

# Mallet

Fei Xia

# Mallet

- Machine learning toolkit
  - Developed at UMass Amherst by Andrew McCallum
  - Java implementation, open source
  - Large collection of machine learning algorithms
    - Targeted to language processing
    - Naïve Bayes, MaxEnt, Decision Trees, Winnow, Boosting
    - Also, clustering, topic models, sequence learners
  - Widely used, but
    - Research software: some bugs/gaps; odd documentation

# Installation

- Installed on patas already under  
/NLP\_TOOLS/tool\_sets/mallet/latest/
- Subdirectories:
  - bin/: script files
  - src/: java source code
  - class/: java classes
  - lib/: jar files
  - sample-data/: wikipedia docs for languages id, etc

# Environment

- Should be set up on patas
  - \$PATH should include  
`/NLP_TOOLS/tool_sets/mallet/latest/bin`
  - \$CLASSPATH should include  
`/NLP_TOOLS/tool_sets/mallet/latest/lib/mallet-deps.jar`
  - Check: type “which text2vectors” or “which mallet”  
The path should be `/NLP_TOOLS/tool_sets/mallet/latest/bin/`

# Mallet Commands

- Mallet command types:
  - Data preparation
  - Data/model inspection
  - Training
  - Classification
- Command line scripts
  - Shell scripts
    - ❖ Set up java environment
    - ❖ Invoke java programs
  - --help lists command line parameters for scripts

# Mallet Data

- Text format: Users of Mallet create training/test instances in this format
  - standard format: InstanceName label f1 v1 f2 v2 .....
  - svmlight format: label f1:v1 f2:v2 ...
- Binary format: used by learner and decoder
  - It stores the mapping from featName to featIdx, from targetLabel to targetIdx, etc.
- Mallet has tools to convert between the two formats

# Data preparation

- Define features
- Create feature vectors for each training/test instance; save them in a text vector format
  - ➔ write your own code
- Run “Mallet import-file” to convert the text format to binary format.

# Convert from standard text format to binary format

- `mallet import-file --input file1 --output file2`
  - `file1`: input file
    - ❖ feature vectors in the new text format, one line per instance:  
InstanceName label f1 v1 f2 v2 ..... fn vn
    - ❖ Features can strings or indexes
  - `file2`: output file
    - Feature vectors in the binary format
- If building test data separately from original
  - `mallet import-file --input train.vectors.txt --output train.vectors`
  - `mallet import-file --input test.vectors.txt --output test.vectors --use-pipe-from train.vectors`



# Convert from svmlight format to binary format

- `mallet import-svmlight --input file1 --output file2`
  - file1: input file
    - ❖ feature vectors in the new text format, one line per instance:  
label f1:v1 f2:v2 ... fn:vn
    - ❖ Features can strings or indexes
  - file2: output file
    - Feature vectors in the binary format
- If building test data separately from original
  - `mallet import-svmlight --input train.vectors.txt --output train.vectors`
  - `mallet import-svmlight --input test.vectors.txt --output test.vectors --use-pipe-from train.vectors`

# Convert from binary to text format

- `vectors2info --input vectors --print-labels TRUE > labelList`
  - Prints list of category labels in data set
- `vectors2info --input vectors --print-matrix sic > vectors.txt`
  - prints all features and values by string and number
  - Returns original text feature-value list
  - Possibly out of order

# Training

- `mallet train-classifier --trainer trainerName --input train.vectors --output-classifier modelName 1>log.stdout 2>log.stderr`
- `trainerName`: MaxEnt, DecisionTree, NaiveBayes, etc
- The code creates the following:
  - `modelName` (the model): features and their weights
  - `log.stdout`: the report, including training acc, confusion matrix
  - `log.stderr` (the training info): iteration values, etc.

# Viewing the model

- `classifier2info --classifier modelName > model.txt`  
It prints out contents of model file
- An example model:

FEATURES FOR CLASS guns

<default> 0.1298

fire 0.3934

firearms 0.4221

government 0.3721

arabic -0.0204

# Accuracy and confusion matrix

- Confusion Matrix, row=true, column=predicted  
accuracy=0.9711111111111111

	label	0	1	2	total
0	misc	846	27	23	896
1	mideast	12	899	2	913
2	guns	12	2	877	891

Train accuracy mean = 0.9711

# Testing

- Use new data to test a previously built classifier
- `mallet classify-svmlight --input testfile --output outputfile --classifier maxent.model`
- It prints (class, score) pairs for each test instance in the format of “Inst\_id class1 score1 class2 score2”
- An example:

array:0	en	0.995	de	0.0046
array:1	en	0.970	de	0.0294
array:2	en	0.064	de	0.935
array:3	en	0.094	de	0.905

# Training + testing + eval

- `vectors2classify --training-file train.vectors --testing-file test.vectors --trainer DecisionTree --report test:raw test:accuracy test:confusion train:confusion train:accuracy > de1.stdout 2>de1.stderr`
- The training and test accuracy is at the end of de1.stdout

# Summary

- Create feature vectors in the text format
- Convert vectors to the binary format:  
mallet import-file --input train.vectors.txt --output train.vectors  
mallet import-file --input test.vectors.txt --output test.vectors --use-pipe-from train.vectors
- mallet train-classifier --input train.vectors --trainer MaxEnt --output-classifier ml.model
  - Trains MaxEnt classifier and stores model
- mallet classify-file --input test.vectors.txt --output result --classifier ml.model
  - Tests on the new data and output classification result
  - It does not show test accuracy



# Other commands

- Viewing the vectors: `vectors2info`
- Viewing the model: `classifier2info`
- `vectors2classify`: training, test and eval

# Other Information

- Website:
  - Download and documentation (as it is)
  - <http://mallet.cs.umass.edu>
- API tutorial:
  - <http://mallet.cs.umass.edu/mallet-tutorial.pdf>

# Split binary vectors into training and test portions

- `vectors2vectors --input input-filename --training-file training-filename --testing-file test-filename --training-portion pct`
  - Creates **random** training/test splits in some ratio, pct
  - Therefore, running the command multiple times will yield different results.