.zip, 14,084,828 Bytes**)
- A bzip'ed tar file containing the Reuters21578 dataset split into separate files according to the ModApte split (**reuters21578-ModApte.tar.bz2, 81,745,032 Bytes**)
- A zip file containing 41 drug design datasets formed using the Adriana.Code software - **www.molecular-networks.com/software/adrianacode** - donated by **Dr. M. Fatih Amasyali** (Yildiz Technical University) (**Drug-datasets.zip, 11,376,153 Bytes**)
- A zip file containing 80 artificial datasets generated from the Friedman function donated by **Dr. M. Fatih Amasyali** (Yildiz Technical University) (**Friedman-datasets.zip, 5,802,204 Bytes**)

After expanding into a directory using your jar utility (or an archive program that handles tar-archives/zip files in case of the gzip'ed tars/zip files), these datasets may be used with Weka.

Other datasets in ARFF format:

- **Protein data sets**, maintained by **Shuiwang Ji**, CS Department, Louisiana State University/USA
- **Kent Ridge Biomedical Data Set Repository**, maintained by **Jinyan Li** and **Huiqing Liu**, Institute for Infocomm Research, Singapore
- **Repository for Epitope Datasets (RED)**, maintained by **Yasser El-Manzalawy**, Iowa State University.

WEKA Datasets

- Alguns datasets em formato **ARFF**

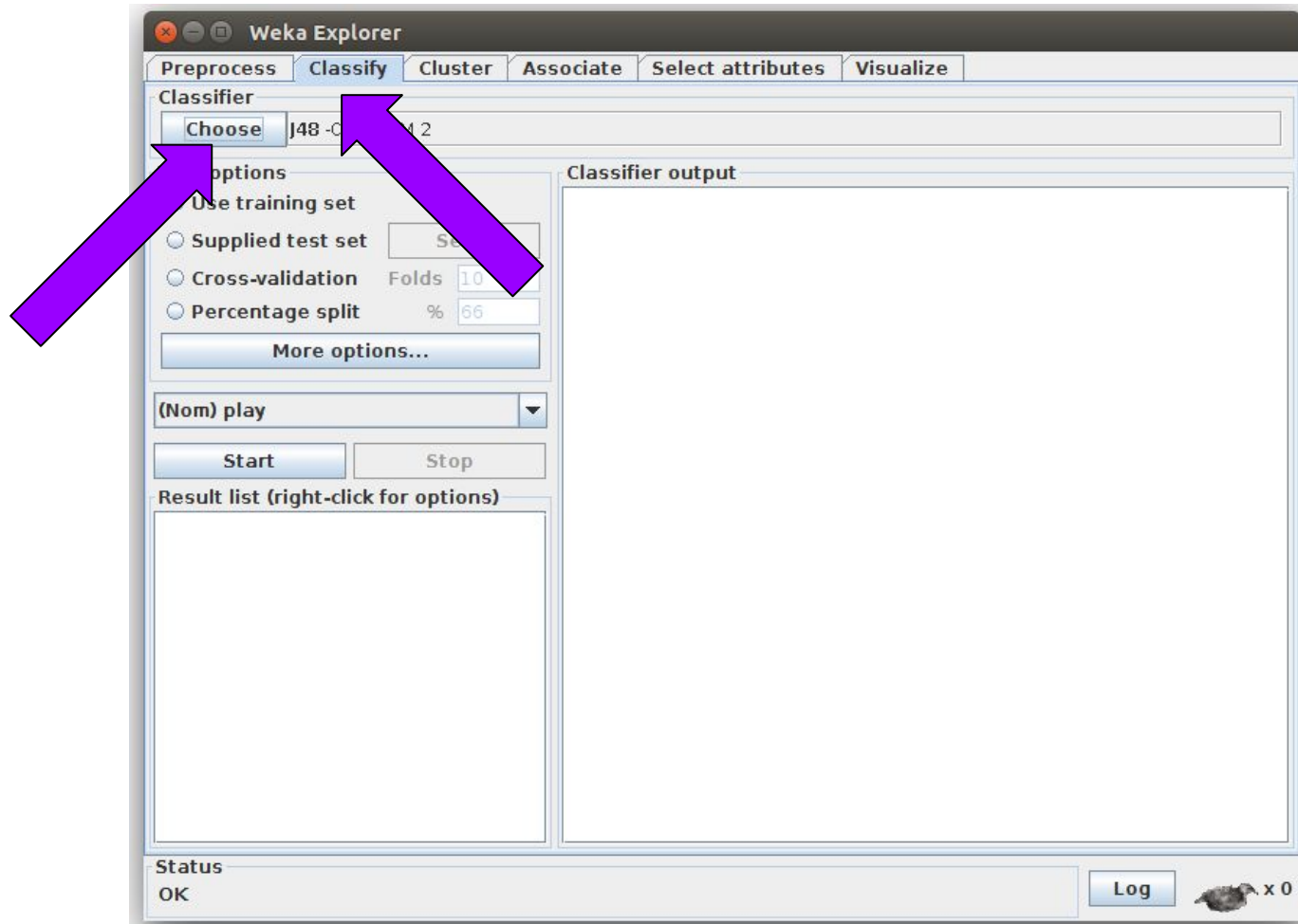
<http://storm.cis.fordham.edu/~gweiss/data-mining/datasets.html>

- [contact-lens.arff](#)
- [cpu.arff](#)
- [cpu.with-vendor.arff](#)
- [diabetes.arff](#)
- [glass.arff](#)
- [ionospehre.arff](#)
- [iris.arff](#)
- [labor.arff](#)
- [ReutersCorn-train.arff](#)
- [ReutersCorn-test.arff](#)
- [ReutersGrain-train.arff](#)
- [ReutersGrain-test.arff](#)
- [segment-challenge.arff](#)
- [segment-test.arff](#)
- [soybean.arff](#)
- [supermarket.arff](#)
- [vote.arff](#)
- [weather.arff](#)
- [weather.nominal.arff](#)

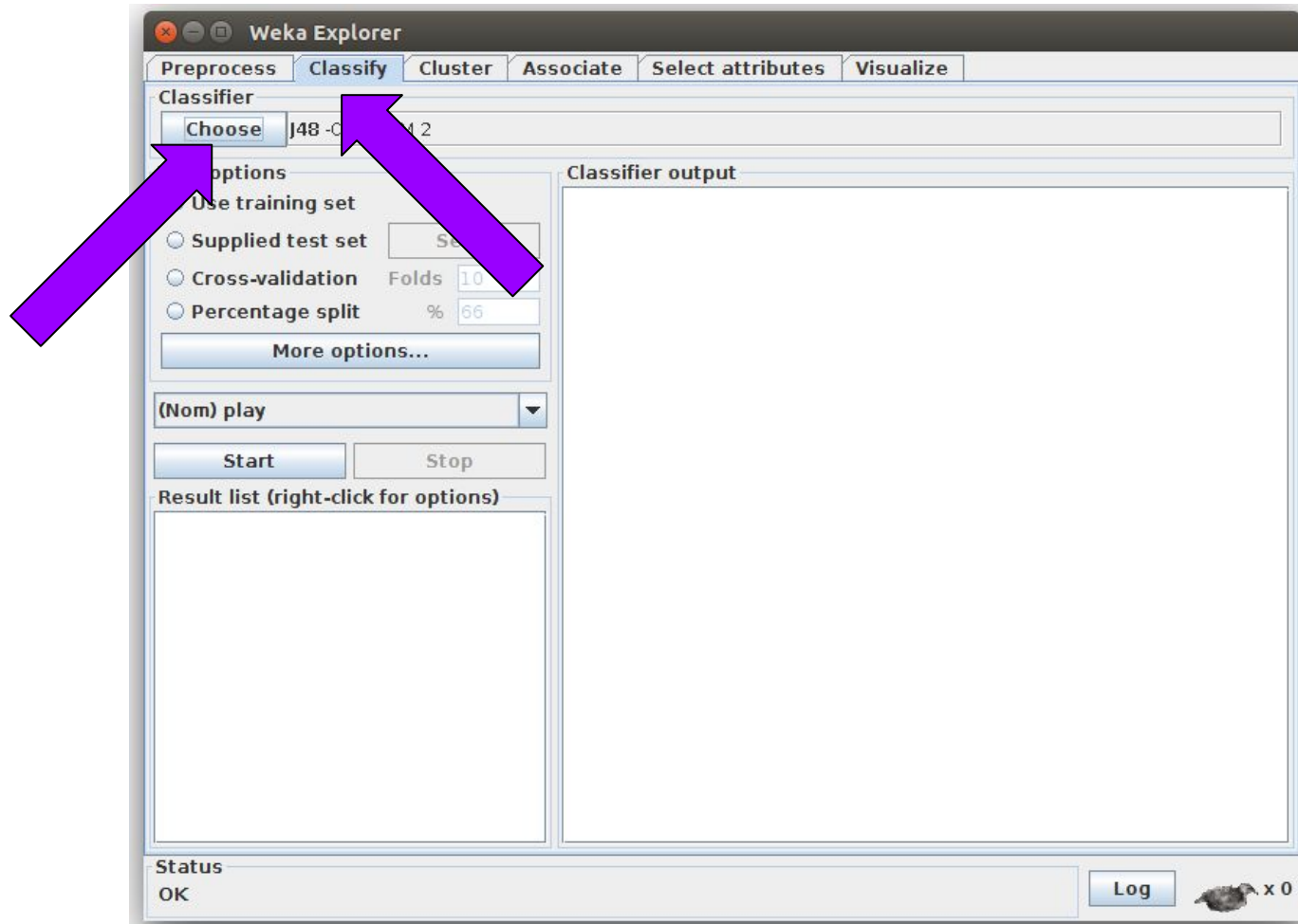
Classificação

- Como gerar:
 - uma árvore de decisão J48
 - um k-NN
 - Naive Bayes classifier
 - MLP
 - SVM
 - PCA

Classify > Choose



Classify > tree > J48



Classify > tree > J48

The screenshot shows the Weka Explorer window with the 'Classify' tab selected. The classifier chosen is 'J48 -C 0.25 -M 2'. The 'Test options' section has 'Use training set' selected. The 'Start' button is highlighted. The 'Classifier output' area displays the following results:

Time taken to build model: 0.01 seconds

=== Evaluation on training set ===

=== Summary ===

Metric	Value	Percentage
Correctly Classified Instances	14	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	14	

=== Detailed Accuracy By Class ===

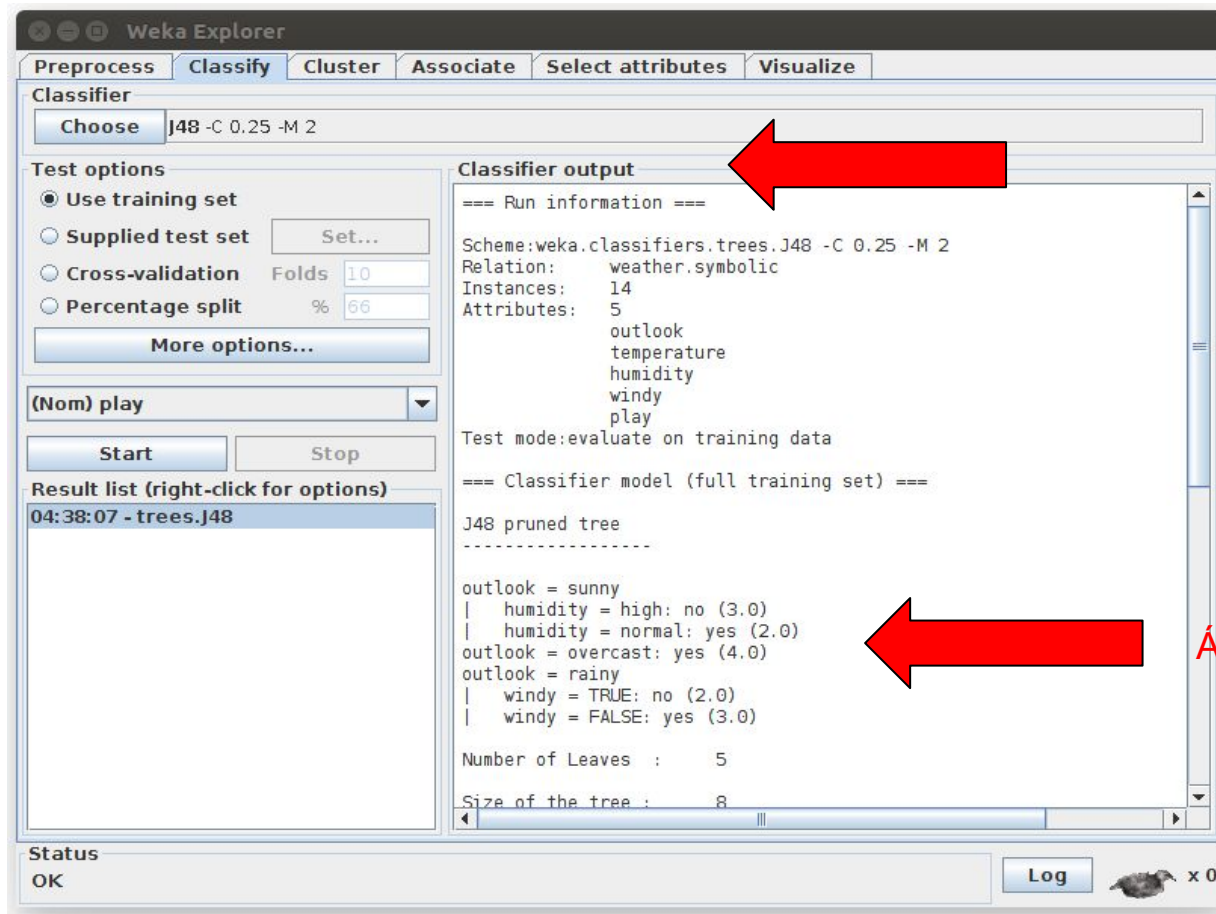
	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC
1	1	0	1	1	1	
1	1	0	1	1	1	
Weighted Avg.	1	0	1	1	1	

=== Confusion Matrix ===

```
a b <-- classified as
9 0 | a = yes
0 5 | b = no
```

Two purple arrows point to the 'Test options' and 'Start' buttons. A red arrow points to the 'Classifier output' area.

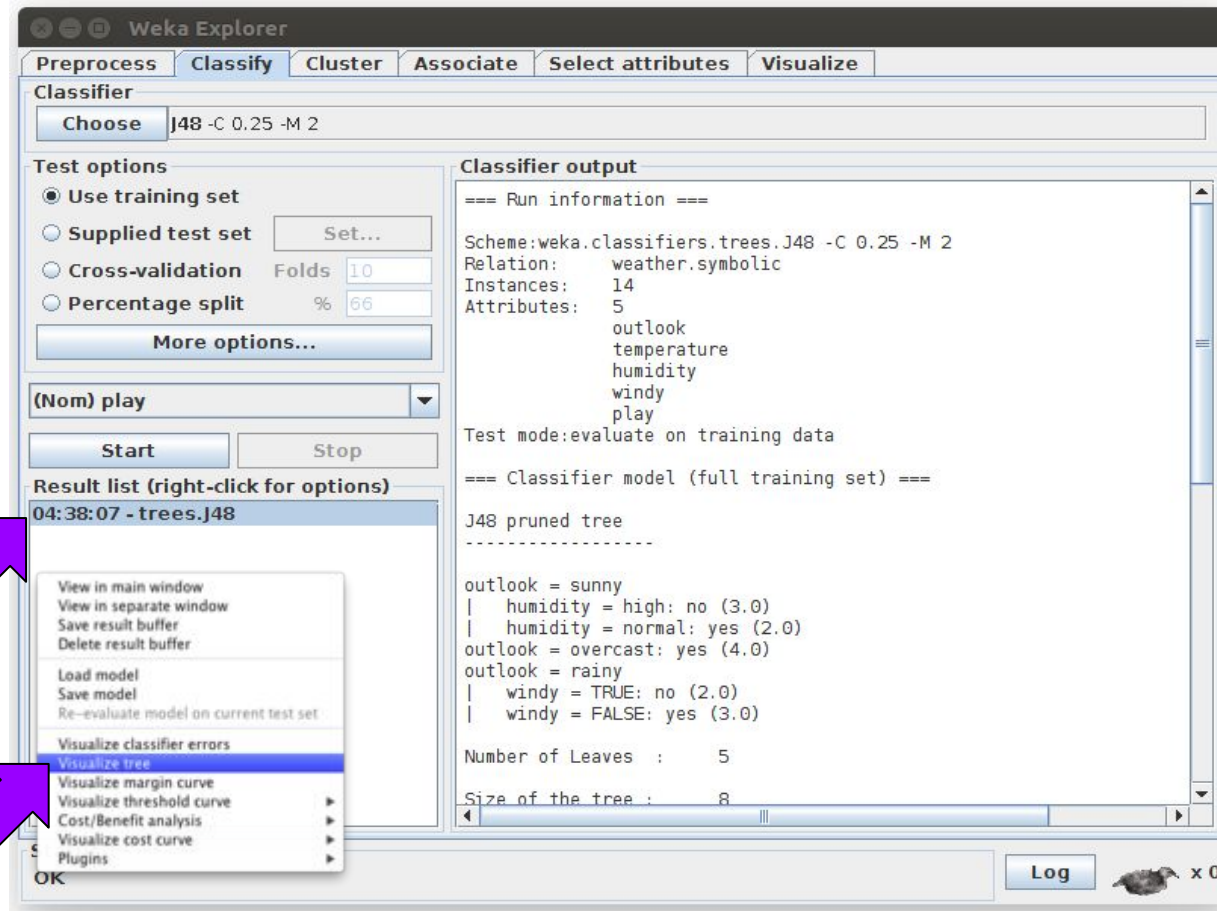
Classifier output



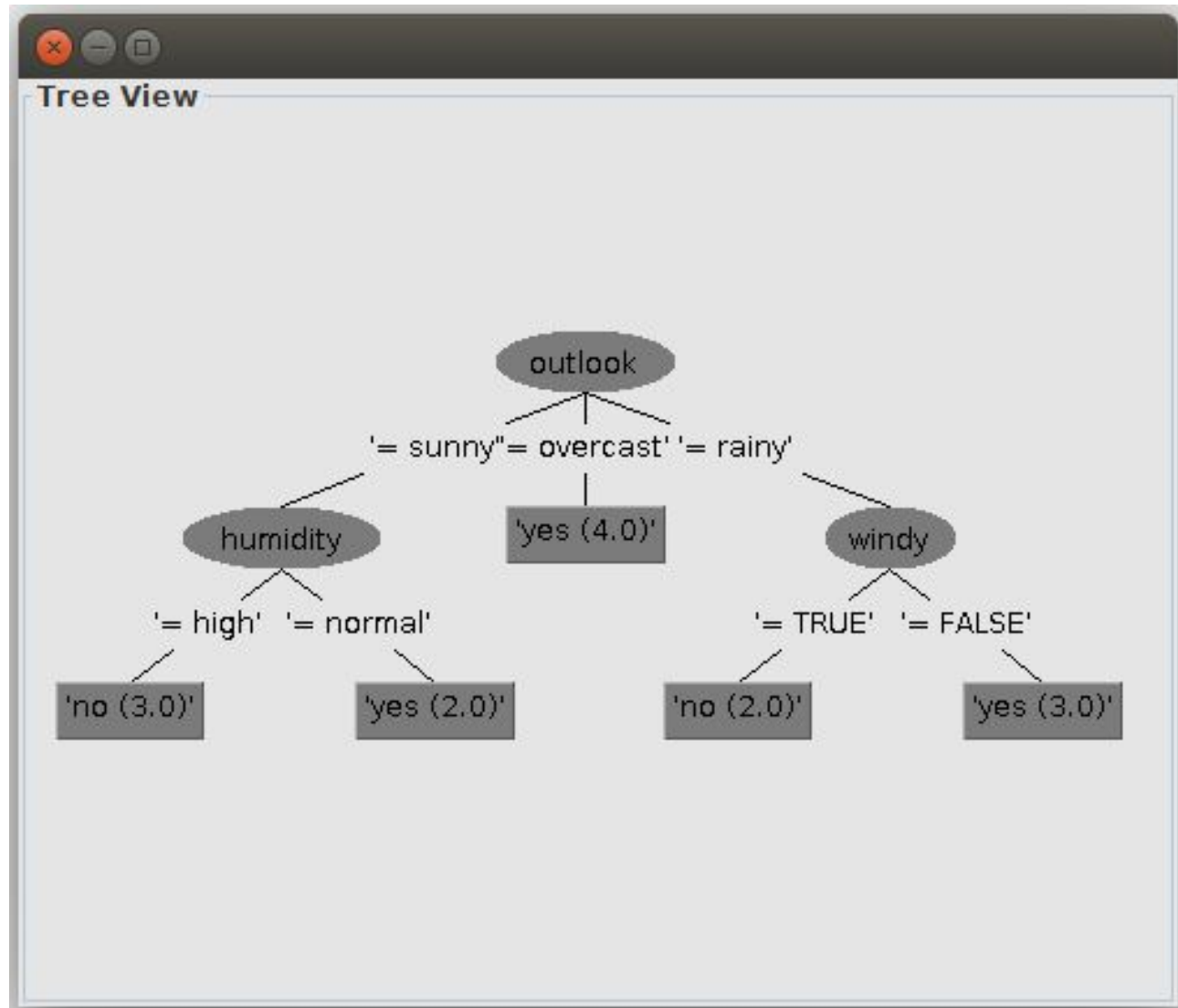
Árvore / Regras Geradas

Visualize

Clique com
botão direito



Visualize



Código em Java

```
import java.awt.BorderLayout;
import java.io.BufferedReader;
import java.io.FileReader;

import weka.classifiers.*;
import weka.classifiers.trees.J48;
import weka.core.Instances;
import weka.gui.treevisualizer.PlaceNode2;
import weka.gui.treevisualizer.TreeVisualizer;

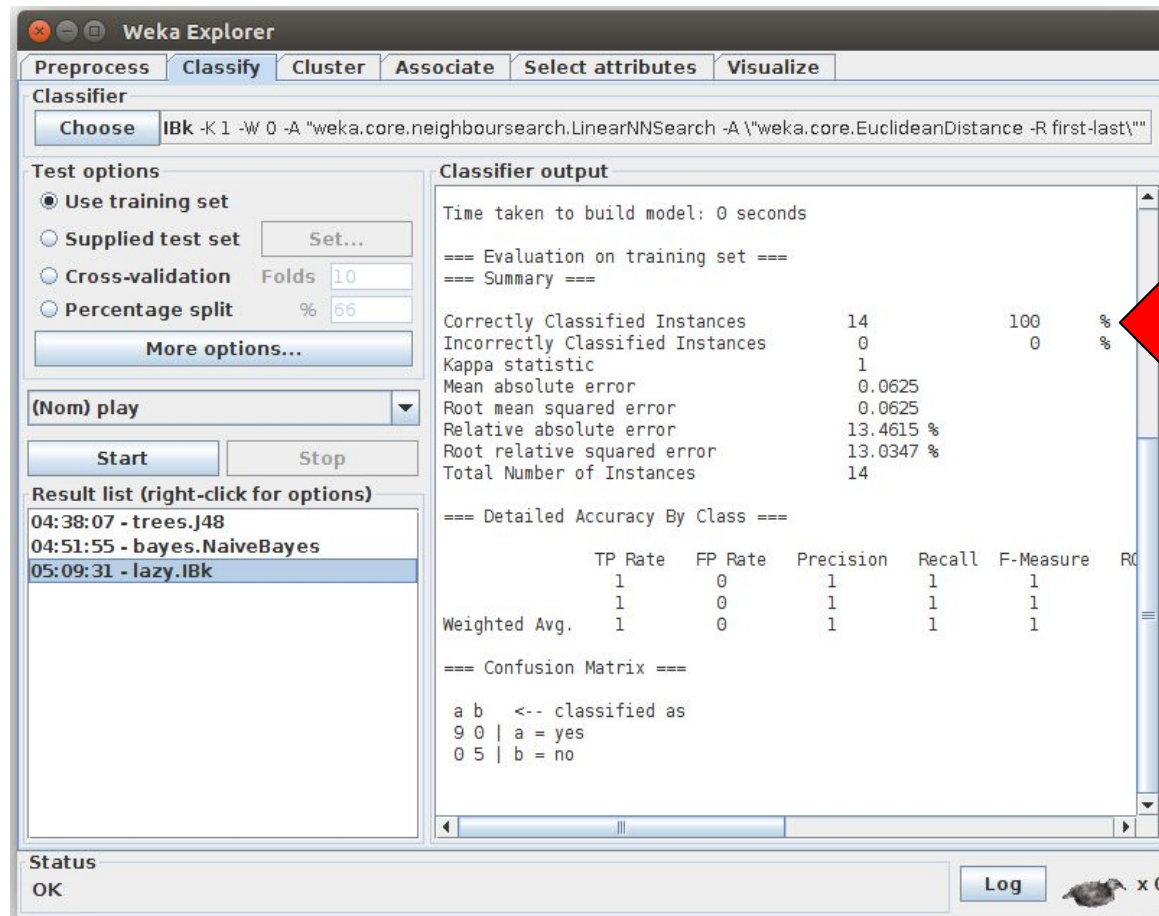
public class WekaJ48 {
    public static void main(String args[]) throws Exception {
        // train classifier
        J48 cls = new J48();
        Instances data = new Instances(new BufferedReader(new FileReader("D:\\sample.arff")));
        data.setClassIndex(data.numAttributes() - 1);
        cls.buildClassifier(data);
    }
}
```

Código em Java

```
// display classifier
final javax.swing.JFrame jf =
    new javax.swing.JFrame("Weka Classifier Tree Visualizer: J48");
jf.setSize(500,400);
jf.getContentPane().setLayout(new BorderLayout());
TreeVisualizer tv = new TreeVisualizer(null,
    cls.graph(),
    new PlaceNode2());
jf.getContentPane().add(tv, BorderLayout.CENTER);
jf.addWindowListener(new java.awt.event.WindowAdapter() {
    public void windowClosing(java.awt.event.WindowEvent e) {
        jf.dispose();
    }
});

jf.setVisible(true);
tv.fitToScreen();
}
```

Classify > Lazy > k-NN (IBk)



The screenshot shows the Weka Explorer application window. The 'Classify' tab is selected. The classifier is set to 'IBk' with the command: `IBk -K 1 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""`.

Test options:

- ☒ Use training set
- ☐ Supplied test set (Set...)
- ☐ Cross-validation (Folds: 10)
- ☐ Percentage split (%: 66)

More options...

(Nom) play

Start **Stop**

Result list (right-click for options)

- 04:38:07 - trees.J48
- 04:51:55 - bayes.NaiveBayes
- 05:09:31 - lazy.IBk

Classifier output:

Time taken to build model: 0 seconds

=== Evaluation on training set ===

=== Summary ===

Correctly Classified Instances	14	100	%
Incorrectly Classified Instances	0	0	%
Kappa statistic	1		
Mean absolute error	0.0625		
Root mean squared error	0.0625		
Relative absolute error	13.4615 %		
Root relative squared error	13.0347 %		
Total Number of Instances	14		

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC
1	1	0	1	1	1	
1	1	0	1	1	1	
Weighted Avg.	1	0	1	1	1	

=== Confusion Matrix ===

a b <-- classified as

9 0 | a = yes

0 5 | b = no

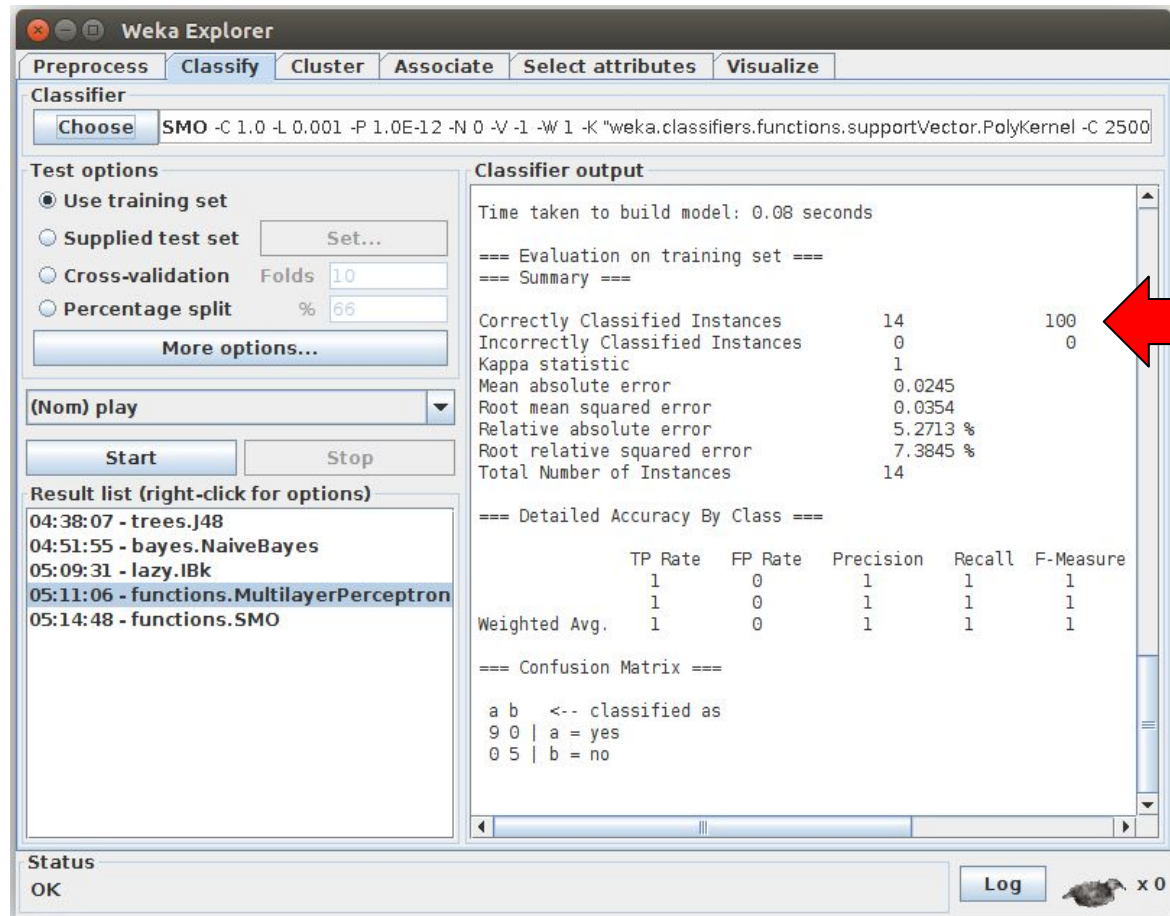
Status: OK

Log x 0

Classify > function > SVM

- Deve-se instalar o LibSVM
 - LIBSVM - A Library for Support Vector Machines
 - <https://www.csie.ntu.edu.tw/~cjlin/libsvm/>

Classify > function > SMO



Weka Explorer

Preprocess | **Classify** | Cluster | Associate | Select attributes | Visualize

Classifier

Choose SMO -C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V -1 -W 1 -K "weka.classifiers.functions.supportVector.PolyKernel -C 2500

Test options

☒ Use training set
☐ Supplied test set Set...
☐ Cross-validation Folds 10
☐ Percentage split % 66

More options...

(Nom) play

Start Stop

Result list (right-click for options)

- 04:38:07 - trees.J48
- 04:51:55 - bayes.NaiveBayes
- 05:09:31 - lazy.IBk
- 05:11:06 - functions.MultilayerPerceptron**
- 05:14:48 - functions.SMO

Classifier output

Time taken to build model: 0.08 seconds

=== Evaluation on training set ===

=== Summary ===

Correctly Classified Instances	14	100
Incorrectly Classified Instances	0	0
Kappa statistic	1	
Mean absolute error	0.0245	
Root mean squared error	0.0354	
Relative absolute error	5.2713 %	
Root relative squared error	7.3845 %	
Total Number of Instances	14	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure
1	1	0	1	1	1
1	1	0	1	1	1
Weighted Avg.	1	0	1	1	1

=== Confusion Matrix ===

a b <-- classified as

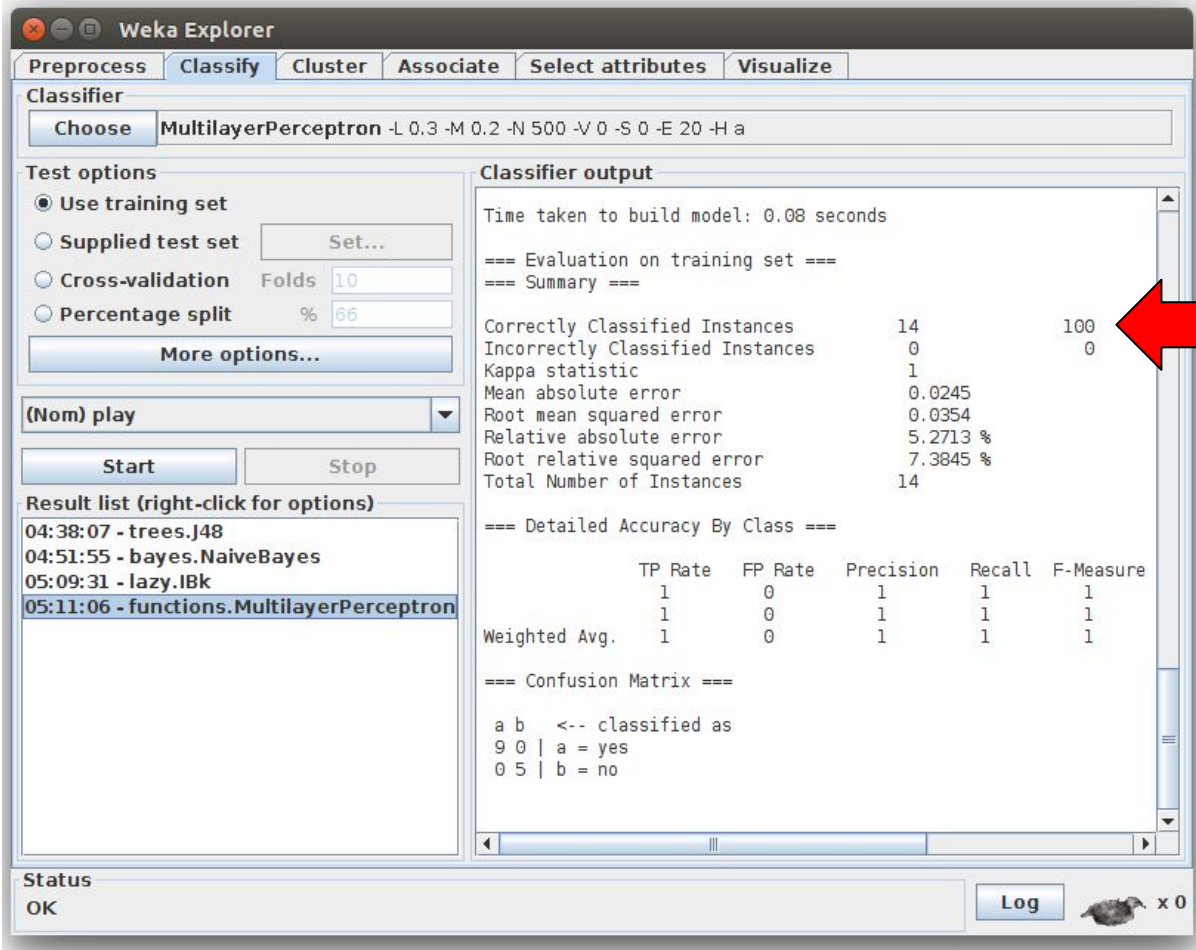
9 0 | a = yes

0 5 | b = no

Status OK

Log x 0

Classify > function > MLP



The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. The classifier chosen is 'MultilayerPerceptron' with parameters '-L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a'. The 'Test options' section shows 'Use training set' selected. The 'Classifier output' pane displays the following results:

Time taken to build model: 0.08 seconds

=== Evaluation on training set ===

=== Summary ===

Correctly Classified Instances	14	100
Incorrectly Classified Instances	0	0
Kappa statistic	1	
Mean absolute error	0.0245	
Root mean squared error	0.0354	
Relative absolute error	5.2713 %	
Root relative squared error	7.3845 %	
Total Number of Instances	14	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure
1	1	0	1	1	1
1	1	0	1	1	1
Weighted Avg.	1	0	1	1	1

=== Confusion Matrix ===

```
a b <-- classified as
9 0 | a = yes
0 5 | b = no
```

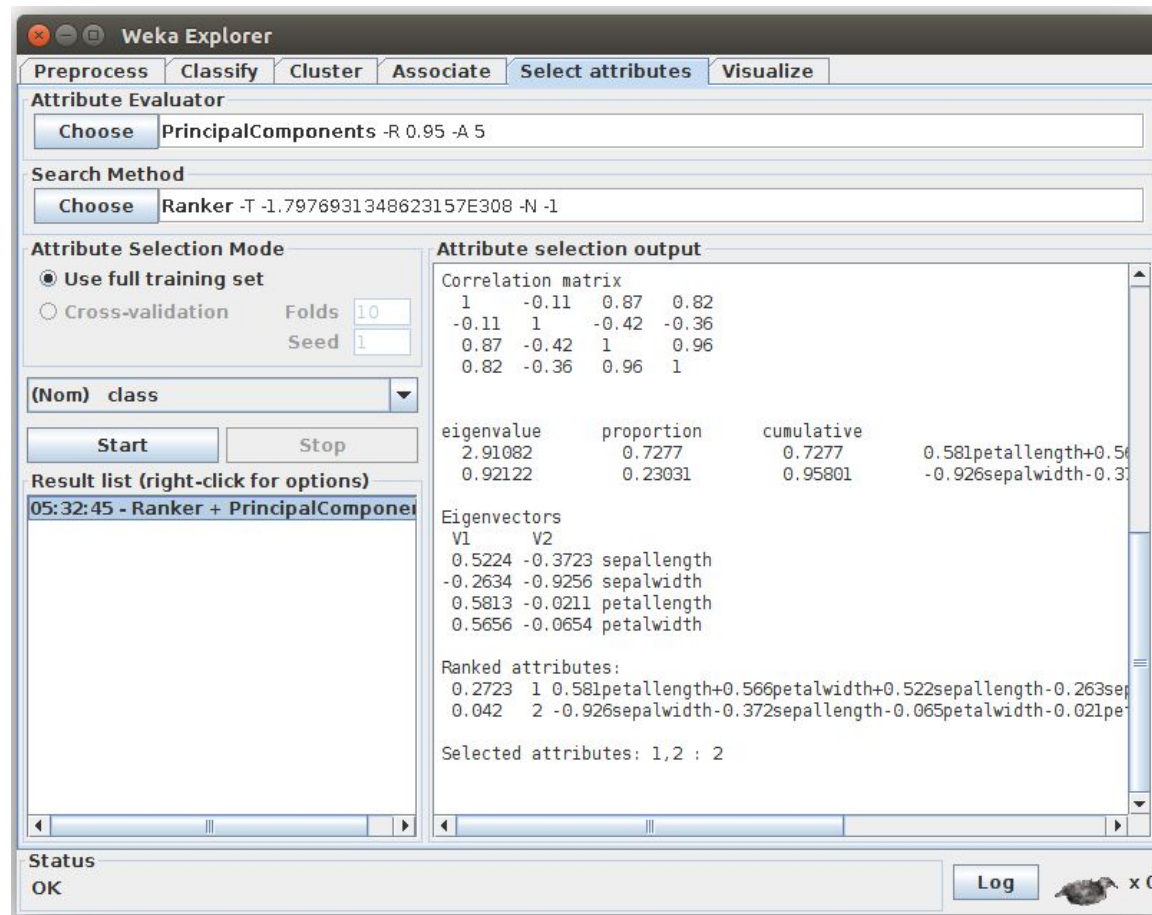
A red arrow points to the '100' value in the 'Summary' table, indicating the overall accuracy of the classifier.

The 'Result list' on the left shows the following entries:

- 04:38:07 - trees.J48
- 04:51:55 - bayes.NaiveBayes
- 05:09:31 - lazy.IBk
- 05:11:06 - functions.MultilayerPerceptron

The status bar at the bottom shows 'Status OK' and a 'Log' button.

Select Attributes > PCA



The screenshot shows the Weka Explorer interface with the 'Select attributes' tab selected. The 'Attribute Evaluator' is set to 'PrincipalComponents -R 0.95 -A 5'. The 'Search Method' is 'Ranker -T -1.7976931348623157E308 -N -1'. The 'Attribute Selection Mode' is 'Use full training set' with 'Folds' set to 10 and 'Seed' set to 1. The 'Result list' shows '05:32:45 - Ranker + PrincipalComponent'. The 'Attribute selection output' pane displays the following information:

Correlation matrix

1	-0.11	0.87	0.82
-0.11	1	-0.42	-0.36
0.87	-0.42	1	0.96
0.82	-0.36	0.96	1

eigenvalue **proportion** **cumulative**

2.91082	0.7277	0.7277	0.581petallength+0.566petalwidth+0.522sepallength-0.263sepalwidth
0.92122	0.23031	0.95801	-0.926sepalwidth-0.372sepallength-0.065petalwidth-0.021petallength

Eigenvectors

V1	V2
0.5224	-0.3723
-0.2634	-0.9256
0.5813	-0.0211
0.5656	-0.0654

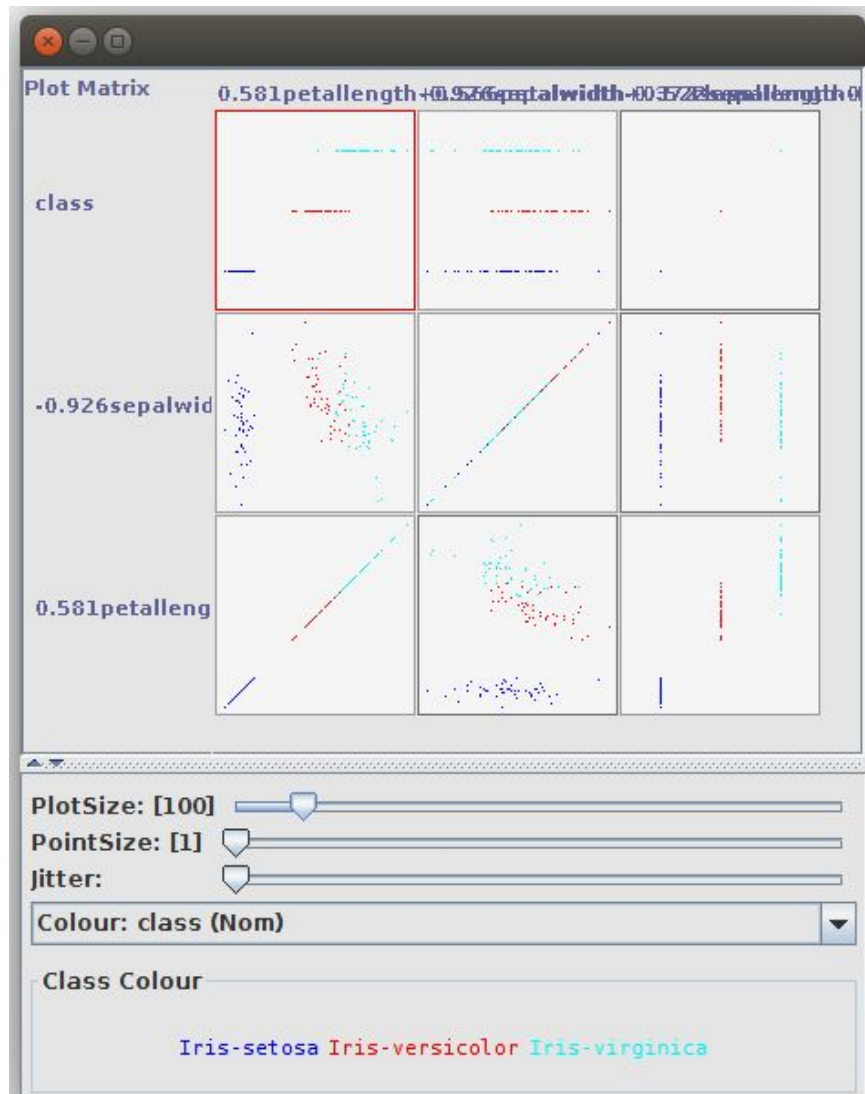
Ranked attributes:

0.2723	1	0.581petallength+0.566petalwidth+0.522sepallength-0.263sepalwidth
0.042	2	-0.926sepalwidth-0.372sepallength-0.065petalwidth-0.021petallength

Selected attributes: 1,2 : 2

The status bar at the bottom shows 'Status OK' and a 'Log' button.

Visualizar



Clustering & Regressão

- Como gerar:
 - Um kMeans
 - Uma regressão linear

Cluster > SimpleKmeans

The screenshot shows the Weka Explorer window with the 'Cluster' tab selected. The 'Clusterer' dropdown is set to 'SimpleKMeans' with parameters: -N 3 -A "weka.core.EuclideanDistance -R first-last" -I 500 -S 10. The 'Cluster mode' section has 'Use training set' selected. The 'Clusterer output' pane displays the following information:

Number of iterations: 3
Within cluster sum of squared errors: 7.817456892309574
Missing values globally replaced with mean/mode

Cluster centroids:

Attribute	Full Data (150)	Cluster# 0 (50)	1 (50)	2 (50)
sepalwidth	5.8433	5.936	5.006	6.588
sepalwidth	3.054	2.77	3.418	2.974
petalwidth	3.7587	4.26	1.464	5.552
petalwidth	1.1987	1.326	0.244	2.026
class		Iris-setosa Iris-versicolor	Iris-setosa	Iris-virginica

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

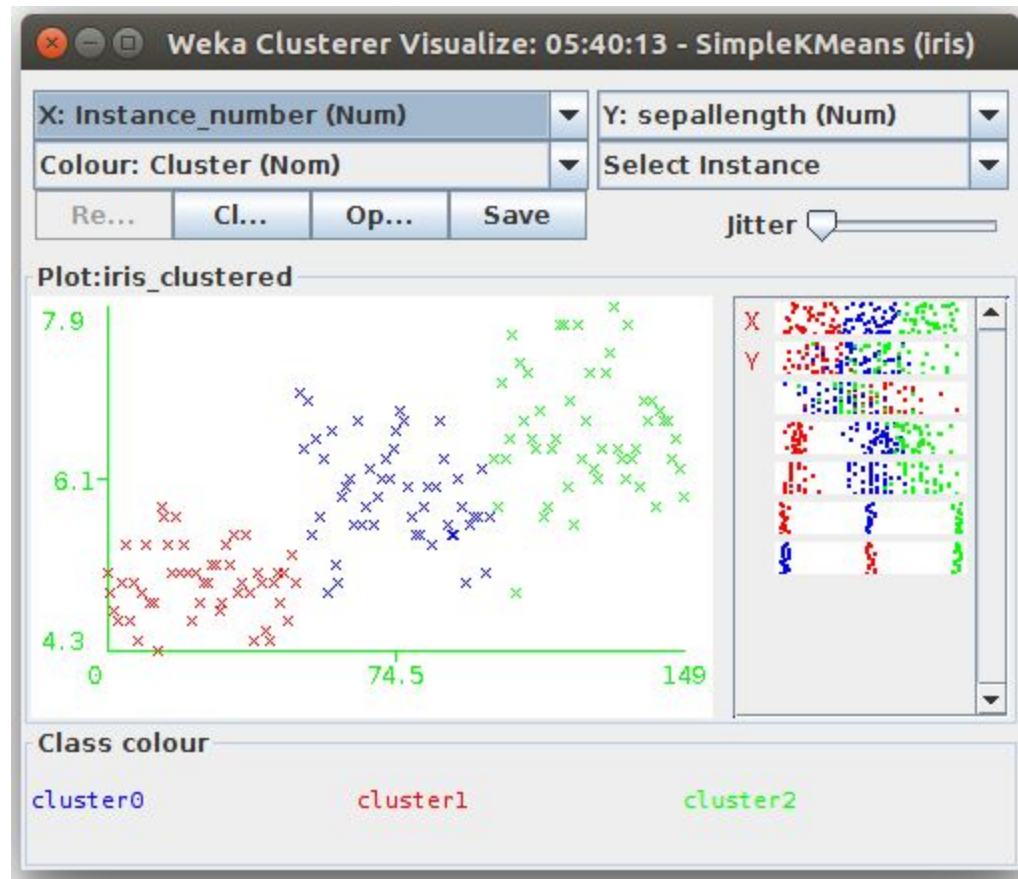
=== Model and evaluation on training set ===

Clustered Instances

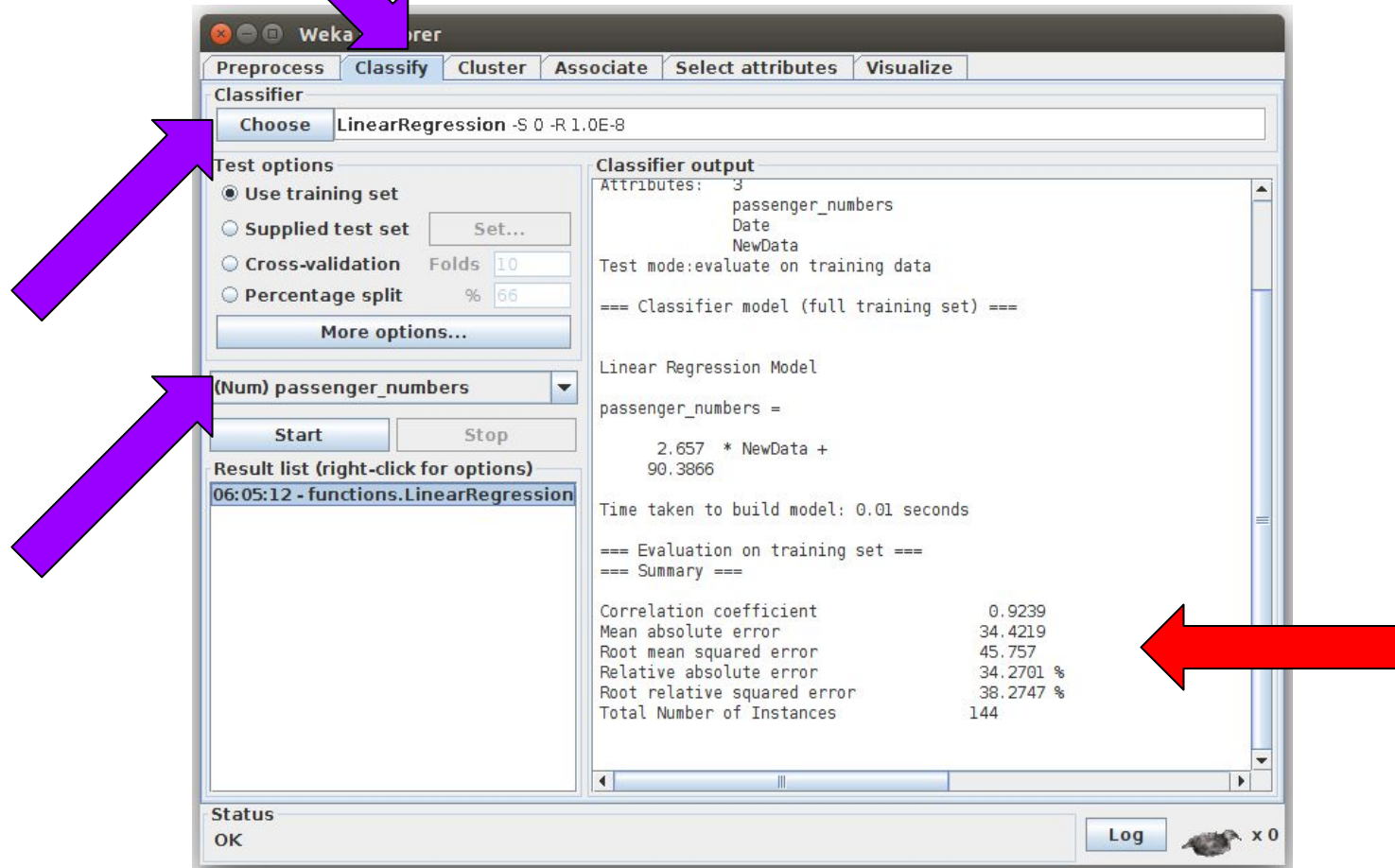
0	50 (33%)
1	50 (33%)
2	50 (33%)

The interface also includes a 'Result list' on the left showing '05:40:13 - SimpleKMeans' and a 'Status' bar at the bottom indicating 'OK'.

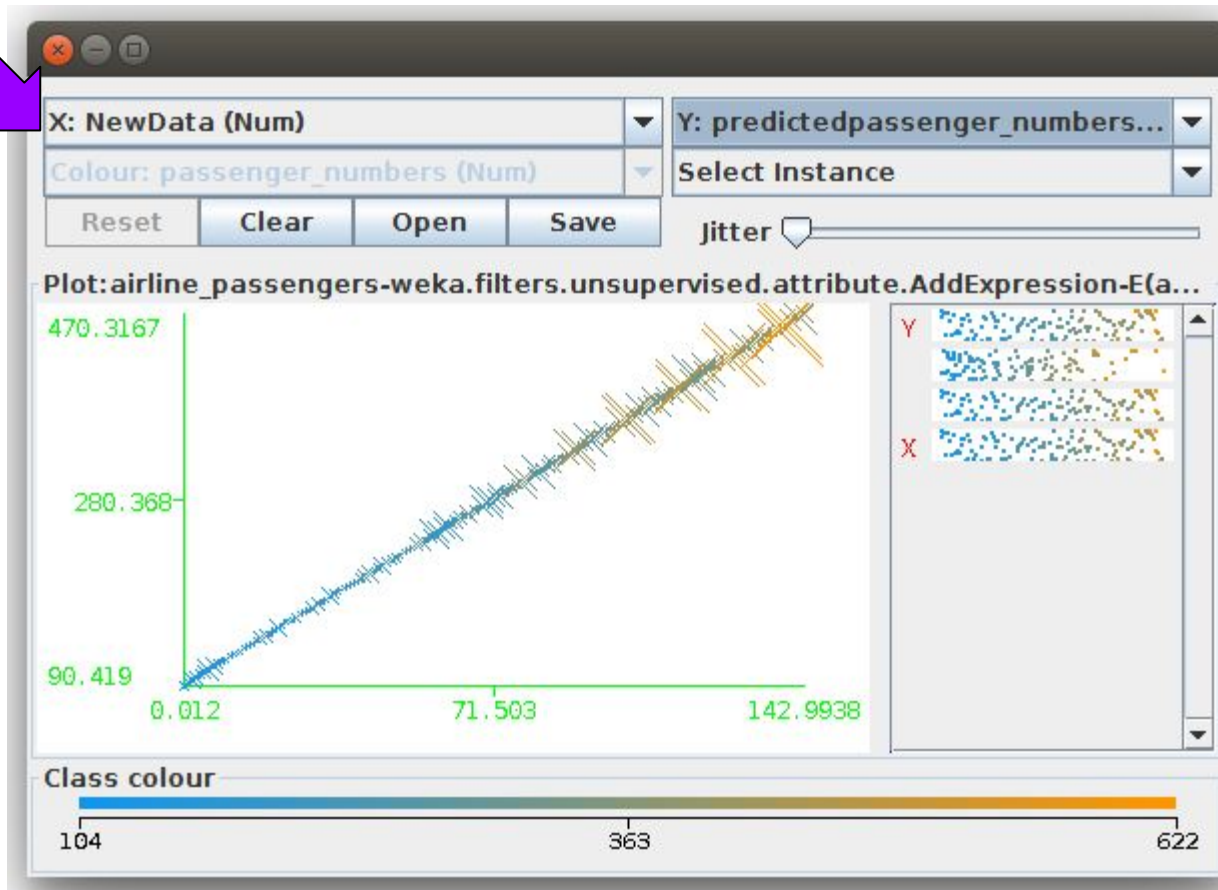
Cluster > SimpleKmeans



Classify > LinearRegression



Classify > LinearRegression





References

- Weka 3: Data Mining Software in Java
 - <https://www.cs.waikato.ac.nz/ml/weka/>
- Weka Datasets
 - <http://storm.cis.fordham.edu/~gweiss/data-mining/datasets.html>