

Text classification with Naïve Bayes

Lab 3

The Task

- Building a model for movies reviews in English for classifying it into positive or negative.
- Test classifier on new reviews



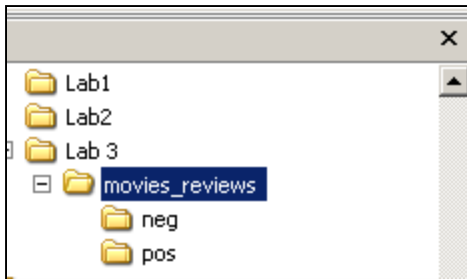
Takes time

Sentiment Polarity Dataset Version 2.0

- 1000 positive movie review and 1000 negative review texts from: *Thumbs up? Sentiment Classification using Machine Learning Techniques. Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan.* Proceedings of EMNLP, pp. 79--86, 2002.
- Our data **source was the Internet Movie Database (IMDb) archive of the rec.arts.movies.reviews newsgroup.**³ We selected only reviews where the **author rating was expressed either with stars or some numerical value (other conventions varied too widely to allow for automatic processing).** Ratings were automatically extracted and converted into one of three categories: positive, negative, or neutral. For the work described in this paper, we concentrated **only** on discriminating between **positive and negative sentiment.**”

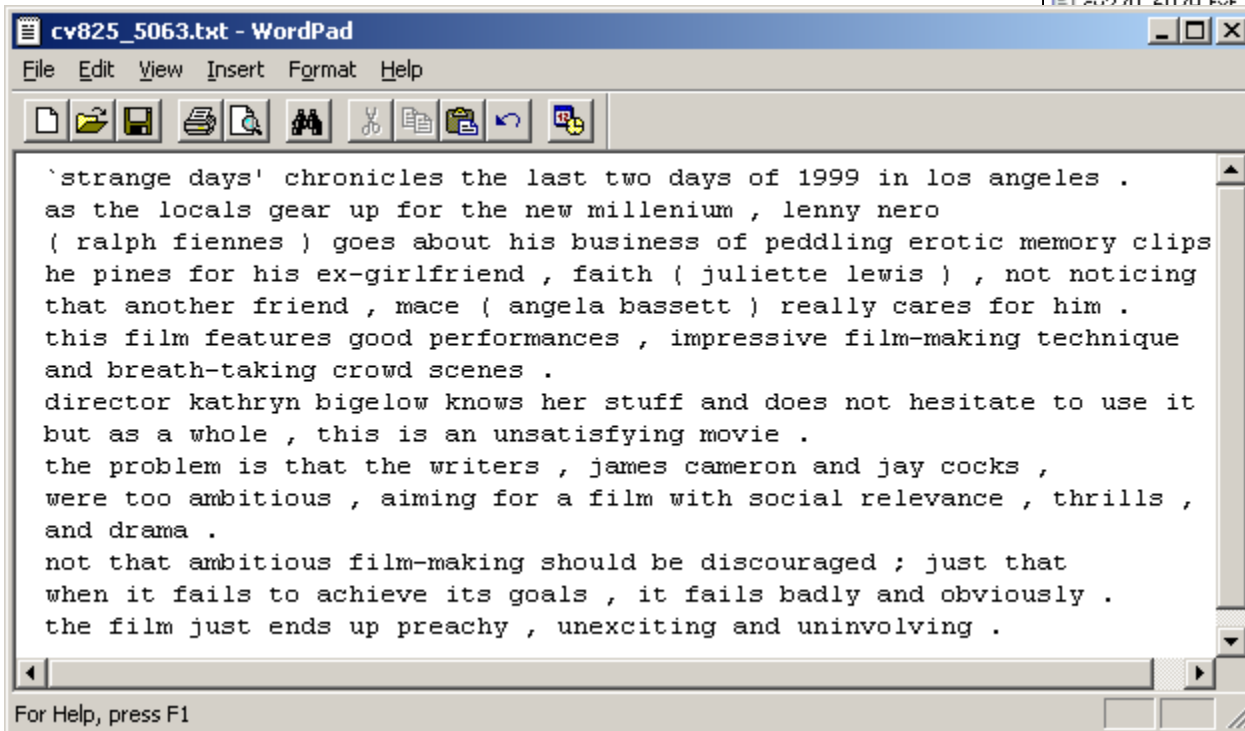
From: <http://www.cs.cornell.edu/people/pabo/movie-review-data/>

The data



A screenshot of a file explorer window showing a list of text files in the 'pos' directory. The path is 'ts\DMCourse\Labs\Lab 3\movies_reviews\pos'. The files are listed in a table with columns for Name, Size, and Type. The file 'cv825_5063.txt' is highlighted in blue. A red box highlights the path 'pos' in the address bar.

Name	Size	Type
cv754_7216.txt	1 KB	Text Document
cv280_8267.txt	1 KB	Text Document
cv825_5063.txt	1 KB	Text Document
cv057_7453.txt	1 KB	Text Document
cv640_5378.txt	2 KB	Text Document
cv987_6965.txt	2 KB	Text Document
cv229_13611.txt	2 KB	Text Document
cv349_13507.txt	2 KB	Text Document
cv841_3967.txt	2 KB	Text Document
cv370_6070.txt	2 KB	Text Document
	2 KB	Text Document
	2 KB	Text Document
	2 KB	Text Document



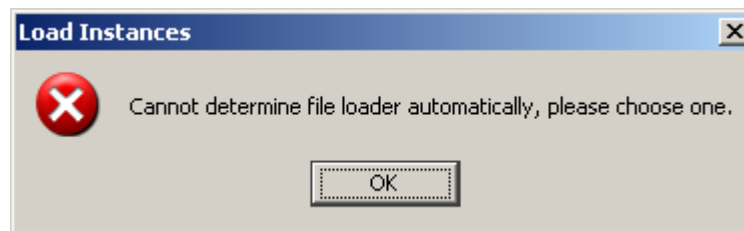
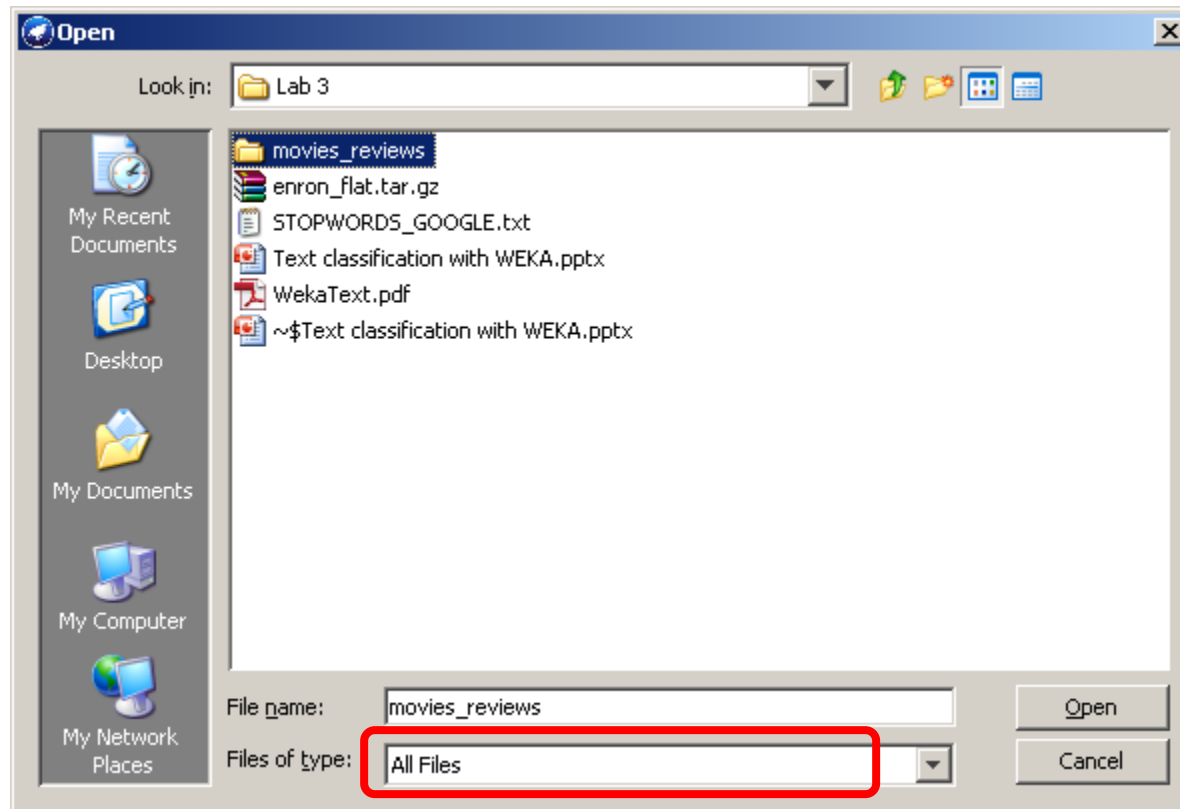
Open WEKA

- Convert file to .arff format using CLI interface:

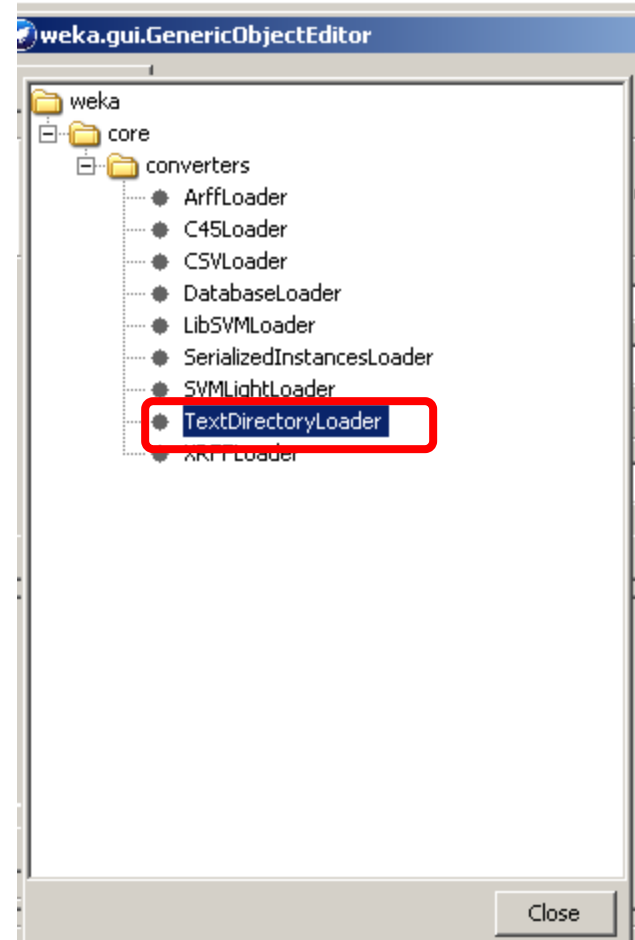
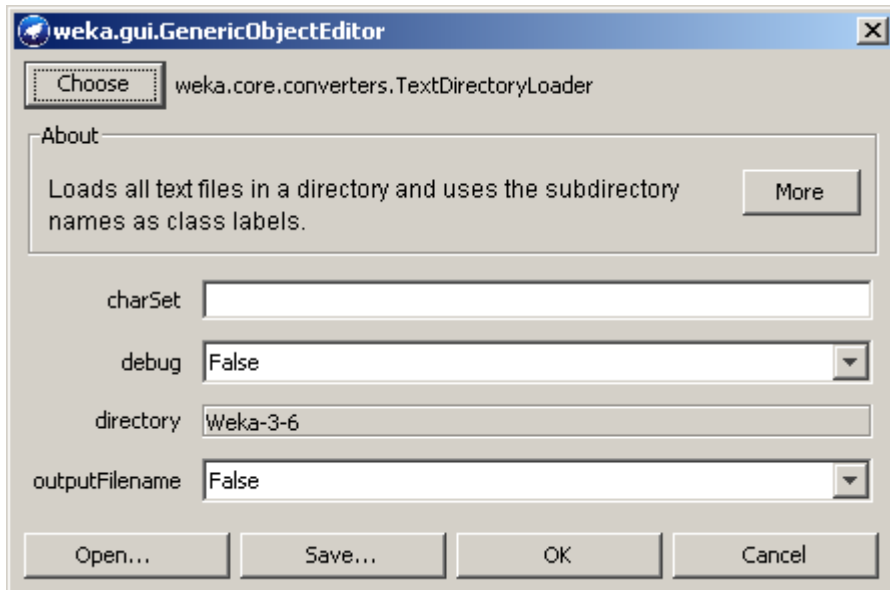
```
java weka.core.converters.TextDirectoryLoader -dir data/movies_reviews > data/movies_reviews.arff
```

- And then open it from WEKA explorer – skip to slide 12

Select folder



Choose converter



Class

The screenshot shows the Weka Explorer application window. The 'Edit...' button in the top toolbar is highlighted with a red rectangle. The interface displays the following information:

- Preprocess** | **Classify** | **Cluster** | **Associate** | **Select attributes** | **Visualize**
- Buttons: Open file..., Open URL..., Open DB..., Generate..., Undo, **Edit...**, Save...
- Filter**: Choose **None** [Apply]
- Current relation**: Relation: C:\Documents and Settings\barskym\My Documents_DMCo...
Instances: 2000 Attributes: 2
- Attributes**: All | None | Invert | Pattern
- Attributes list:

No.	Name
1	text
2	@@class@@
- Selected attribute**: Name: @@class@@ Type: Nominal
Missing: 0 (0%) Distinct: 2 Unique: 0 (0%)
- Attribute details table:

No.	Label	Count
1	neg	1000
2	pos	1000
- Class: @@class@@ (Nom) [Visualize All]
- Visualizations: Two colored squares representing the class distribution. The left square is blue and labeled '1000', and the right square is red and labeled '1000'.
- Status**: OK [Log] x 0

Edit->View

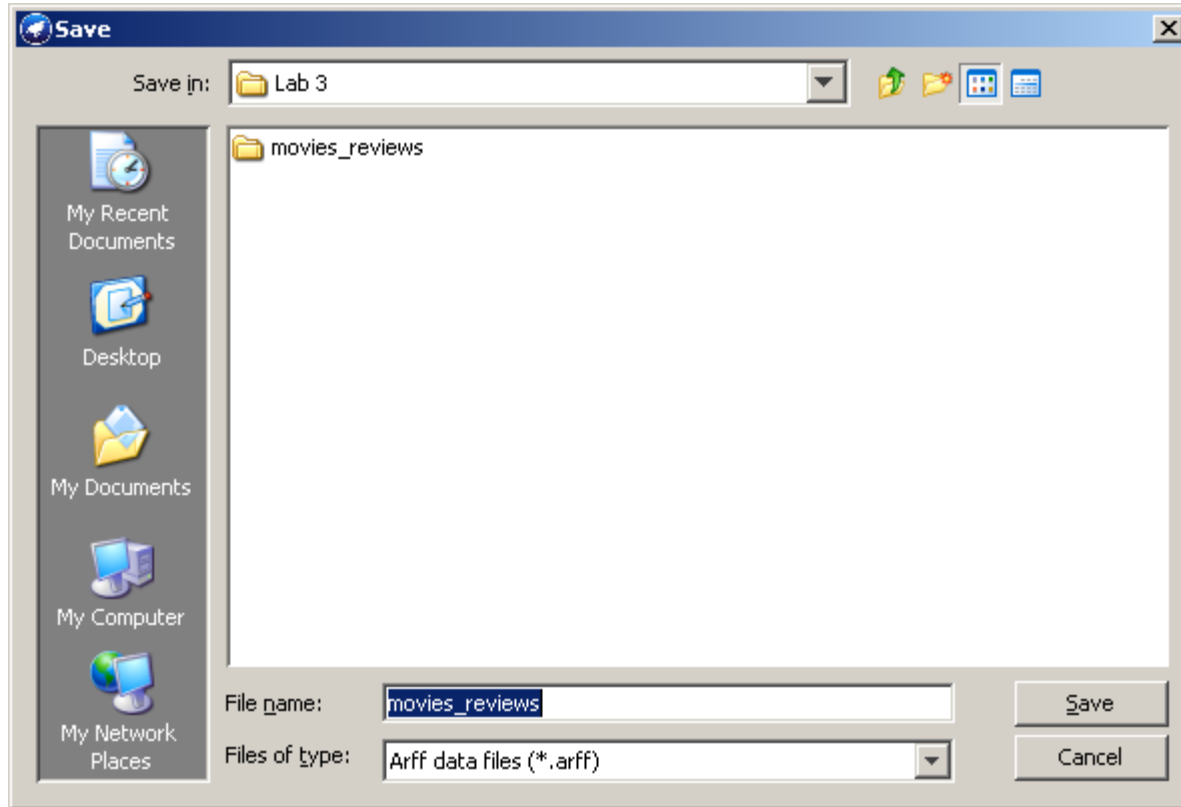
Relation: C:_Documents and Settings_barskym_My Documents_DMCourse_Labs_Lab 3_mo...

No.	text String	@@class@@ Nominal
1	plot : two teen couples go to a church party , drink and then drive .	neg
10	plot : a young french boy sees his parents killed before his eyes by	neg
100	whether you like the beatles or not , nobody wants to see the bee	neg
1000	two party guys bob their heads to haddaway's dance hit " what is	neg
1001	films adapted from comic books have had plenty of success ,	pos
1002	every now and then a movie comes along from a suspect studio ,	pos
1003	you've got mail works alot better than it deserves to . \n in order to	pos
1004	" jaws " is a rare film that grabs your attention before it shows you	pos
1005	moviemaking is a lot like being the general manager of an nfl team in	pos
1006	on june 30 , 1960 , a self-taught , idealistic , yet pragmatic , young	pos
1007	apparently , director tony kaye had a major battle with new line	pos
1008	one of my colleagues was surprised when i told her i was willing to	pos
1009	after bloody clashes and independence won , lumumba refused to	pos
101	warning : spoilers are included in this review . . . \n but it doesn't	neg
1010	the american action film has been slowly drowning to death in a sea	pos
1011	after watching " rat race " last week , i noticed my cheeks were sore	pos
1012	i've noticed something lately that i've never thought of before .	pos
1013	synopsis : bobby garfield (yelchin) lives in a small town with his	pos
1014	synopsis : in this movie , steven spielberg , one of today's finest	pos
1015	the police negotiator is the person with the entirely unenviable job	pos
1016	plot : a young man who loves heavy metal music and especially the	pos
1017	carry on matron is the last great carry-on film in my opinion .	pos
1018	the ultimate match up between good and evil , " the untouchables "	pos

Right click (or left

Undo OK Cancel

Save converted file in arff format



From text to vectors

- $D = [w_1, w_2, w_3, \dots, \text{class}]$
- review1 = "great movie"
- review2 = "excellent film"
- review3 = "worst film ever"
- review4 = "sucks"

	ever	excellent	film	great	movie	sucks	worst	class
V1=[0,	0,	0,	1,	1,	0,	0,	+]
V2=[0,	1,	1,	0,	0,	0,	0,	+]
V3=[1,	0,	1,	0,	0,	0,	1,	-]
V4=[0,	0,	0,	0,	0,	1,	0,	-]

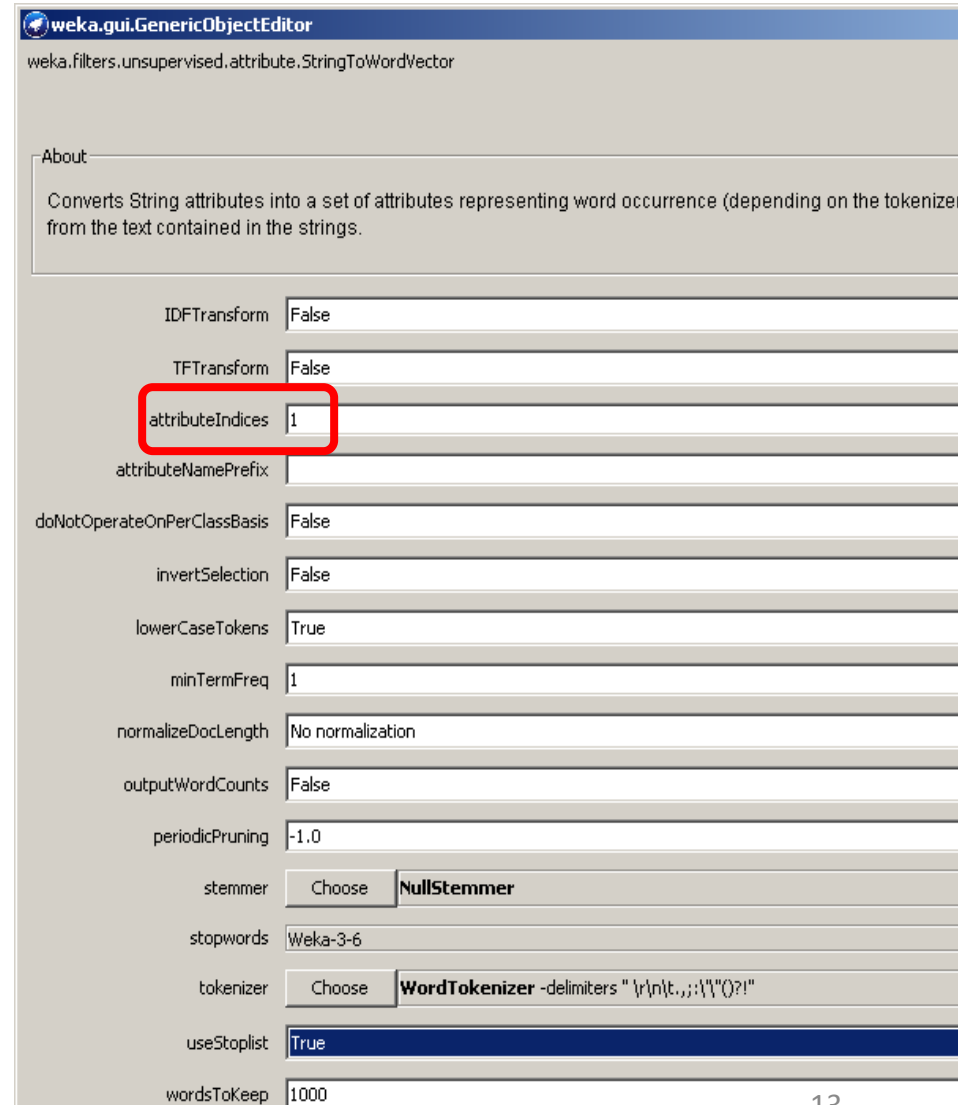
Convert text field into word vectors

- Filter->
- Unsupervised->
- Attribute ->
- StringToWordVector

- This will convert each word in string field into a numeric attribute
- The name of the attribute would be the word itself
- The value of the attribute would be 0 (absent) or 1 (present) in the current document

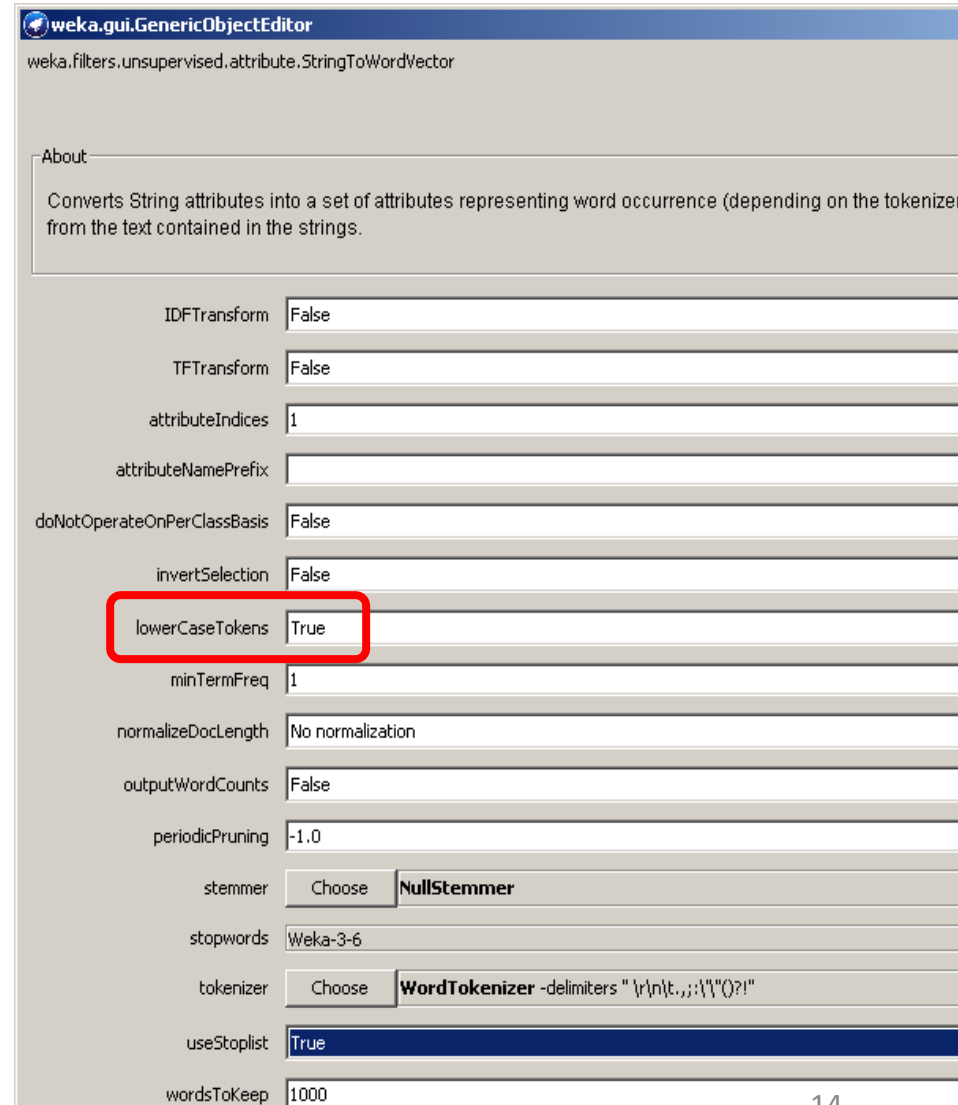
Right-click for options

- Select attribute to convert



Options

- Can convert all words to lower case – preferred



Options

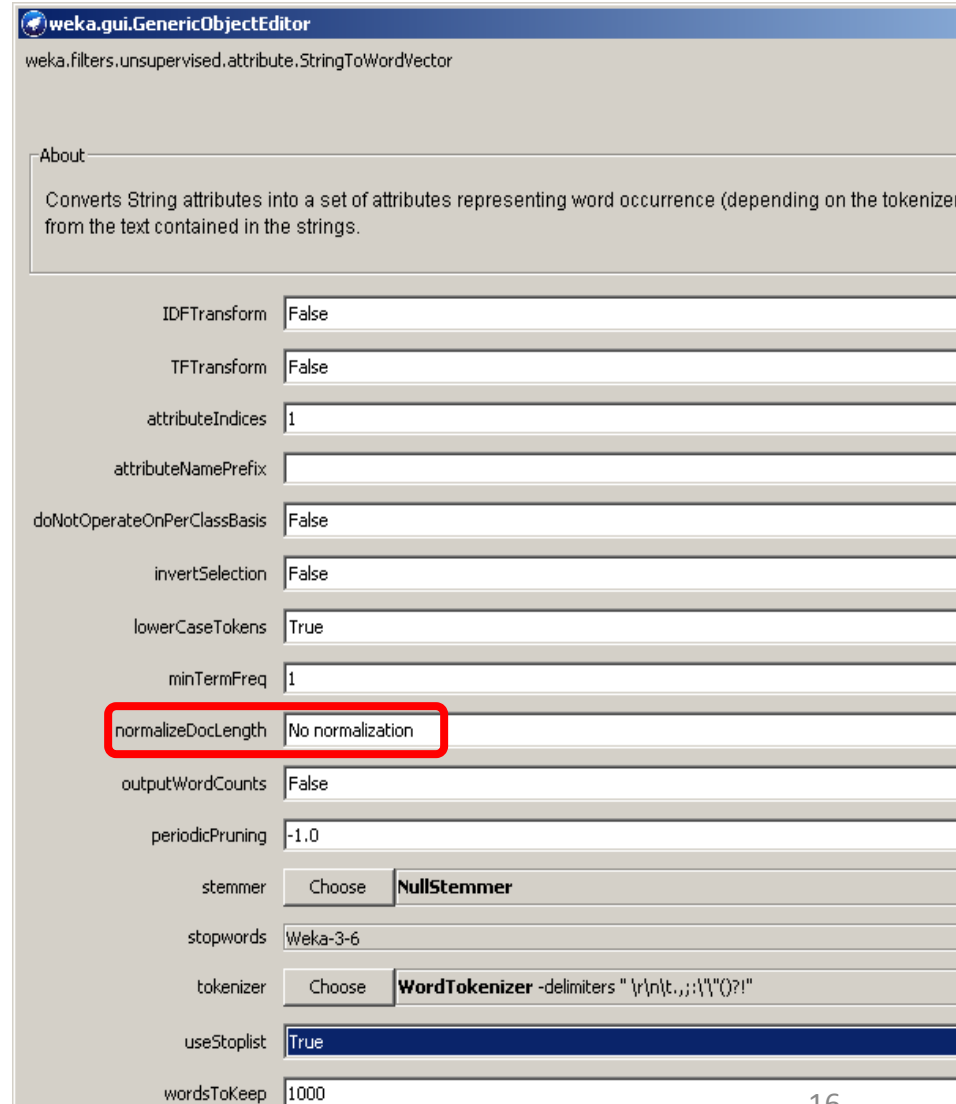
- Can output the words counts in each document, instead of just occurrence.
- We will use the boolean 'present-absent' for this lab

The screenshot shows the Weka GUI Generic Object Editor for the `weka.filters.unsupervised.attribute.StringToWordVector` filter. The 'About' section states: "Converts String attributes into a set of attributes representing word occurrence (depending on the tokenizer) from the text contained in the strings." The configuration options are as follows:

Option	Value
IDFTransform	False
TFTransform	False
attributeIndices	1
attributeNamePrefix	
doNotOperateOnPerClassBasis	False
invertSelection	False
lowerCaseTokens	True
minTermFreq	1
normalizeDocLength	No normalization
outputWordCounts	False
periodicPruning	-1.0
stemmer	Choose NullStemmer
stopwords	Weka-3-6
tokenizer	Choose WordTokenizer -delimiters " \r\n\t,.;: '\"{}?!"
useStoplist	True
wordsToKeep	1000

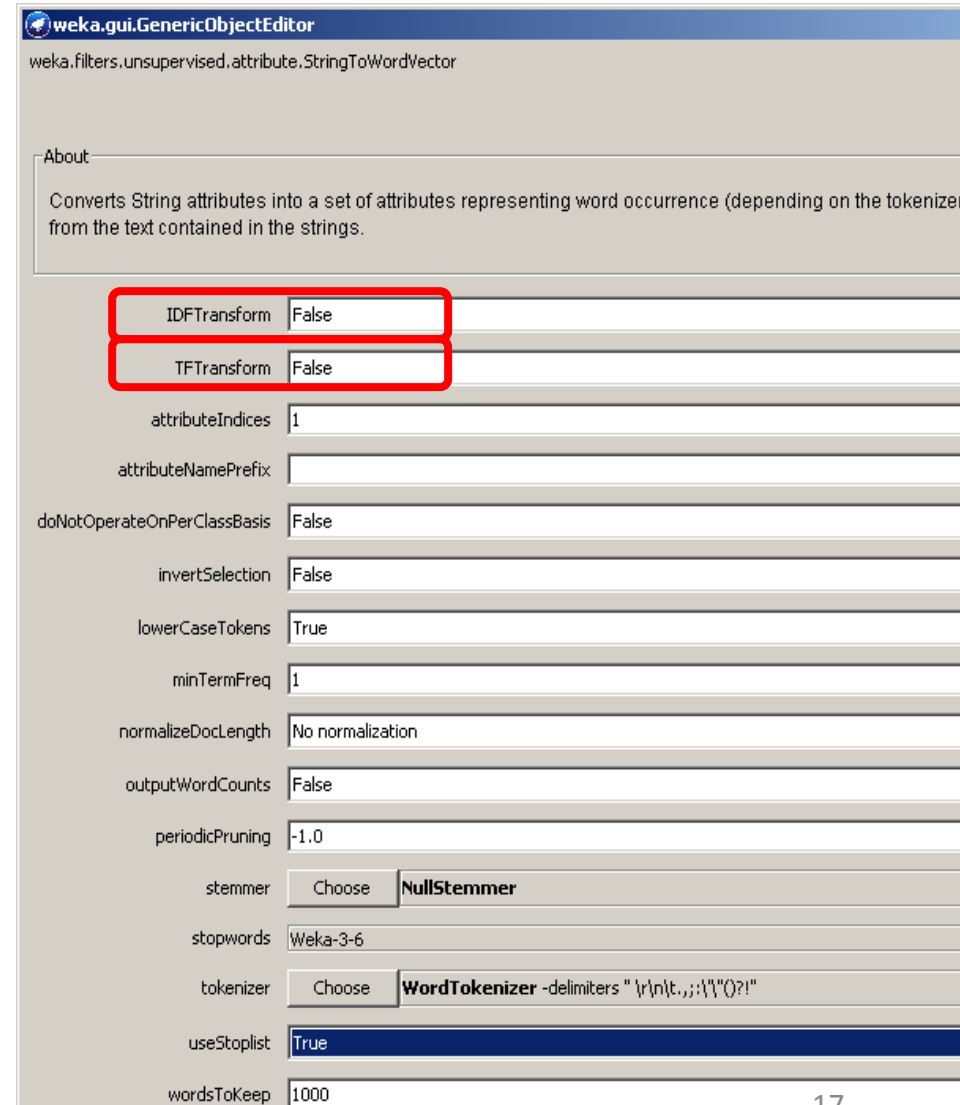
Options

- If 'outputWordsCounts' is selected, can normalize word counts by dividing to the total number of words in the document – thus creating the normalized word vector



Options

- If 'outputWordsCounts' is selected, can do TF or IDF-transform or both for each document:



Term Frequency – Document Frequency

$Tf(\text{word}_i, d) = \text{frequency}(\text{word } i) / [\text{total words in } d]$

$DF(\text{word}_i) = \text{number of documents containing } \text{word}_i / \text{total number of documents}$

The bigger TF the more discriminative is word_i

The smaller DF the more discriminative is word_i

TF-IDF index for each word_i in document d

- Term Frequency – Inverted Document Frequency model

$$\text{TF-IDF}(\text{word}_i, d) = \text{TF}(\text{word}_i) * \underbrace{\log(1/\text{DF}(\text{word}_i))}_{\text{Inverse document frequency - IDF}}$$

Inverse document
frequency - IDF

The bigger TF-IDF score, the more discriminative is word_i

TF-IDF example

- Consider a document containing 100 words wherein the word *cow* appears 3 times. Following the previously defined formulas, the term frequency (TF) for *cow* is then $(3 / 100) = 0.03$. Now, assume we have 10 million documents and *cow* appears in one thousand of these. Then, the inverse document frequency is calculated as $\log(10\,000\,000 / 1\,000) = 4$. The tf-idf score is the product of these quantities: $0.03 \times 4 = 0.12$.

Options

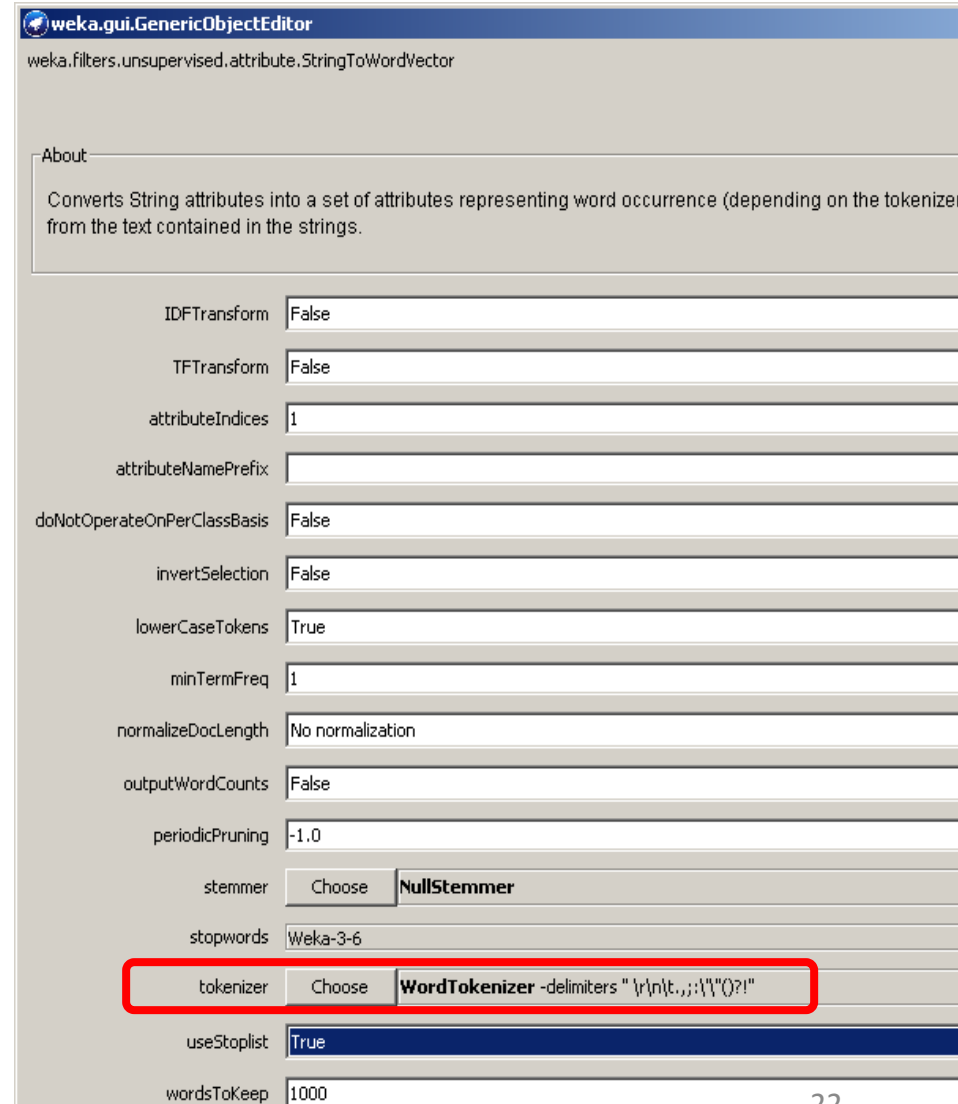
- Keeps 1000 most frequent words for each class.
Sometimes a little more, if there is a tie
- Less frequent words are discarded

The screenshot shows the 'weka.gui.GenericObjectEditor' window for the 'weka.filters.unsupervised.attribute.StringToWordVector' filter. The 'About' section states: 'Converts String attributes into a set of attributes representing word occurrence (depending on the tokenizer from the text contained in the strings.' The configuration options are as follows:

IDFTransform	False
TFTransform	False
attributeIndices	1
attributeNamePrefix	
doNotOperateOnPerClassBasis	False
invertSelection	False
lowerCaseTokens	True
minTermFreq	1
normalizeDocLength	No normalization
outputWordCounts	False
periodicPruning	-1.0
stemmer	Choose NullStemmer
stopwords	Weka-3-6
tokenizer	Choose WordTokenizer -delimiters " \r\n\t,;:\\""?"
useStoplist	True
wordsToKeep	1000

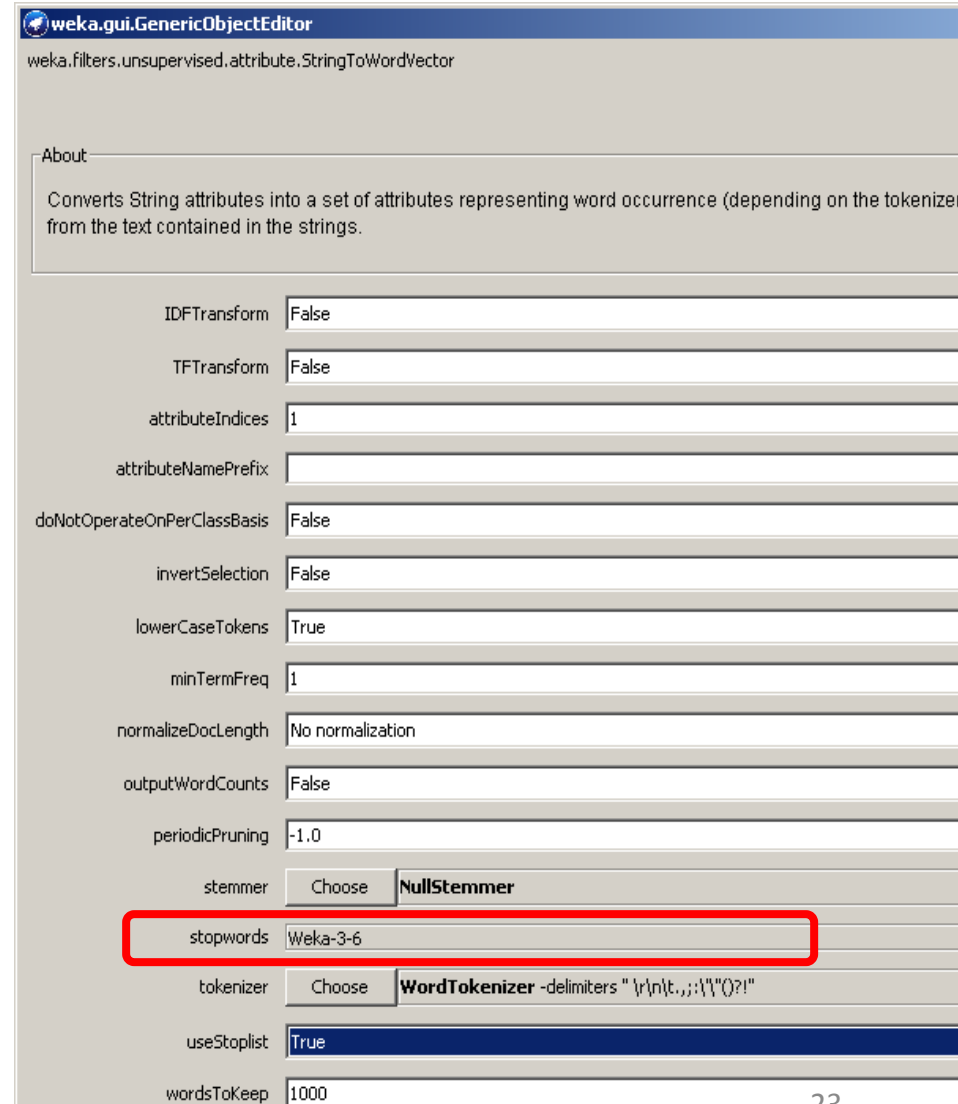
How the words are extracted

- The tokenizer is supplied with the list of characters which are considered as delimiters. The extracted word is the trimmed string between two delimiters
- You can provide your own application-dependent tokenizer



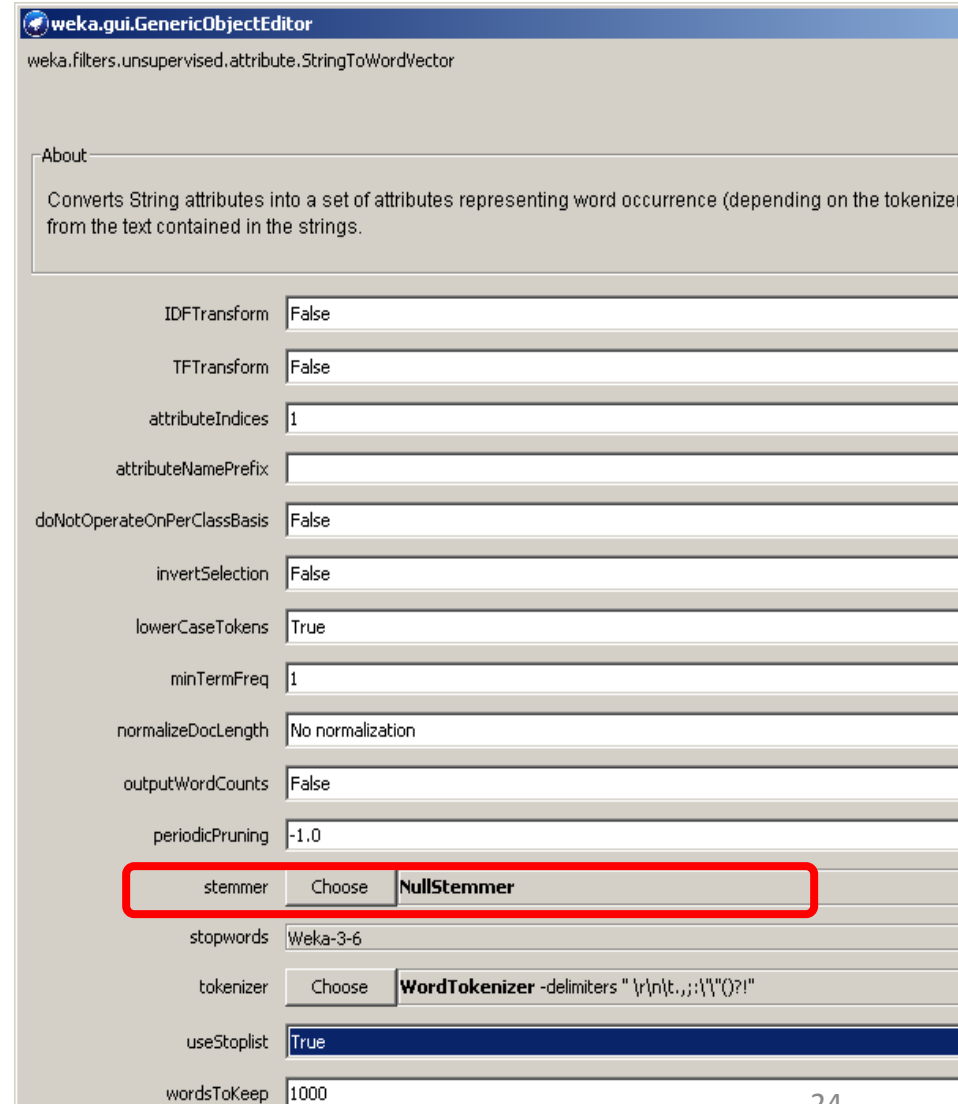
How the words are extracted

- Words such as “the”, “in”, “of” are removed – they occur in each document
- You can replace the default list with your own stop word list such as supplied stopwords_google.txt



How the words are extracted

- Stemmer – identifies words which have the same root, for example: “cat”, “cats”, “catlike”, “catty” have similar meaning related to the root “cat” and stemmer treats them all as the same word. No stemmer by default



Done with options

- Select attributeIndex=1, the rest are default attributes and apply the filter
- Note: now the class attribute is the first, in addition – its name is not a valid attribute name

Move class attribute to the end of vector

The screenshot shows a data viewer application with a table of data and a context menu open over it. The table has columns for a nominal attribute and several numerical attributes. The context menu includes options like 'Attribute as class', 'Sort data (ascending)', and 'Optimal column width'. A red box highlights the 'Attribute as class' option. To the right, a bar chart displays the distribution of the nominal attribute, with two bars: a blue bar for 'neg' and a red bar for 'pos', both with a count of 1000. The 'Edit...' button in the top right of the application is also highlighted with a red box.

Relation: C:\Documents and Settings\barskym\My Documents\DMCourse_Labs_Lab 3_mo...

Sort view: left click = ascending / Shift + left click = descending
Menu: right click (or left+alt)

No.	@@class@@	&	*	-	--	000	1
1	neg						
2	neg						
3	neg						
4	neg						
5	neg						
6	neg						
7	neg						
8	neg						
9	neg						
10	neg						
11	neg						
12	neg						
13	neg						
14	neg						
15	neg	0.0	0.0	1.0	0.0	0.0	0.0
16	neg	0.0	0.0	1.0	0.0	0.0	1.0
17	neg	0.0	0.0	0.0	0.0	0.0	0.0
18	neg	0.0	0.0	0.0	0.0	0.0	0.0
19	neg	0.0	0.0	0.0	0.0	0.0	0.0
20	neg	0.0	0.0	1.0	0.0	1.0	0.0
21	neg	1.0	0.0	0.0	0.0	1.0	0.0
22	neg	0.0	0.0	0.0	1.0	0.0	0.0
23	neg	0.0	0.0	1.0	0.0	0.0	0.0

Attribute: @@class@@
Type: Nominal
Distinct: 2
Unique: 0 (0%)

Label	Count
neg	1000
pos	1000

@@class@@ (Nom) Visualize All

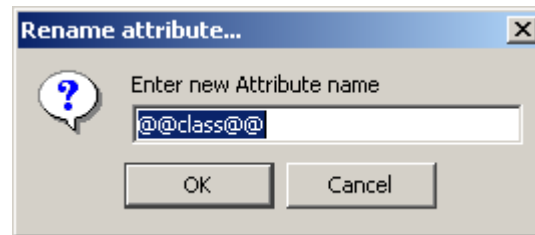
Remove

Status: OK

Log

Rename class attribute

- documentClass



Clean some junk words

Weka Explorer

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

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

Filter: Choose **Reorder -R first-last** Apply

Current relation
Relation: C:\Documents and Settings\barskym\My Documents_DMC...
Instances: 2000 | Attributes: 1241

Attributes

All | None | Invert | Pattern

No.		Name
7	<input checked="" type="checkbox"/>	1
8	<input checked="" type="checkbox"/>	10
9	<input checked="" type="checkbox"/>	1998
10	<input checked="" type="checkbox"/>	1999
11	<input checked="" type="checkbox"/>	2
12	<input checked="" type="checkbox"/>	3
13	<input checked="" type="checkbox"/>	4
14	<input checked="" type="checkbox"/>	90
15	<input checked="" type="checkbox"/>	=
16	<input type="checkbox"/>	ability
17	<input type="checkbox"/>	absolutely
18	<input type="checkbox"/>	accent
19	<input type="checkbox"/>	act
20	<input type="checkbox"/>	acting

Remove

Selected attribute

Name: = | Type: Numeric
Missing: 0 (0%) | Distinct: 2 | Unique: 0 (0%)

Statistic	Value
Minimum	0
Maximum	1
Mean	0.009
StdDev	0.092

Class: documentClass (Nom) Visualize All

0 0.5 1 28

Status

Convert all numeric to nominal (boolean)

The screenshot shows the Weka Explorer interface. The 'Filter' tab is active, and the 'NumericToNominal -R first-last' filter is selected and highlighted with a red box. The 'Current relation' is 'C:\Documents and Settings\barskym\My Documents_DMCo...', with 2000 instances and 1226 attributes. The 'Attributes' list shows various nominal attributes, with 'documentClass' at the bottom. The 'Selected attribute' section shows 'ability' with 2 distinct values (0 and 1) and 0 missing values. A table below shows the distribution: 1874 instances for '0' and 126 instances for '1'. A bar chart visualizes this distribution, with a red bar for '0' (1874) and a blue bar for '1' (126). The 'Class' is set to 'documentClass (Nom)' and 'Visualize All' is clicked.

Weka Explorer

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

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

Filter

Choose **NumericToNominal -R first-last** Apply

Current relation

Relation: C:\Documents and Settings\barskym\My Documents_DMCo...
Instances: 2000 Attributes: 1226

Attributes

All None Invert Pattern

No.	Name
1213	unique
1214	united
1215	vampire
1216	view
1217	viewing
1218	vincent
1219	visually
1220	wars
1221	washington
1222	witty
1223	wonderfully
1224	woody
1225	younger
1226	documentClass

Remove

Selected attribute

Name: ability
Missing: 0 (0%)
Distinct: 2
Type: Nominal
Unique: 0 (0%)

No.	Label	Count
1	0	1874
2	1	126

Class: documentClass (Nom) Visualize All

1874

126

Status

OK Log x 0

Visualize all attributes

The screenshot shows the Weka Explorer interface with the 'Visualize' tab selected. The 'Filter' section shows 'NumericToNominal -R 1-1155' applied. The 'Current relation' is 'C:_Documents and Settings_barskym_My Documents_DMCo...' with 2000 instances and 1156 attributes. The 'Attributes' list on the left includes 'toy', 'trek', 'truman', 'truth', 'typical', 'unique', 'unlike', 'view', 'wars', 'wedding', 'wonderful', 'wonderfully', 'woody', and 'documentClass'. The 'Selected attribute' section shows 'documentClass' with a nominal type and 2 distinct values. A table below shows the distribution: 'neg' (1000) and 'pos' (1000). The 'Visualize All' button is highlighted with a red box. The bar chart below shows two bars: a blue bar for 'neg' and a red bar for 'pos', both with a count of 1000.

Weka Explorer

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

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

Filter

Choose **NumericToNominal -R 1-1155** Apply

Current relation

Relation: C:_Documents and Settings_barskym_My Documents_DMCo...
Instances: 2000 Attributes: 1156

Attributes

All | None | Invert | Pattern

No.	Name
1143	toy
1144	trek
1145	truman
1146	truth
1147	typical
1148	unique
1149	unlike
1150	view
1151	wars
1152	wedding
1153	wonderful
1154	wonderfully
1155	woody
1156	documentClass

Remove

Selected attribute

Name: documentClass Type: Nominal
Missing: 0 (0%) Distinct: 2 Unique: 0 (0%)

No.	Label	Count
1	neg	1000
2	pos	1000

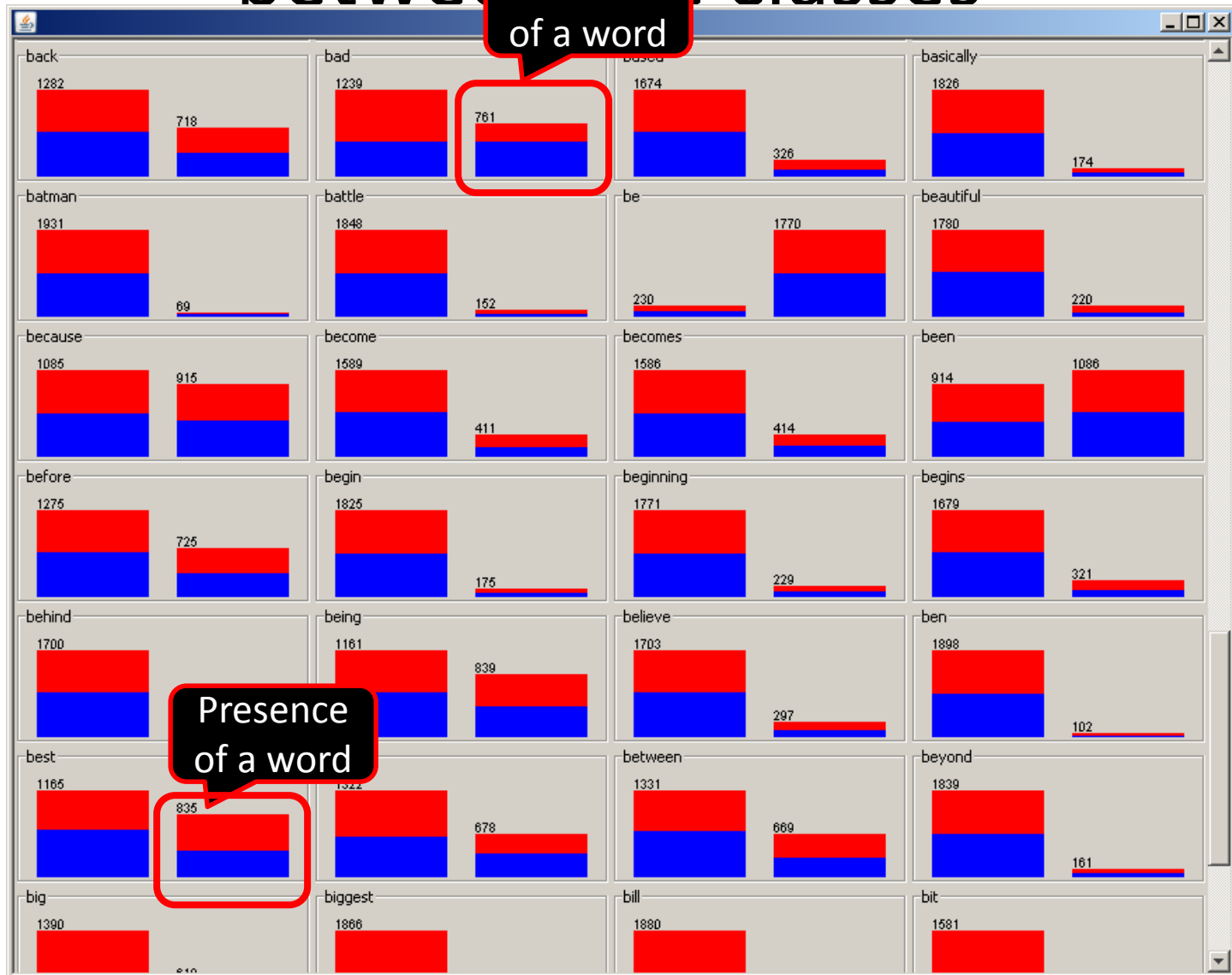
Class: documentClass (Nom) Visualize All

1000 1000

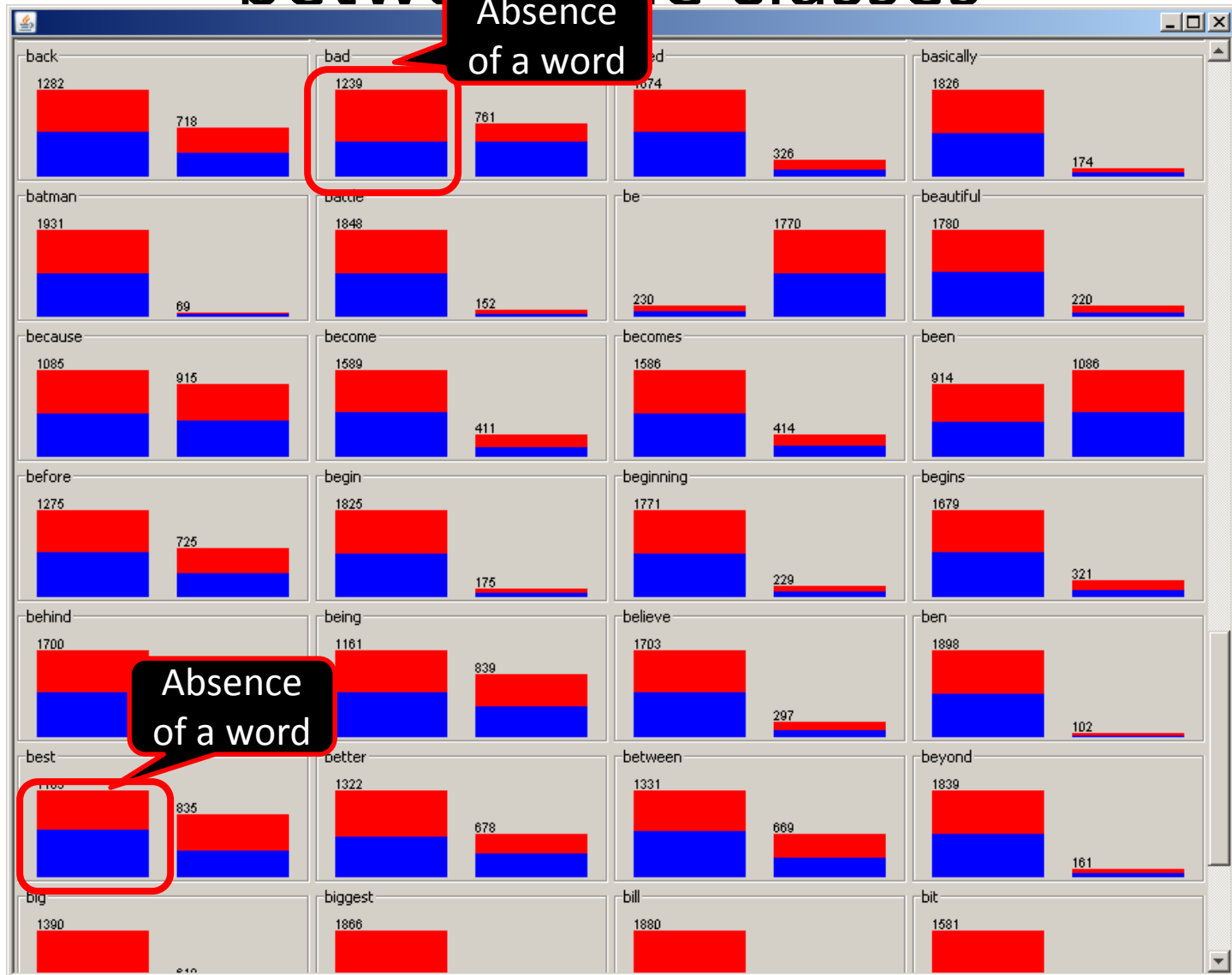
neg [1000]

Status: OK Log x 0

Presence of a word discriminates between the classes



Absence of a word discriminates between the classes



Classify using Naïve Bayes

The screenshot shows the Weka Explorer interface with the Naive Bayes classifier selected. The 'Test options' section has 'Percentage split' selected with a value of 66%. The 'Classifier output' pane shows the following information:

```
=== Run information ===  
  
Scheme:weka.classifiers.bayes.NaiveBayes  
Relation: C:\Documents and Settings\barskym\My Documents\DMCourse_Labs_Lab  
Instances: 2000  
Attributes: 1156  
[list of attributes omitted]  
Test mode:split 66.0% train, remainder test
```

The status bar at the bottom indicates 'Building model on training data...'. A red speech bubble on the left contains the text: 'Don't use cross-validation - takes too much time'.

Don't use cross-validation - takes too much time

Result

- Accuracy: 78.67 %
- We might try to improve it by selecting only the best words

Feature (discriminative words) selection

- WEKA class [CfsSubsetEval](#) evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them. Subsets of features that are highly correlated with the class while having low inter-correlation are preferred.

To find such attributes WEKA uses [BestFirst](#) search: Searches the space of attribute subsets by greedy hillclimbing augmented with a backtracking facility. Setting the number of consecutive non-improving nodes allowed controls the level of backtracking done. Best first may start with the empty set of attributes and search forward, or start with the full set of attributes and search backward, or start at any point and search in both directions (by considering all possible single attribute additions and deletions at a given point).

Attribute (words) selection

The screenshot shows the Weka Explorer interface with the 'Select attributes' tab selected. The 'Attribute Evaluator' is set to 'CfsSubsetEval' and the 'Search Method' is 'BestFirst -D 1 -N 5'. The 'Attribute Selection Mode' is set to 'Use full training set'. The 'Result list' shows a single entry: '10:08:41 - BestFirst + CfsSubsetEval'. The 'Attribute selection output' window displays the following information:

```
=== Run information ===  
  
Evaluator:   weka.attributeSelection.CfsSubsetEval  
Search:weka.attributeSelection.BestFirst -D 1 -N 5  
Relation:   C:_Documents and Settings_barskym_My Documents_DMCourse_Labs_La  
Instances:  2000  
Attributes: 1156  
[list of attributes omitted]  
Evaluation mode:evaluate on all training data
```

The status bar at the bottom indicates 'Evaluating on training data...' and includes a 'Log' button and a small bird icon.

Selected discriminative attributes (words): 51

- Right-click result -> save reduced data as Movies_reviews_reduced.arff

also	world
awful	worst
bad	animation
boring	definitely
both	deserves
dull	effective
fails	flaws
great	greatest
joke	hilarious
lame	memorable
life	overall
many	perfectly
maybe	realistic
mess	share
nothing	solid
others	subtle



Some of selected words

Classify again

- Accuracy 78.67 – no improvement, but only 51 words instead of 1155

Rules-Decision table

- Accuracy – 69.85
- Feature set: 3,4,10,14,17,33,48,51,52
- Leave only these attributes, plus class attribute, and use naïve bayes again
- Accuracy – 72.05


Classifying new reviews

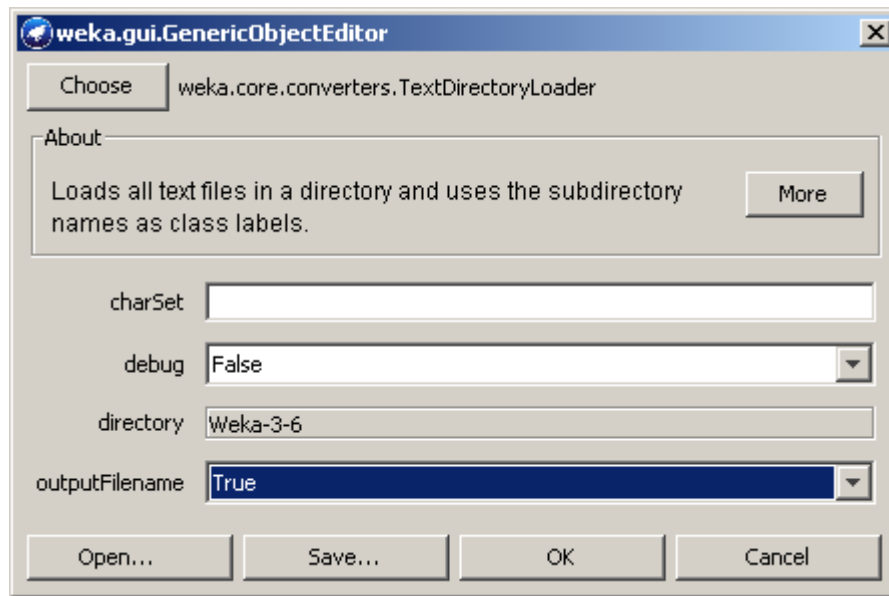
- Append – put new reviews into a positive folder
- Do the same manipulation on the new dataset
- Remove new reviews into a new test set
- Classify

New reviews to classify



- Go to: <http://www.rottentomatoes.com/>
- or
<http://www.cinemaclock.com/Nanaimo.html>
- Select one movie and find reviews
- Generate test set by copying the text of each review into a separate .txt file
- Store all new files in the 'pos' folder under the names NewReview1-5.txt

Repeat all the steps with new reviews included

- Open file-> all files -> select folder  movie_reviews (Select option-outputFileName – to be sure where your new reviews are)



Continue work on attributes

- Filter -> StringToWordVector: AttributeIndex = 1 
- Remove some remaining meaningless words from the top of the list 
- Edit -> select @class@ -> right-click -> AttributeAsClass
- select @class@ -> right-click -> rename to DocumentClass

Select attributes

- Select attributes with default parameters and save data file as `movies_review_training.arff`

Continue work on attributes

- Convert numeric 0-1 values to nominal
- Save as `movies_reviews_training.arff`



Transfer new records to a new text file

- Open your `movie_reviews_training.arff` in text editor
- Copy the entire header up to the `@data` tag into a new text file
- Cut and paste the last 5 lines for new reviews into the same new file
- Save as `movie_reviews_new.arff`
- Load `movie_reviews_training.arff` and remove the `fileID` attribute. Save file
- Load `movie_reviews_new.arff` :
 - Remove `fileID` attribute
 - Replace class value with `'?'` (Edit-> Replace values with ...->pos to `'?'`)
 - Save file

Run classifier for prediction

- inst#, actual, predicted, error, probability distribution
 - 1 ? 1:neg + *1 0
 - 2 ? 1:neg + *1 0
 - 3 ? 1:neg + *1 0
 - 4 ? 1:neg + *0.999 0.001
 - 5 ? 1:neg + *0.993 0.007
-
- Test if the classifier is correct