Jack Burdick

Weka document classification

Task Information

Goal

Given a document collection, classify the information using a Naive Bayes and SVM approach

Setup/Dependencies

- Install Weka
  - http://www.cs.waikato.ac.nz/ml/weka/
- Python (for data preprocessing)
  - https://www.python.org/downloads/

Dataset

The dataset consists of two files(webkb-train-stemmed.txt and webkb-test-stemmed.txt) that look like this (sample image below);

```
faculty prof georg georg receiv degre electr engin univers california berkelei degre electr engin
univers california lo angel gordon marshal professor comput scienc director usc center manufactur
autom research director robot research laboratori univers southern california research interest area
intellig robot system applic robot medicin plan control manufactur system director usc robot
institut chairman comput scienc depart addit teach univers research profession experi includ year
research engin depart engin ucla work primarili comput year beckman instrument comput applic engin
manag lo angel comput center section head analysi simul section senior staff engin trw system lo
angel assign trw organ staff group concern simul control man space vehicl publish technic paper area
biomed engin robot comput simul control system human machin system author text hybrid comput wilei
editor book recent neural network robot kluwer editor autonom robot found editor ieee transact robot
autom member editori board mathemat comput simul transact societi comput simul profession societi
membership includ associ comput machineri acm american associ artifici intellig aaai societi comput
simul sc intern neural network societi inn fellow institut electr electron engin ieee fellow
american associ advanc scienc aaa member nation academi engin
student deepak master engin depart comput scienc cornel univers resum educ cours person deepak
cornel resum html postscript back main page educ undergradu complet undergradu june karnataka region
engin colleg india major comput scienc major interest multimedia cours relat comput scienc oper
system artifici intellig compil construct data commun comput graphic graduat present cornel univers
pursu master engin degre comput scienc multimedia graduat involv project deal multimedia web server
program back main page cours cours list fall semest multimedia system prof brian smith advanc
databas system prof praveen seshadri engin comput network prof srinivasan keshav softwar engin prof
michael godfrei back main page person start time long long ago novemb land call bharat india world
precis born cute babi approxim pound deepak mean light process chang world incident divin interfer
rai miss world born dai isn lucki born dai leav detail earlier life dive straight high school lucki
nation public school bangalor greater part school place colleg major comput scienc long year holidai
part conquer class joi match cornel univers pursu master degre comput scienc hope final link friend
ashish vineet back main page
```

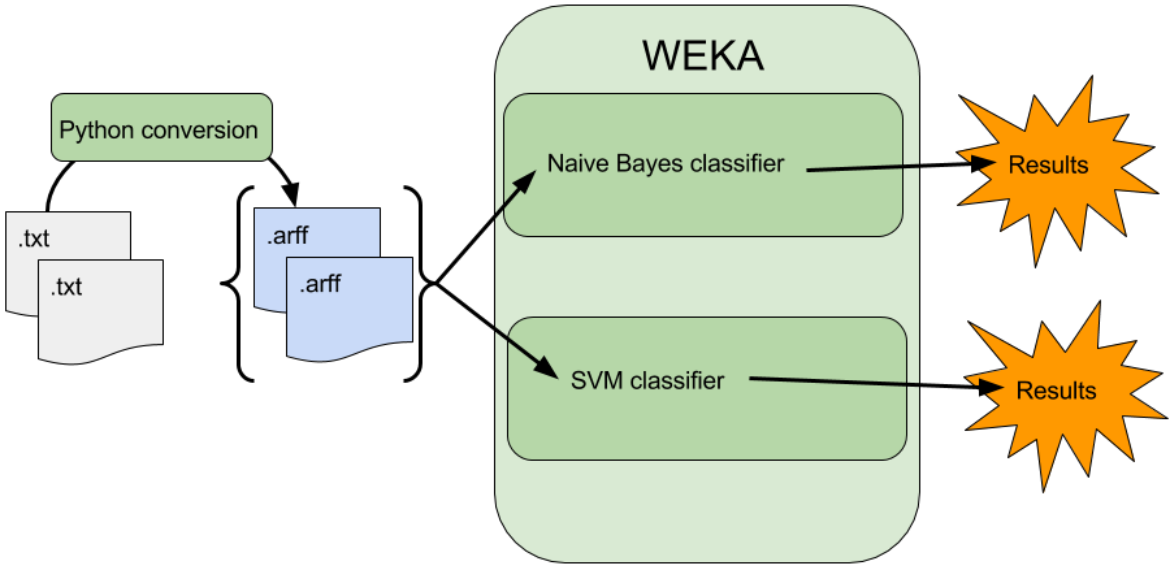
The dataset contains 2803 training sets and 1396 test sets -- where sets is equivalent to a label:data pair. The label could be of four categories [student, faculty, project, course]. It should be noted that the data is already pre-processed (stop words removed and stemming performed).

Dataset preprocessing (python conversion .txt → .arff)

Weka works well with .arff files. Though I'm sure weka is capable of handling and converting .txt files, I chose to convert the input .txt files into .arff format. I was not previously familiar with this format and found information here: http://www.cs.waikato.ac.nz/ml/weka/arff.html

Method

The dataset is received as .txt file. Weka *prefers* .arff files and a python script was created to convert the .txt files to .arff files. The .arff files are then accepted as input and classified with both a Naive Bayes and SVM classifier. The output, results, will then be interpreted.

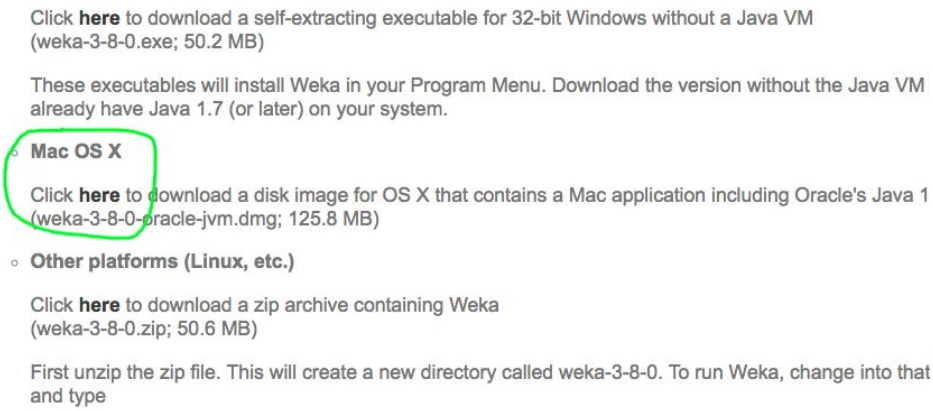


Tutorial

Download Weka and Install

There are many blogs about how to get started with weka -- this one was adequate http://machinelearningmastery.com/download-install-weka-machine-learning-workbench/

1. Visit weka download page:
    a. http://www.cs.waikato.ac.nz/ml/weka/downloading.html
2. Select your download (I'm using mac)

a. 
Click **here** to download a self-extracting executable for 32-bit Windows without a Java VM (weka-3-8-0.exe; 50.2 MB)

These executables will install Weka in your Program Menu. Download the version without the Java VM already have Java 1.7 (or later) on your system.

- **Mac OS X**

  Click **here** to download a disk image for OS X that contains a Mac application including Oracle's Java 1 (weka-3-8-0-oracle-jvm.dmg; 125.8 MB)

- **Other platforms (Linux, etc.)**

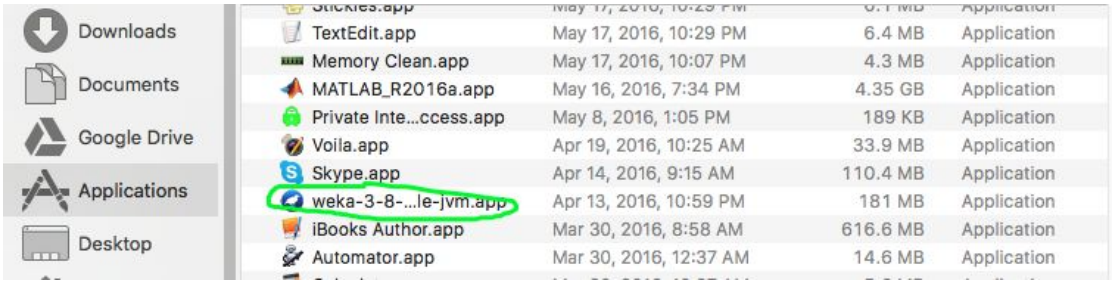
  Click **here** to download a zip archive containing Weka (weka-3-8-0.zip; 50.6 MB)

  First unzip the zip file. This will create a new directory called weka-3-8-0. To run Weka, change into that and type

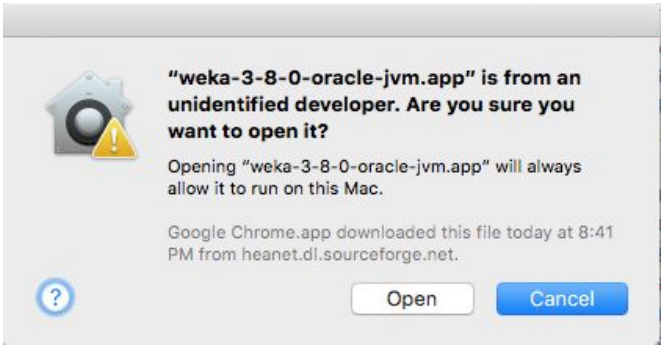
3. Open downloaded .dmg and drop .app into application folder

a. 

4. Run the .app from the application folder

a. 

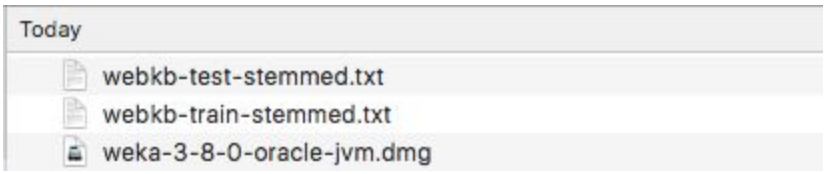
5. You must right-click and “open” (mac) since it is from an “unauthorized” source

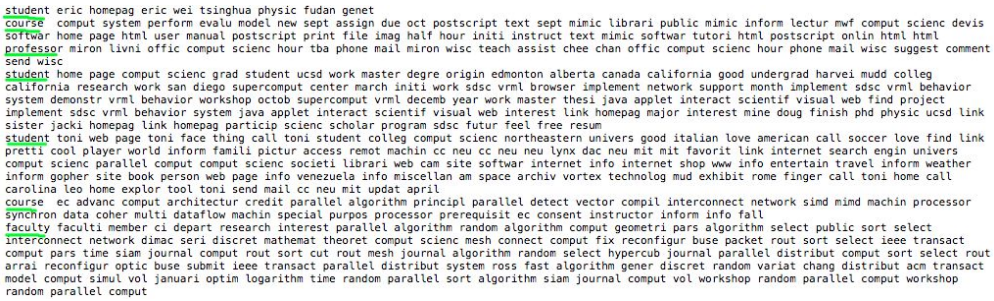
a. 

6. Now installed (optional, right click on the icon in the dock and select “keep in dock” if you want to keep it, ….any guesses?...., in your dock)
   a. 

7. [OK] - We’re up and running!

Dataset Collection and inspection

1. Gather dataset
   a. 

2. Inspection
   a. 
```
student eric homepag eric wei tsinghua physic fudan genet
course  comput system perform evalu model new sept assign due oct postscript text sept mimic librari public mimic inform lectur mwf comput scienc devis
softwar home page html user manual postscript print file imag half hour initi instruct text mimic softwar tutori html postscript onlin html html
professor miron livni offic comput scienc hour tba phone mail miron wisc teach assist chee chan offic comput scienc hour phone mail wisc suggest comment
send wisc
student home page comput scienc grad student ucsd work master degre origin edmonton alberta canada california good undergrad harvei mudd colleg
california research work san diego supercomput center march initi work sdsc vrml browser implement network support month implement sdsc vrml behavior
system demonstr vrml behavior workshop octob supercomput vrml decemb year work master thesi java applet interact scientif visual web find project
implement sdsc vrml behavior system java applet interact scientif visual web interest link homepag major interest mine doug finish phd physic ucsd link
sister jacki homepag link homepag particip scienc scholar program sdsc futur feel free resum
student toni web page toni face thing call toni student colleg comput scienc northeastern univers good italian love american call soccer love find link
pretti cool player world inform famili pictur access remot machin cc neu cc neu neu lynx dac neu mit mit favorit link internet search engin univers
comput scienc parallel comput comput scienc societi librari web cam site softwar internet info internet shop www info entertain travel inform weather
inform gopher site book person web page info venezuela info miscellan am space archiv vortex technolog mud exhibit rome finger call toni home call
carolina leo home explor tool toni send mail cc neu mit updat april
course  ec advanc comput architectur credit parallel algorithm principl parallel detect vector compil interconnect network simd mimd machin processor
synchron data coher multi dataflow machin special purpos processor prerequisit ec consent instructor inform info fall
faculty faculti member ci depart research interest parallel algorithm random algorithm comput geometri pars algorithm select public sort select
interconnect network dimac seri discret mathemat theoret comput scienc mesh connect comput fix reconfigur buse packet rout sort select ieee transact
comput pars time siam journal comput rout sort cut rout mesh journal algorithm random select hypercub journal parallel distribut comput sort select rout
arrai reconfigur optic buse submit ieee transact parallel distribut system ross fast algorithm gener discret random variat chang distribut acm transact
model comput simul vol januari optim logarithm time random parallel sort algorithm siam journal comput vol workshop random parallel comput workshop
random parallel comput
```
      i. Each “newline”{\n} separates a label:data pair. That pair then has the label at the first position and the data in the remaining positions for that line

What’s a(n) .arff file?

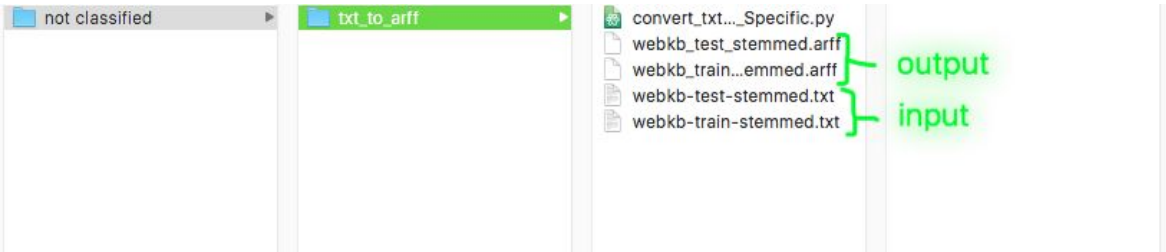
- [SPOILER ALERT] → this is where we’re headed

```
@relation 'WebKB Test'

@attribute Text string
@attribute class-att{student, faculty, project, course}

@data

'eric homepag eric wei tsinghua physic fudan genet',student
'comput system perform evalu model new sept assign due oct postscript text sept mimic librari public
mimic inform lectur mwf comput scienc devis softwar home page html user manual postscript print file
imag half hour initi instruct text mimic softwar tutori html postscript onlin html html professor
miron livni offic comput scienc hour tba phone mail miron wisc teach assist chee chan offic comput
scienc hour phone mail wisc suggest comment send wisc',course
'home page comput scienc grad student ucsd work master degre origin edmonton alberta canada california
good undergrad harvei mudd colleg california research work san diego supercomput center march initi
work sdsc vrml browser implement network support month implement sdsc vrml behavior system demonstr
vrml behavior workshop octob supercomput vrml decemb year work master thesi java applet interact
scientif visual web find project implement sdsc vrml behavior system java applet interact scientif
visual web interest link homepag major interest mine doug finish phd physic ucsd link sister jacki
homepag link homepag particip scienc scholar program sdsc futur feel free resum',student
'toni web page toni face thing call toni student colleg comput scienc northeastern univers good
```

- Header information is at the top -- (http://www.cs.waikato.ac.nz/ml/weka/arff.html), followed by a @data and the corresponding *data*

From .txt to .arff with python

- Folder structure



- Code explanation

- 

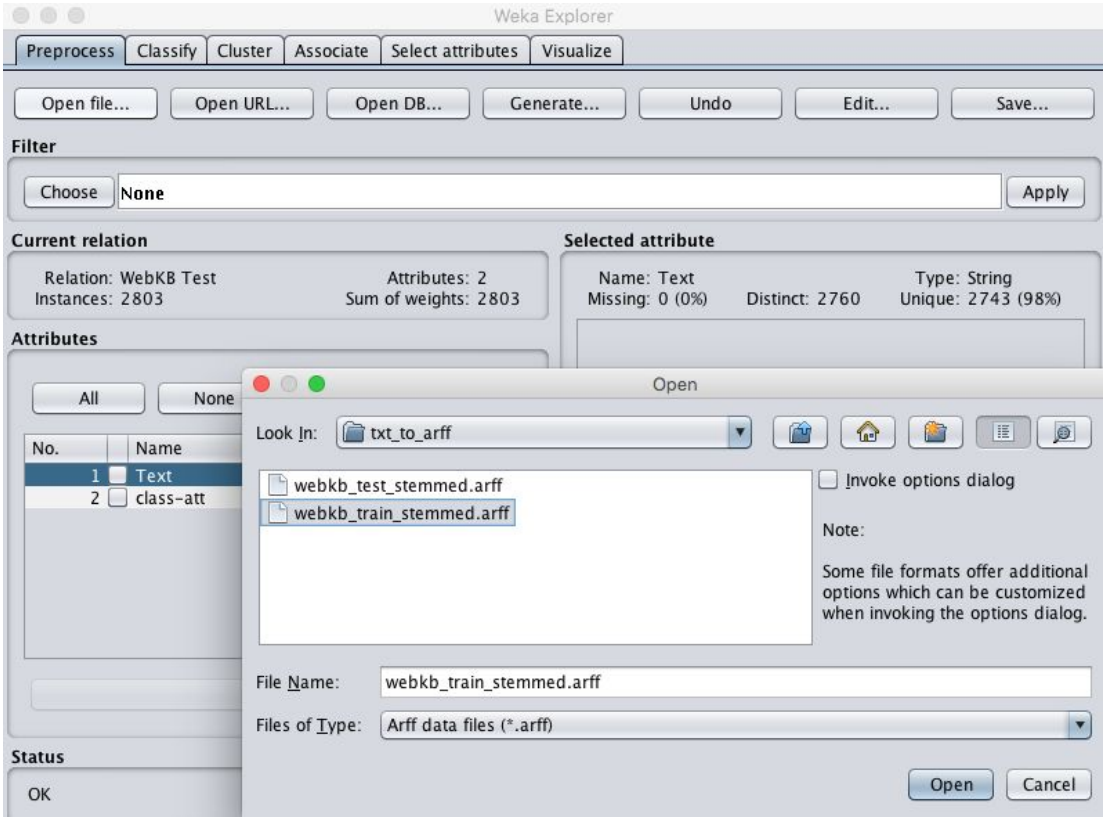

```python
#!/usr/bin/python3


# open file path (.txt), read information, return list of datasets
def read_in_file_return_data_list(file_input_path):
    data_line_list = []
    try:
        with open(file_input_path, 'r') as file_current:
            for data_line in file_current:
                data_line_list.append(data_line)
        return data_line_list
    except:
        error_message = "ERROR: input ( " + file_input_path + " ) not found"
        exit(error_message)


# create .arff file
def handle_data_write_arff(list_of_txt_entries, output_path):
    # will create file if not created
    with open(output_path, "w") as output_file:
        # write header information
        output_file.write("@relation 'WebKB Test'\n\n"
                          + "@attribute Text string\n"
                          + "@attribute class-att"
                          + "{student, faculty, project, course}\n\n"
                          + "@data\n\n")

        # write main data block
        for data_block in list_of_txt_entries:
            # split into classifier and data
            data_list = data_block.split('\t')  # .txt file is split by a "\t"
            class_name = data_list[0]
            data = data_list[1]
            data = data.rstrip('\n')   # remove trailing newline

            # write the main data
            output_file.write("'" + data + "'" + "," + class_name)
            output_file.write('\n')    # Move to newline to designate new entry


def convert_file(file_path, output_path):
    print("[START] Convert .txt to .arff")
    # read in data
    try:
        list_of_txt_entries = read_in_file_return_data_list(file_path)
    except:
        exit("ERROR: Unable to read input data - unsure what the error is")

    print("--update: ", len(list_of_txt_entries), "'datasets' were found")

    # convert and create .arff file
    try:
        handle_data_write_arff(list_of_txt_entries, output_path)
    except:
        exit("ERROR: Unable to create/write an .arff file - error uncertain")

    print("[END] Convert .txt to .arff: ", output_path)


def main():
    # convert_file(file_path, output_path)
    convert_file("./webkb-train-stemmed.txt", "webkb_train_stemmed.arff")
    convert_file("./webkb-test-stemmed.txt", "webkb_test_stemmed.arff")
    # NOTE: these will create this file path if it is not already created
    # Otherwise, if found, it will overwrite the previous information


if __name__ == "__main__":
    main()
```

Python output

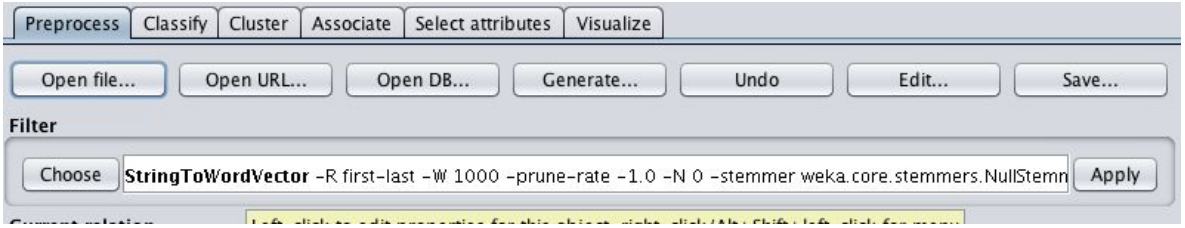
```
MrBurdick txt_to_arff $ python convert_txt_to_arff_IRHW2_Specific.py
[START] Convert .txt to .arff
--update:  2803 'datasets' were found
[END] Convert .txt to .arff:  webkb_train_stemmed.arff
[START] Convert .txt to .arff
--update:  1396 'datasets' were found
[END] Convert .txt to .arff:  webkb_test_stemmed.arff
MrBurdick txt_to_arff $
```

Weka

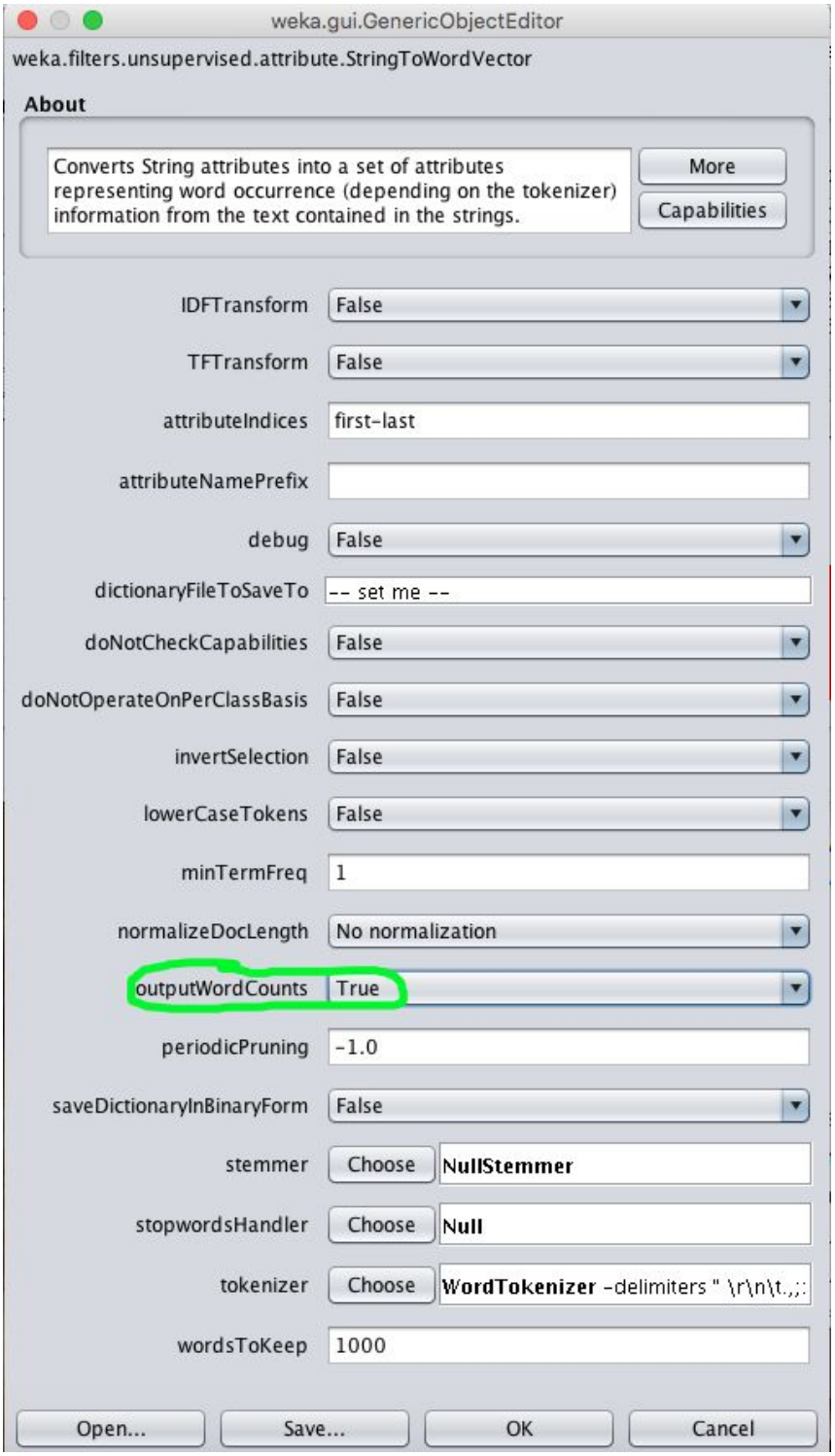
Preprocess [**note about this later, this preprocessing step was ultimately un-done]

1. Open file(s)
   a. 

2. String to word vector filter - /[choose]/weka/filters/unsupervised/attribute/stringtowordvector

a. 

3. StringToWordVector options
    a. To access this you must click on the StringToWordVector (in white, above)

b. 

4. [Apply]
    a. Click [edit] to view table. Then right click and select [class as attribute]

Relation: WebKB Test-weka.filters.unsupervised.attribute.StringToWordVector-R1-W1000-prune-rate-1.0-C-N0-stemmerweka.core.stemmers.NullStem...

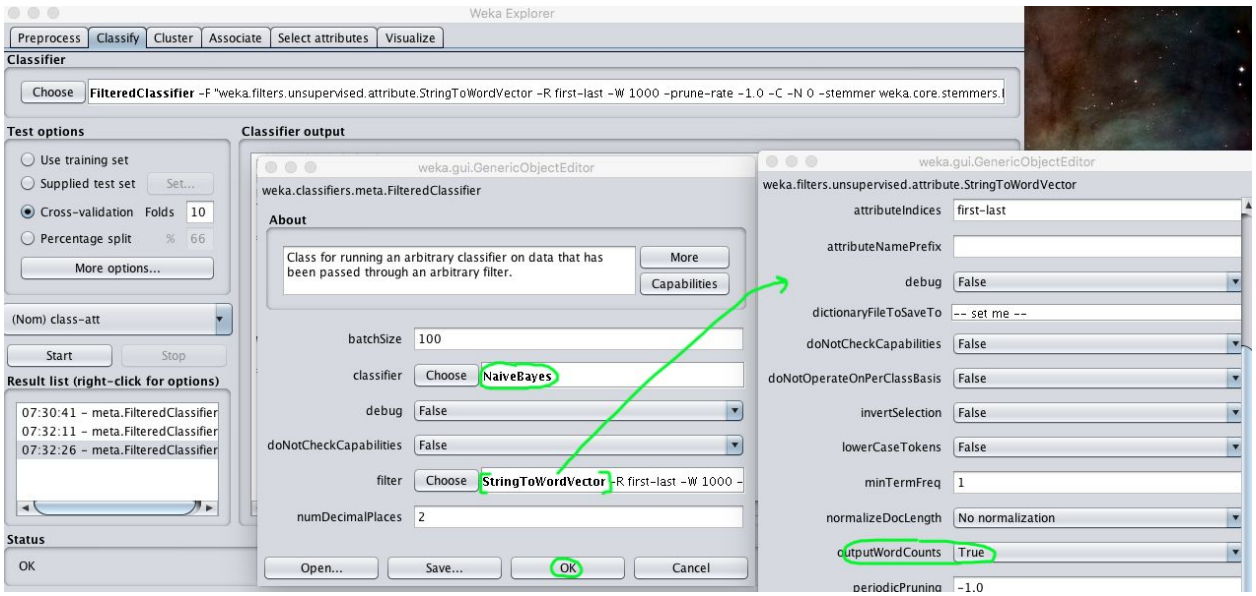
No.	1: aaai	2: abstract	3: academ	4: access	5: acm	6: action	7: activ	8: adam	9: adapt	10: addit	11: address	12: administr	13: advanc	14: advisor
	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	3.0	0.0	0.0	0.0	0.0	2.0	0.0	0.0	0.0
5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
8	2.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	0.0	0.0	0.0
...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0
...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

5. Select the “classify” tab **Steps 1-4(above) were skipped when producing the final results. Otherwise, we will receive an error message saying something to the effect that the training and test sets are not compatible. If we ignore this error and continue with the weka recommended mapped classifier for the training and test sets, the results produced are different than our final outcome. The way I choose to get around this, as shown below, was to use the FilteredClassifier option in weka. This allows us to select our classifier and our filter and produces a desirable output, without error.

   NOTE: The default “folds = 10” is selected for all sets

Naive Bayes

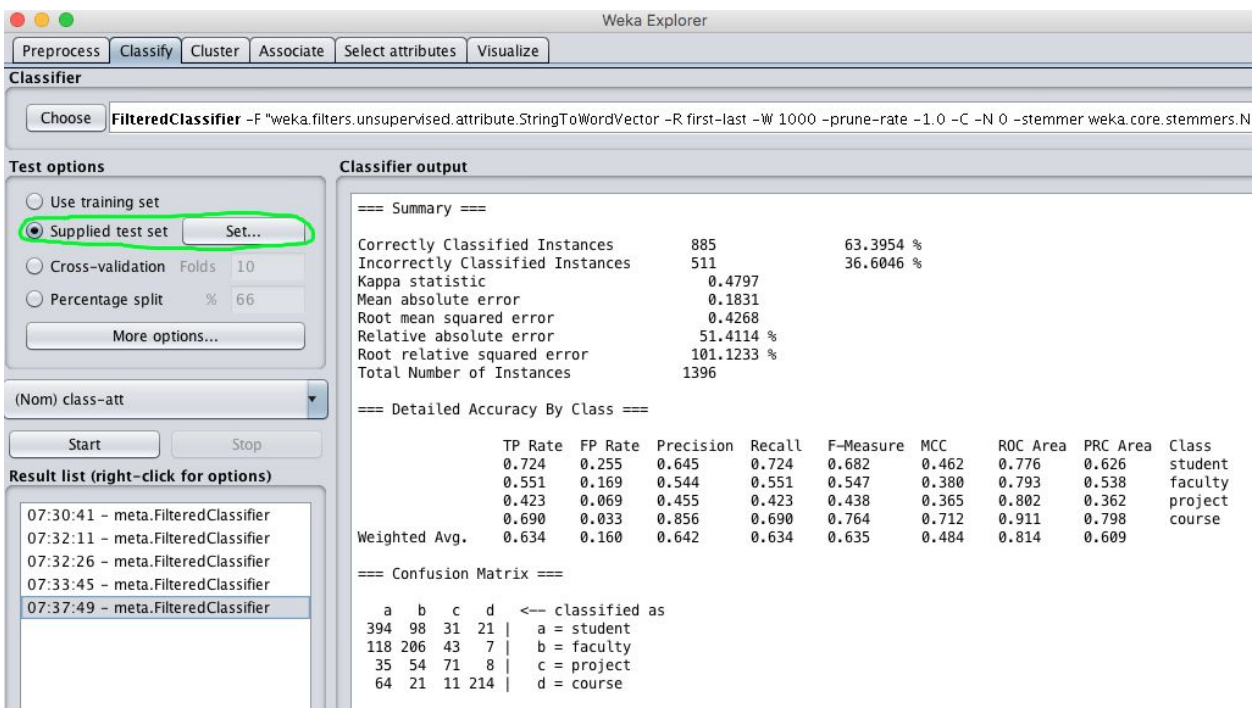
Setup



10-fold cross-validation on the training set

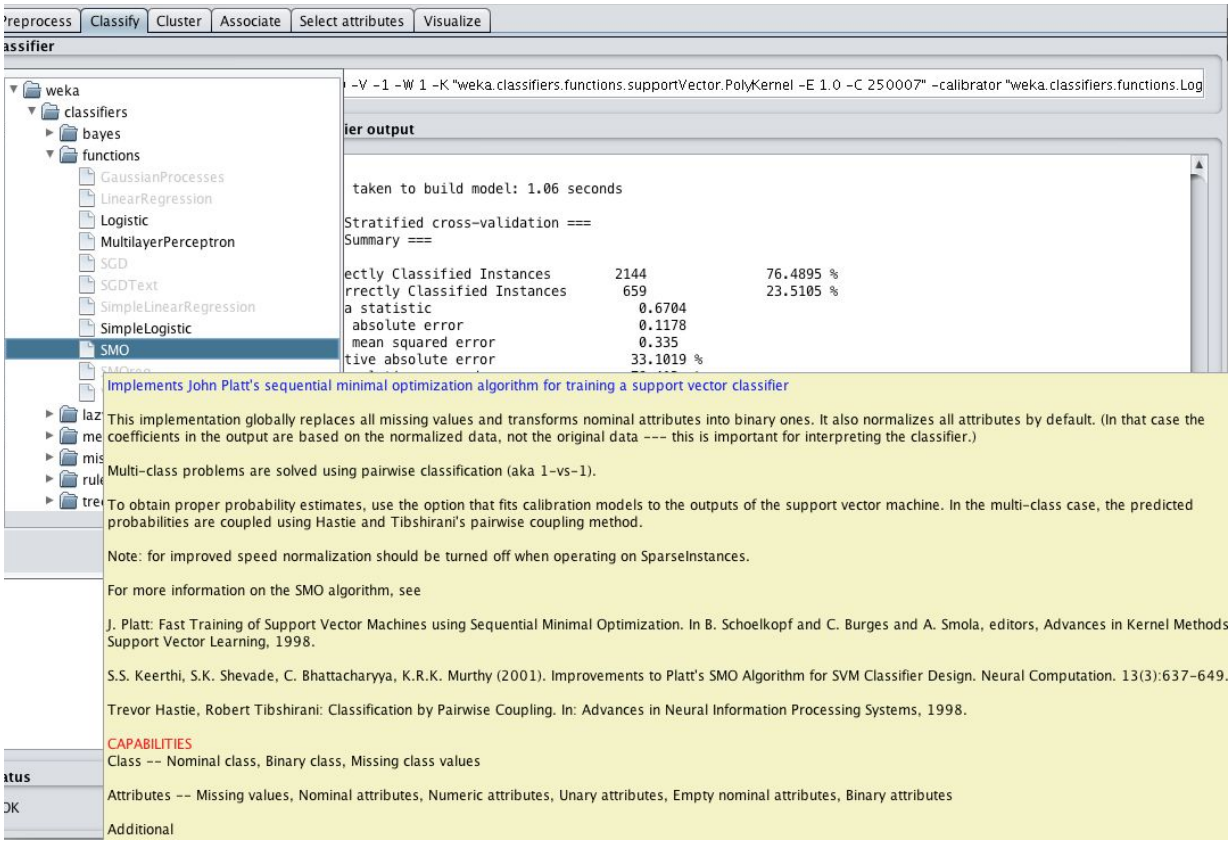
```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        1818               64.8591 %
Incorrectly Classified Instances       985               35.1409 %
Kappa statistic                          0.4997
Mean absolute error                      0.1755
Root mean squared error                  0.4177
Relative absolute error                 49.2962 %
Root relative squared error             99.0101 %
Total Number of Instances             2803

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.755    0.261    0.650      0.755    0.698      0.484    0.793     0.638     student
                 0.476    0.142    0.550      0.476    0.510      0.350    0.765     0.524     faculty
                 0.491    0.064    0.511      0.491    0.501      0.434    0.850     0.435     project
                 0.755    0.041    0.840      0.755    0.795      0.743    0.941     0.828     course
Weighted Avg.    0.649    0.157    0.649      0.649    0.646      0.499    0.825     0.625

=== Confusion Matrix ===

   a   b   c   d   <-- classified as
 828 166  60  43 |   a = student
 271 357  88  34 |   b = faculty
  69  90 165  12 |   c = project
 106  36  10 468 |   d = course
```

Test set setup and results (same FilteredClassifier options, different document)

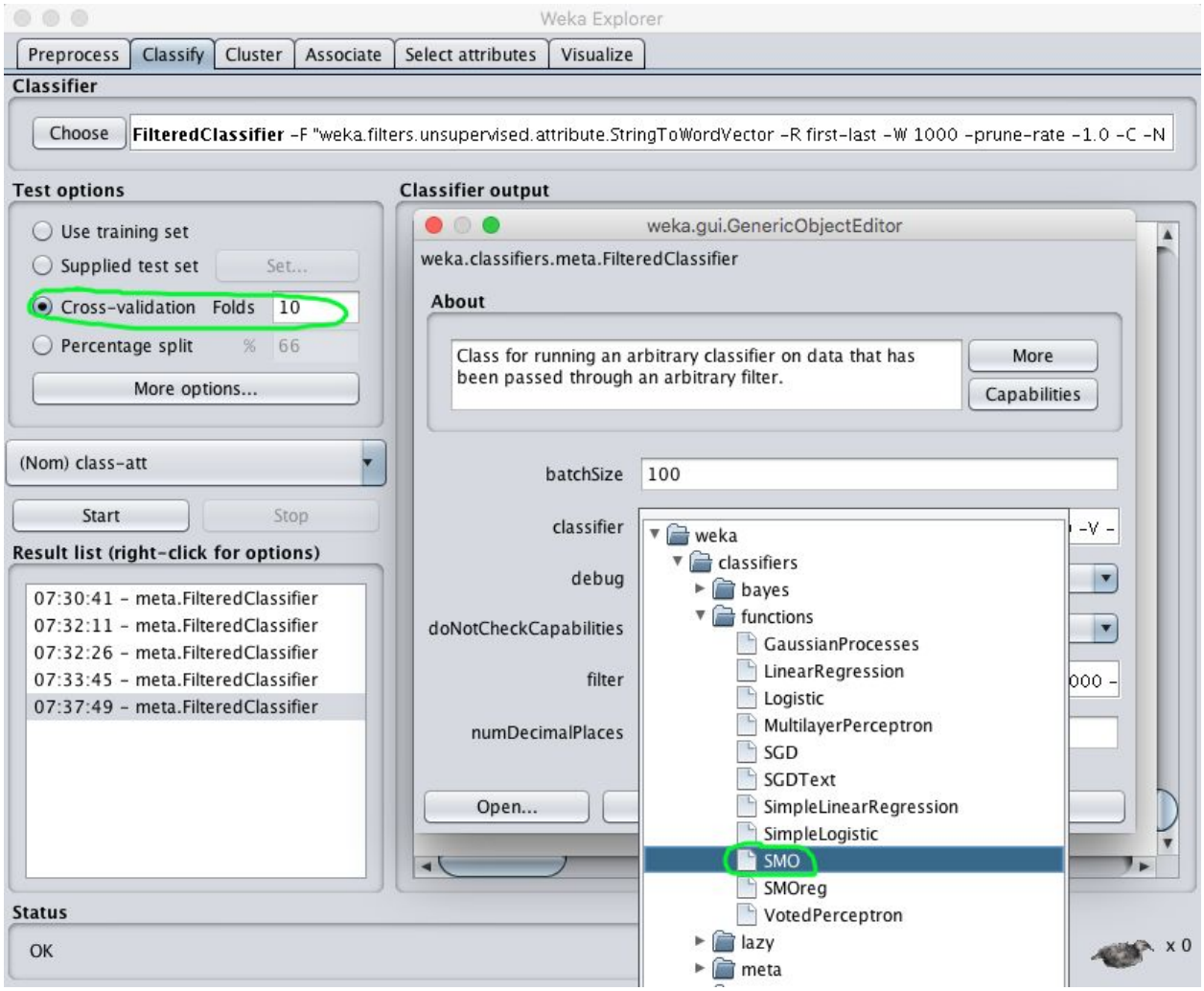


SVM

This is harder to find → [choose] / weka/classifiers/functions/ SMO (see included screenshot for information)



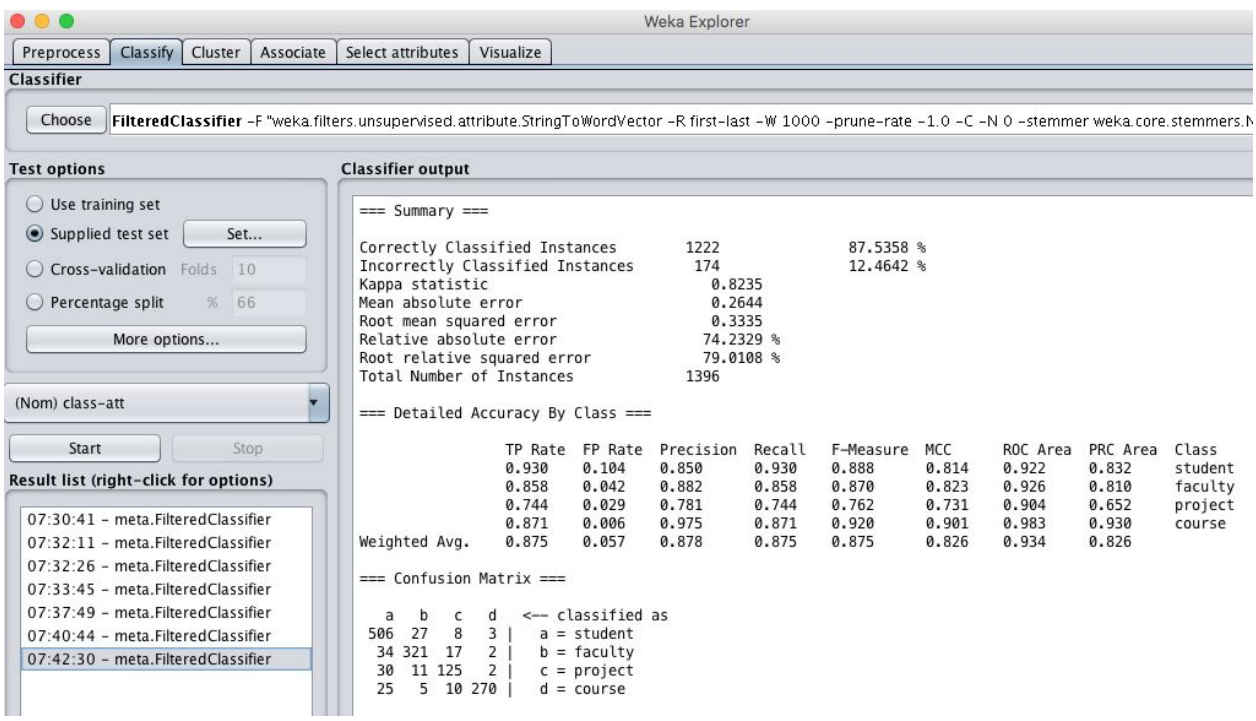
Setup



10-fold cross-validation on the training set

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

Correctly Classified Instances        2446               87.2636 %
Incorrectly Classified Instances       357               12.7364 %
Kappa statistic                          0.8186
Mean absolute error                      0.2642
Root mean squared error                  0.3332
Relative absolute error                 74.2161 %
Root relative squared error             78.9877 %
Total Number of Instances             2803

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.943    0.121    0.834      0.943    0.885      0.808    0.924     0.825     student
                 0.835    0.036    0.894      0.835    0.863      0.817    0.913     0.804     faculty
                 0.682    0.022    0.806      0.682    0.739      0.710    0.908     0.649     project
                 0.897    0.010    0.962      0.897    0.928      0.910    0.985     0.929     course
Weighted Avg.    0.873    0.062    0.875      0.873    0.871      0.821    0.933     0.821

=== Confusion Matrix ===

    a    b    c    d   <-- classified as
 1035   42   12    8 |    a = student
   84  626   34    6 |    b = faculty
   75   24  229    8 |    c = project
   47    8    9  556 |    d = course
```

Test set setup and results (same FilteredClassifier options, different document)



Development Notes

Classifying

Initially I ran into many problems trying to run the classifiers. I documented this experience above in steps 1-4 with an explanation in step 5. Ultimately, the FilteredClassifier weka option was used with the parameters of a string to word vector and our classifier option (Naive Bayes or SVM/SMO).

Word Count

As noted above (in screenshot, circled in green), when using the StringToWordVector filter, I changed the default value of `false` to `true` for the outputWordCounts option. The assignment specifies that we should consider the word frequency when generating our document-word matrix.

Results

[see included table attachment, screen shot below]

Results	Weighted Average [4 class: student, faculty, project, course]								
Training set*	Correctly Classified Rate	TP-Rate	FP-Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area
Naive Bayes	64.86%	64.90%	15.70%	64.90%	64.90%	64.60%	49.90%	82.50%	62.50%
SMO	**87.26%**	**87.30%**	**6.20%**	**87.50%**	**87.30%**	**87.10%**	**82.10%**	**93.30%**	**82.10%**
Test set									
Naive Bayes	63.40%	63.40%	16.00%	64.20%	63.40%	63.50%	48.40%	81.40%	60.90%
SMO	**87.54%**	**87.50%**	**5.70%**	**87.80%**	**87.50%**	**87.50%**	**82.60%**	**93.40%**	**82.60%**
Terms									
TP-Rate	True Positive Rate								
FP-Rate	False Positive Rate								
Precision	(TP)/(TP+FP)								
Recall	(TP)/(TP+FN)								
F-Measure	2*Precision*Recall/(Precision+Recall)								
MCC	Mathews Correlation Coefficient								
ROC Area	Receiver Operating Characteristics								
PRC Area	Precision Recall Curve								
Sources: https://weka.wikispaces.com/Primer, https://en.wikipedia.org/wiki/Matthews_correlation_coefficient,									
https://list.waikato.ac.nz/pipermail/wekalist/2012-May/055512.html, https://en.wikipedia.org/wiki/Precision_and_recall									
							*Cross Validation (10-fold)		

Table 1: The differences in results of the two classifiers