CSC 458 Data Mining and Predictive Analytics I, Spring 2021

Dr. Dale E. Parson, Assignment 2, Using Weka rules and trees to correlate several stream-sampling attributes with dissolved oxygen in milligrams per liter. Due by 11:59 PM on Saturday March 13 via D2L. There will be a 10% per-day late penalty after that, and I cannot award points after I go over my solution during the following class period.

Perform the following steps to set up for this semester's projects and to get my handout. Start out in your login directory on csit (a.k.a. acad).

```
cd $HOME
mkdir DataMine # This should already be there from assignment 1.
cd ./DataMine
cp ~parson/DataMine/csc458trees2sp2021.problem.zip csc458trees2sp2021.problem.zip
unzip csc458trees2sp2021.problem.zip
cd ./csc458trees2sp2021
```

TO GET THE FILES FROM ACAD TO YOUR Mac OR Linux machine use this command line:

```
scp YOURLOGIN@acad.kutztown.edu://home/kutztown.edu/parson/DataMine/csc458trees2sp2021.problem.zip csc458trees2sp2021.problem.zip
```

You can use the reverse command line to copy files from your machine into your acad account.

WINDOWS USERS should log into https://download.kutztown.edu/ , log in, go to Computer Science, and download WinSCP. It appears you can just go to http://winscp.net/eng/download.php To get it.

EDIT THE SUPPLIED FILE README.txt when the following questions starting at Q1 below. Keep with the supplied format, and do not turn in a Word or PDF or other file format. I will deduct 20% for other file formats, because with this many varying assignments being turned in, I need a way to grade these in reasonable time, which for me is a batch edit run on the **vim** editor. Please turn in your final files README.txt and USGS_PA_STREAM_2012_NOMINAL.arff by the deadline using the D2L Assignment page 2.

NOTE ON A TYPO IN THE README (March 8):

Q5: Does your answers to Q2 & Q3 confirm, refute, or neither, information in the PART II linked readings? Explain why.

SHOULD SAY:

Q5: Does your answers to Q3 & Q4 confirm, refute, or neither, information in the PART II linked readings? Explain why.

Running Weka

The best way to run Weka during the pandemic is to load it onto your PC or laptop, downloading the latest stable version (currently 3.8.5) from here.
https://www.cs.waikato.ac.nz/ml/weka/
I will prepare the assignments by clicking on the Weka icon, not by running the command line to expand Weka's memory. This way I will try to ensure that you will not run out of memory using the default amount.

If you do your work on campus Windows PCs, make sure to save your work on a USB drive that you remember to take with you. Campus PCs discard any file changes when you log out. Campus PC users can run S:\ComputerScience\WEKA\WekaWith4GBcampus, which contains this batch command:

```
java -Xmx4096M -jar "S:\ComputerScience\WEKA\weka.jar"
```

Starting Weka from that command line allows you to increase its memory allotment, to 4GB in this case. Here is my command-line command on my Mac:

```
java -server -Xmx4000M -jar /Applications/weka-3-8-5/weka.jar
```

For assignment 2 the default memory size for Weka should be sufficient.

Open your site-specific ARFF file via Weka's **Preprocess** tab to investigate the following attributes. The time-related attributes derive from datetime.

agency_cd	USGS (US Geological Survey) These data are automated water samples from Pennsylvania streams In 2012 obtained as a text file from https://waterdata.usgs.gov/nwis, Water Quality, Historical Observations, then run through my Python data extraction script.
site_no	The USGS site number for the sampling station.
site_name	The USGS name for the site.
datetime	When the sample was taken.
tz_cd	Time zone.
OxygenMgPerLiter	Dissolved oxygen in milligrams per liter will be our target attribute.
pH	Base / acidity pH scale. Low numbers are acidic, with 7 being neutral.
TempCelsius	Water temperature in centigrade.
Conductance	Electrical conductance in microsiemens per centimeter at 25°C.
GageHt	Stream gage height in feet.
DischargeRate	Flow discharge rate in cubic feet per second.
TimeOfYear	Nominal value for one of the seasons, derived from datetime.
TimeOfDay	Nominal value for one time of day of sample, derived from datetime.
OxygenClass	Nominal range for OxygenMgPerLiter, with the ranges coming from A PA Dept. of Environmental web site. Below is my Python function That shows the mapping from numeric levels to nominal values.
month	Month as a number 2 through 12; there were no January measures.
MinuteOfDay	Number of minutes since the preceding midnight for this sample.
MinuteFromMidnite	Number of minutes to the closest midnight for this sample.
MinuteOfYear	Sampling time in minutes since the previous start of the year.
MinuteFromNewYear	Sampling time in minutes to the closest start of the year.

Here is the Python 3.x function for deriving OxygenClass from OxygenMgPerLiter.

```
def oxygen2Class(paramMap):
   if (not 'OxygenMgPerLiter' in paramMap.keys()):
      return None
   level = paramMap['OxygenMgPerLiter']
```

```
if level == None:
    return None
level = float(level)
result = 'Unsafe'
if level >= 6.0:
    result = 'NorthernPikeExcellent'
elif level >= 5.5:
    result = 'BlackBassGood'
elif level >= 4.2:
    result = 'CommonSunfishMedium'
elif level >= 3.3:
    result = 'BlackBullheadLow'
return str(result)
```

PART I – Preparing the data. 22% for the correct saved ARFF file.

1. Open ARFF file **USGS_PA_STREAM_2012.arff** in Weka and observe that the attribute names and types in your dataset match those on the previous page; bring the **Edit** Preprocessor Window up, full screen, and scroll around inspecting for missing values that are grayed out in this Editor. You can click on a heading such as datetime to sort the instances on that attribute. Shift-click on a heading gives a descending sort. Close the Edit window when you are done.
2. Run Weka's **unsupervised -> attribute -> StringToNominal** filter to turn strings into sets of values that you can read in Preprocess. After selecting this filter, click its command line display and set the **attributeRange** to **first-last**, then click the **Apply** button in the upper right of the Preprocess window. The **attributeRange** also accepts numeric attribute ranges separated by a "-", and individual attribute numbers separated by a ",", using the attribute numbers in the lower left of the Preprocess window. Some filters require you to be more precise with the **attributeRange** ranges, but StringToNominal converts only strings to sets of symbols, leaving non-string attributes unchanged. I usually save this step until I am ready to analyze a dataset, since adding new instances later on may add strings not in the current nominal set.

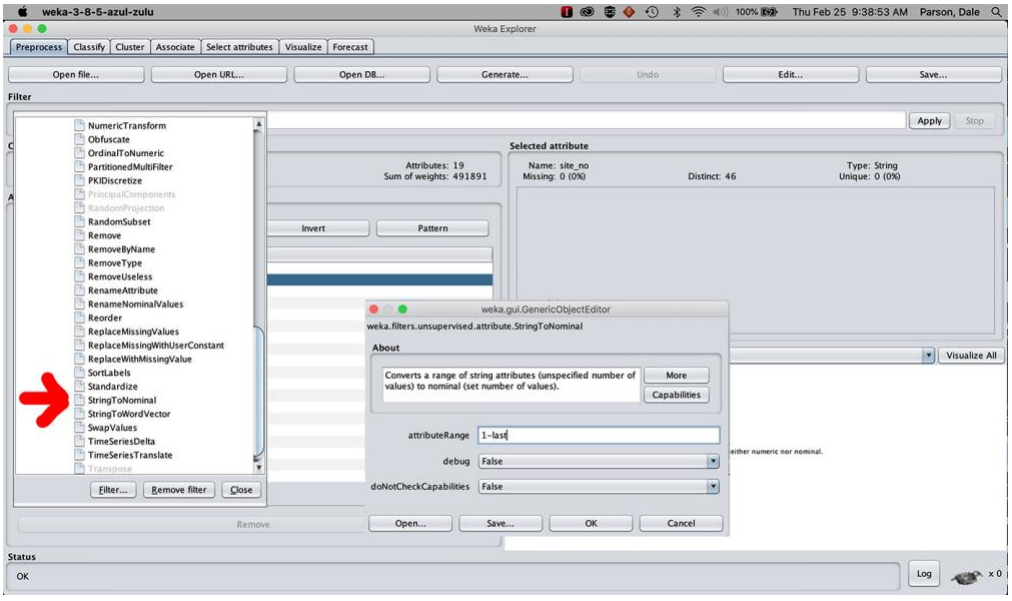


Figure 1: StringToNominal filter

3. Run Weka's **unsupervised -> attribute -> RemoveUseless** filter.

   Q1 in README.txt: Which attributes did RemoveUseless remove, and why? Read the pop-up RemoveUseless documentation in Weka, and execute Undo in the Weka preprocessor if you need to inspect the pre-RemoveUseless attribute values. Make sure to re-run RemoveUseless if you execute Undo.

4. Click the Visualize tab in Preprocess and pop up a plot of OxygenMgPerLiter on the Y axis as a function of MinuteOfYear on the X axis. It will look like Figure 2 at the top of the next page. The four highest peaks all come from the same site (site_no & site_name), and the two lowest troughs at the lower right come from the same site. These sites provide outlying data that we want to eliminate from our training set and test set of data.

   Q2 in README.txt: What are the **site_no** and **site_name** values for these two outlying sites?

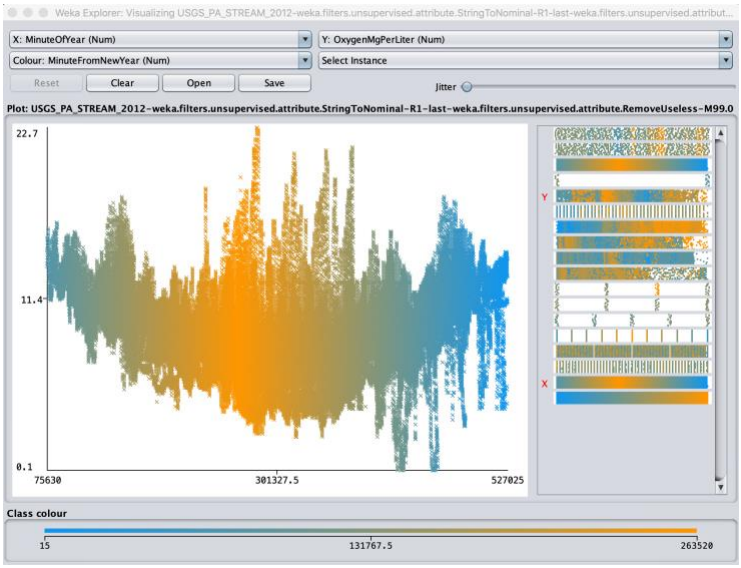


Figure 2: OxygenMgPerLiter on the Y axis as a function of MinuteOfYear on the X axis

5. In the Preprocess tab use the **unsupervised -> instance -> RemoveWithValues** Filter to remove all instances coming from these two sites. We do not want to train our models on these outliers, and we will skip testing them for now. In the RemoveWithValues parameter pop-up you can use the **attributeIndex** of either site_no or site_name (its number to te left of the attribute name) and you can use the two **nominalIndices** values for those sites by scrolling through the **Selected attribute** scrolling region. After clicking **Apply** for **RemoveWithValues** to remove those sites, very that they are gone using Visualize as in Figure 2 and verifying that those peaks and troughs are gone. There will be new peaks and troughs that do not repeat for the same site.

6. In the Preprocess tab delete attributes so that only these remain:

site_no	The USGS site number for the sampling station.
OxygenMgPerLiter	Dissolved oxygen in milligrams per liter will be our target attribute.
pH	Base / acidity pH scale. Low numbers are acidic, with 7 being neutral.
TempCelsius	Water temperature in centigrade.
Conductance	Electrical conductance in microsiemens per centimeter at 25ºC.
DischargeRate	Flow discharge rate in cubic feet per second.
MinuteOfDay	Number of minutes since the preceding midnight for this sample.
MinuteOfYear	Sampling time in minutes since the previous start of the year.

We are deleting datetime, tz_cd, and the more coarse-grain temporal attributes because they are redundant with the remaining temporal attributes, which are numeric and therefore more useful to Weka. We delete OxygenClass because it is redundant with OxygenMgPerLiter. We are deleting GageHt because it has many missing values for some sites. We will delete site information because it is trivial for some machine learning algorithms to simply memorize site+time -to- OxygenMgPerLiter mappings without analyzing underlying patterns in physical attribute correlations. First we must partition the instances into training data and testing data using a Python script that I am supplying.

7. Reorder attributes using the **Unsupervised -> Attribute -> Reorder** filter so that OxygenMgPerLiter moves to the bottom while all others stay the same. Weka expects the target attribute (a.k.a. *class*) being predicted to be the last one; Weka forces you to explicitly set the target attribute if you do not perform this reordering.

site_no	The USGS site number for the sampling station.
pH	Base / acidity pH scale. Low numbers are acidic, with 7 being neutral.
TempCelsius	Water temperature in centigrade.
Conductance	Electrical conductance in microsiemens per centimeter at 25ºC.
DischargeRate	Flow discharge rate in cubic feet per second.
MinuteOfDay	Number of minutes since the preceding midnight for this sample.
MinuteOfYear	Sampling time in minutes since the previous start of the year.
OxygenMgPerLiter	Dissolved oxygen in milligrams per liter will be our target attribute.

8. Partition OxygenMgPerLiter into 10 bins using the **Unsupervised -> Attribute -> Discretize** filter. You will have to set config parameter **ignoreClass** to **True**, since this attribute is the class (i.e., the target attribute). Set **useEqualFrequency** to **False** to main the roughly normal, bell-shaped distribution of OxygenMgPerLiter in the 10 bins. **Apply** the filter.

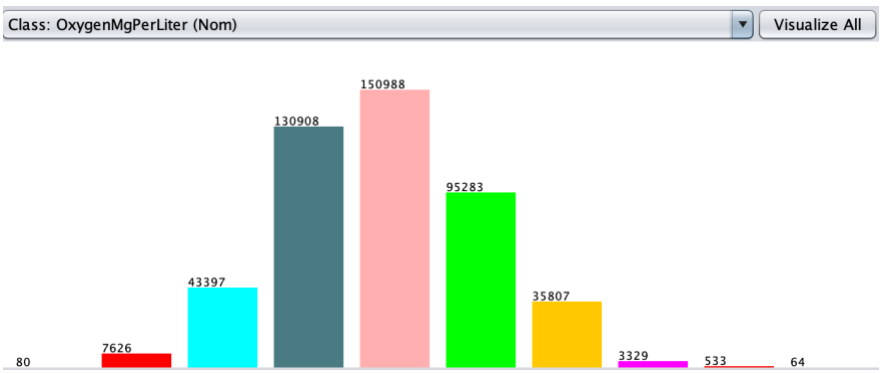


Figure 3: Discretized OxygenMgPerLiter with useEqualFrequency = False

SAVE THIS DATA SET AS file USGS_PA_STREAM_2012_NOMINAL.arff. It must contain all of these attributes & instances. You will perform other temporary attribute deletions, but you will not save those reduced datasets. You must turn in this ARFF file via D2L along with your completed **README.txt** file when you are ready.

Next run my supplied script in the directory containing this ARFF file like this:

python splitTrainTest.py

where python is python3.x. If you copy the above ARFF file back onto acad or mcgonagall, you can run this:

/usr/local/bin/python3.7 splitTrainTest.py

That will partition **USGS_PA_STREAM_2012_NOMINAL.arff** into two files:

USGS_PA_STREAM_2012_TRAIN.arff contains training data from 4 training sites determined to have a good cross-section of representative data for the entire dataset. There are 49066 instances.

USGS_PA_STREAM_2012_TEST.arff contains additional testing data from the remaining sites. There are 418949 instances.

PART II – Analyzing the data. 6.5% for each of Q1 through Q12.

Exit & restart Weka and **load USGS_PA_STREAM_2012_TRAIN.arff into Weka. Remove the site_no attribute for all learning.** We do not want Weka to memorize site+time -to- OxygenMgPerLiter mappings as previously discussed.

LINKED READINGS. Read over this essay. It was the best match for the Google query "dissolved oxygen in water".
http://www.fondriest.com/environmental-measurements/parameters/water-quality/dissolved-oxygen/
Here is another, in case that site is down when you go to it. Try to at least skim them both.
https://water.usgs.gov/edu/dissolvedoxygen.html

Also, review the Kappa statistic so you can interpret its significance. For any analysis question in Q3-Q13 for which you paste the Kappa statistic into README.txt, make sure to interpret the Kappa statistic in your analysis.
"Landis and Koch considers 0-0.20 as slight, 0.21-0.40 as fair, 0.41-0.60 as moderate, 0.61-0.80 as substantial, and 0.81-1 as almost perfect. Fleiss considers kappas > 0.75 as excellent, 0.40-0.75 as fair to good, and < 0.40 as poor. It is important to note that both scales are somewhat arbitrary." Read the rest of my Kappa page yourself. We will go over Kappa statistic in class.

EDIT THE SUPPLIED FILE README.txt and answer the following questions. Please read these instructions closely. I will deduct points for missing requirements.

Q3: Run the rule **OneR** classifier. What are the "Correctly Classified Instances" as a percentage correct, and the "Kappa statistic", for class attribute OxygenMgPerLiter?

Q4: Copy & paste OneR's rule into README.txt. (Use mouse sweep and control-C on Windows, command-C on Mac)What attribute does OneR use to predict OxygenMgPerLiter, and what pattern can you see, if any in the mappings of this rule? If there are a lot of IF rules, just look at a contiguous block of them at a time. There is a pattern.

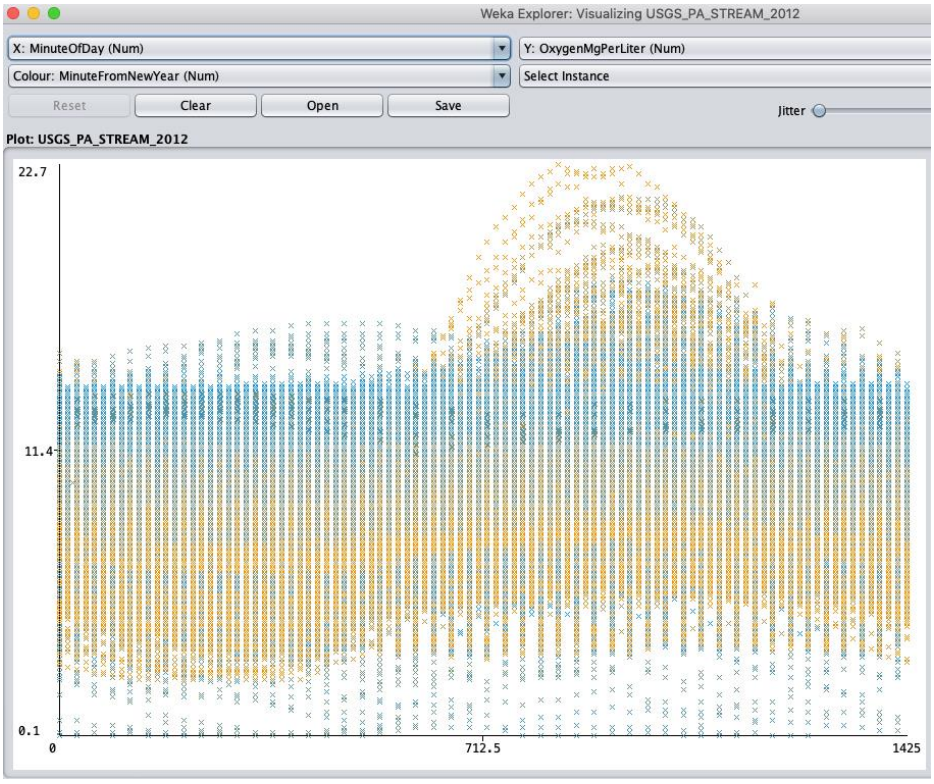
Q5. Does your answer to Q3 and Q4 confirm, refute, or neither, information in the PART II linked readings? Explain why.

Q6. Does running the **J48** classification tree on these attributes add any accuracy to OneR's results on the same data? Explain your answer. Copy & paste "Correctly Classified %, and the "Kappa statistic" into README.txt.

Q7: After removing attributes so that only **MinuteOfYear**, **MinuteOfDay** and **OxygenMgPerLiter** remain, run rule **OneR**. What are the "Correctly Classified Instances %" and "Kappa statistic" values?

Q8: Use the Visualize tab to inspect **OxygenMgPerLiter** on the Y axis as a function of **MinuteOfDay** on the X axis. Can you see a pattern of value changes in the afternoon that indicate photosynthesis per your Part II linked readings? Why or why not?

UPDATE March 8: Reading the 2D visualizations after Discretize on OxygenMgPerLiter is difficult. Here is the visualization of the numeric OxygenMgPerLiter value. This is easier to see:



Q9. Repeat Q7 using the J48 tree. Do you see any change in "Correctly Classified Instances %" and "Kappa statistic" values? Show these values in README.txt.

Load your saved file USGS_PA_STREAM_2012_TRAIN.arff **and remove site_no** to get back to a full set of attributes.

Apply **OneR**, **J48**, and **RandomTree** to your file's dataset to predict **OxygenMgPerLiter**.

Q10. What are their respective "Correctly Classified Instances" and "Kappa statistic" values? Which is the most predictive?

Q11. Which of Q12 is the easiest to understand in terms of looking at the rule or tree structure? Why?

In the Classify tab click the **Supplied test set** radio button and set it to USGS_PA_STREAM_2012_TEST.arff. We are changing from cross-validation on the training data to validation against USGS_PA_STREAM_2012_TEST.arff test data, which contains some non-representative sampling sites. A decrease in kappa accuracy indicates some degree of over-fitting the learned models to the training data.

You may get a warning because the site_no attribute in USGS_PA_STREAM_2012_TEST.arff is missing from USGS_PA_STREAM_2012_TRAIN.arff from which your models are learned. Just click through the warning and proceed with testing. Weka will show this attribute mapping:

Attribute mappings:

Model attributes	Incoming attributes
(numeric) pH	--> 2 (numeric) pH
(numeric) TempCelsius	--> 3 (numeric) TempCelsius
(numeric) Conductance	--> 4 (numeric) Conductance
(numeric) DischargeRate	--> 5 (numeric) DischargeRate
(numeric) MinuteOfDay	--> 6 (numeric) MinuteOfDay
(numeric) MinuteOfYear	--> 7 (numeric) MinuteOfYear
(nominal) OxygenMgPerLiter	--> 8 (nominal) OxygenMgPerLiter

Q12. Apply **OneR**, **J48**, and **RandomTree** to predict **OxygenMgPerLiter**. What are their respective "Correctly Classified Instances" and "Kappa statistic" values? Did they increase, decrease, or stay about the same? Do you think that there was some over-fitting to the training data compared with cross-validation? Explain.

NOTE: Building the trees takes some time. You will see "building model for fold 1" through "fold 10" at the bottom left during its run. Weka uses a subset of the data set instances for training and the other instances for testing. It is important to separate training data from test data, so as not to pollute the tests with over-fitting. In this case Weka is using *ten-fold cross-validation*. It randomly picks 10 equal-size, distinct partitions (folds) of the instances. It uses 9 for training and 1 for testing, then swaps the 1 testing fold into the training set and pulls an unused fold for testing an learns again, and so on, until each fold has appeared as the test data set once, and in the training data 9 times. Weka can maximize the generality of its learned structures on moderate data set sizes this way. It is also possible to use distinct training and testing files. We may do so later in the semester.

When you have completed all of your work and double-checked the assignment requirements, make sure that both **USGS_PA_STREAM_2012_NOMINAL.arff** saved at the end of Part I, and your **README.txt** that answers Q1 through Q12, turned in to the D2L Assignment 2 page Late assignments lose 10% per day late, and I will not accept an assignment after I go over its solution in class.

Filters
Attribute Unsupervised
Reorder (to make the target the last attribute at the bottom)
StringToNominal (to turn a limited number of strings into a usable set of symbols, for example, we turned "d" "r" "l" "p" for lsTOarff files into nominal set {d, r, l, p} which means directory, regular-file, symbolic-link, or named-pipe. If you added more file types by getting more data (e.g., "b" for block IO, and "c" for character IO device files), you would have to extend the arff file nominals to include {d, r, l, p, b, c}.
In addition to StringToNominal, just remove the date fields.
If we had time intervals, we could compute a numeric time interval (onedate - otherdate), in terms of hours, days, etc. in Python.
Discretize (to turn numeric attributes into nominal BINS of values, e.g., like turning numeric grades into A, Aminus, etc.)
useEqualFrequency of FALSE distributes bins across numeric range of the attribute.
useEqualFrequency of TRUE tries to distribute bins across equals sizes.
If the filter such as Discretize has no effect, try setting ignoreClass to TRUE and run it again. When ignoreClass is false, some of the filters try to correlate the attribute being filtered with the target attribute, also known as the class.
Instance Unsupervised filters
RemoveWithValues to eliminate two unrepresentative sites.
Rules
ZeroR just to see it. It just picks the most popular target value.
OneR maps the most predictive attribute to the target attribute.
We iterated and used OneR to pick the next most predictive attribute, removed it temporarily, and did this again, until this process became less predictive, or we got enough attributes.
This process is partially redundant with Weka's "Select attributes" Tab.
Trees
J48 -- top-down partition (splitting) of attribute-to-target mappings
Sometimes setting unpruned to TRUE increases accuracy at the cost of human intelligibility of the tree.
RandomTree -- bottom-up partition (splitting) of attribute-to-target mappings