



Module 11: Machine Learning



What is Machine Learning and why do we want to do it?

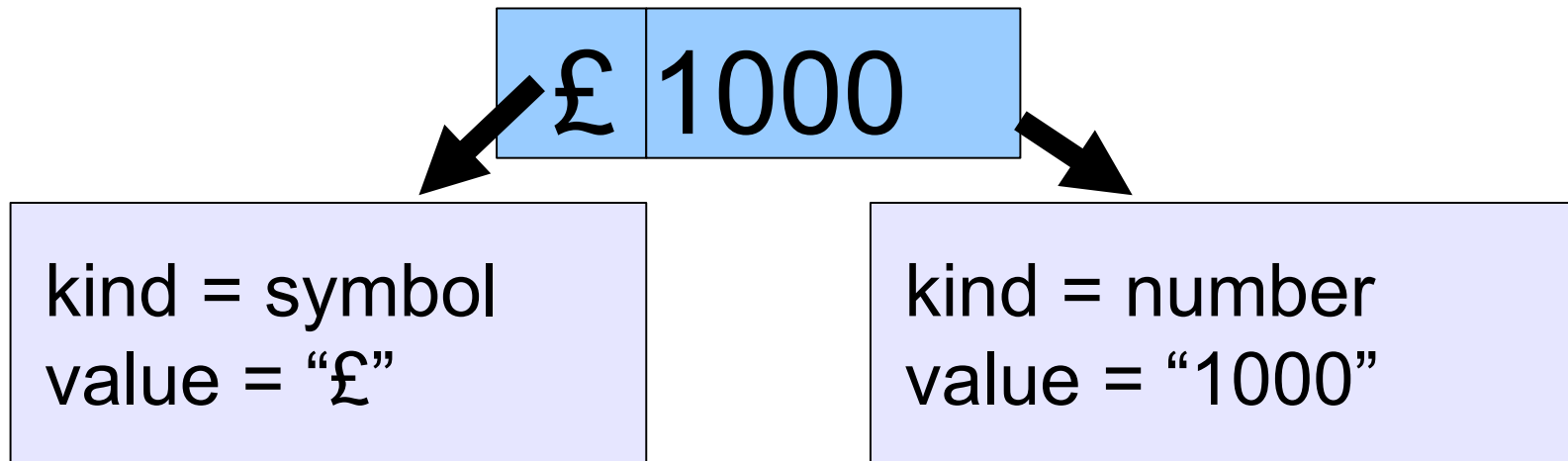


What is ML?

- Automating the process of inferring new data from existing data
- In GATE, that means creating annotations by learning how they relate to other annotations

Learning a pattern

- For example, we have “Token” annotations with “kind” and “value” features



- ML could learn that a “£” followed by a number is an amount of currency

How is that better than making rules?

- It is different to the rule-based approach
- Humans are better at writing rules for some things, and ML algorithms are better at finding some things
- With ML you don't have to create all the rules
- However, you have to manually annotate a training corpus (or get someone else to do it!)
- Rule-based approaches (e.g. JAPE) and ML work well together; JAPE is often used extensively to prepare data for ML



Terminology: Instances, attributes, classes

California Governor Arnold Schwarzenegger proposes deep cuts.

Instances:

Any annotation
Tokens are often convenient



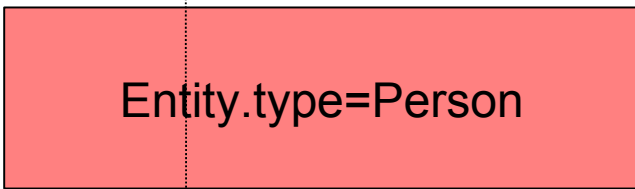
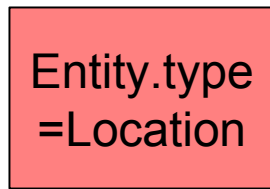
Attributes:

Any annotation feature relative to instances
Token.String
Token.category (POS)
Sentence.length



Class:

The thing we want to learn
A feature on an annotation





Instances

- Instances are cases that may be learned
- Every instance is a decision for the ML algorithm to make
- To which class does this instance belong?
 - “California” → Location



Attributes

- Attributes are pieces of information about instances
- They are sometimes called “features” in machine learning literature
- Examples
 - `Token.string == “Arnold”`
 - `Token.orth == upperInitial`
 - `Token(-1).string == “Governor”`



Classes

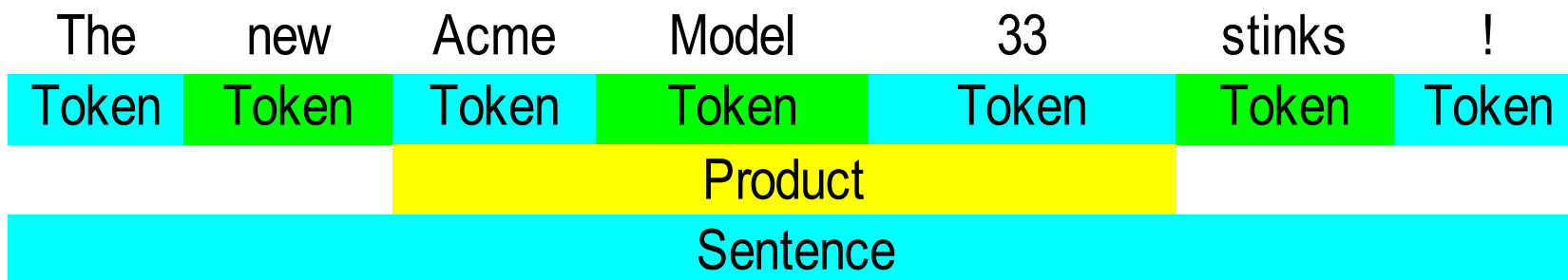
- The class is what we want to learn
- Suppose we want to find persons' names: for every instance, the question is “is this a person name?” and the classes are “yes” and “no”
- Sometimes there are many classes, for example we may want to learn entity types
 - For every instance, the question is “which type from the list does this instance belong to?”
 - One answer is “none of them”



ML Tasks

- GATE supports 3 types of ML tasks:
 - chunk recognition (named entity recognition, NP chunking) as in the previous example
 - text classification (sentiment classification, POS tagging) as in the following example
 - relation annotation (this requires special techniques that are not covered in this module, although materials are available)

Example: text classification



- instance: Sentence annotation
- attributes: Token and Product annotations and their features (suppose that the Product annotations have been created earlier with gazetteers and rules)
- class: polarity= “negative”
- ML could learn that a Product close to the Token “stinks” expresses a negative sentiment, then add a polarity=“negative” feature to the Sentence.



Training

- Training involves presenting data to the ML algorithm from which it creates a model
- The training data (instances) have been annotated with class annotations as well as attributes
- Models are representations of decision-making processes that allow the machine learner to decide what class the instance has based on the attributes of the instance



Application

- When the machine learner is applied, it creates new class annotations on data using the model
- The corpus it is applied to must contain the required attribute annotations
- The machine learner will work best if the application data is similar to the training data



Evaluation

- We want to know how good our machine learner is before we use it for a real task
- Therefore we apply it to some data for which we already have class annotations
 - The “right answers”, sometimes called “gold standard”
- If the machine learner creates the same annotations as the gold standard, then we know it is performing well
- The test corpus must not be the same corpus as you trained on
 - This would give the machine learner an advantage, and would give a false idea of how good it is
- GATE's ML PR has a built-in evaluation mode that splits the corpus into training and test sets and cross-validates them

Setting up a Corpus



Load the corpus

- Create a corpus (any name is fine; you can even leave it blank)
- Populate it from `ner/corpus/*.xml` in the Module 11 hands-on materials
 - Set the encoding to UTF-8
- You should get 93 documents (numbered 0 to 92 in the corpus)
- Open a document and examine its annotations



Examining the corpus

- The corpus contains an annotation set called “Key”, which has been manually prepared
- Within this annotation set are annotations of types “Date”, “Location”, “Money”, “Organization” and so forth
- There are also some annotations in the “Original markups” set (these represent HTML tags)



What are we going to do with this corpus?

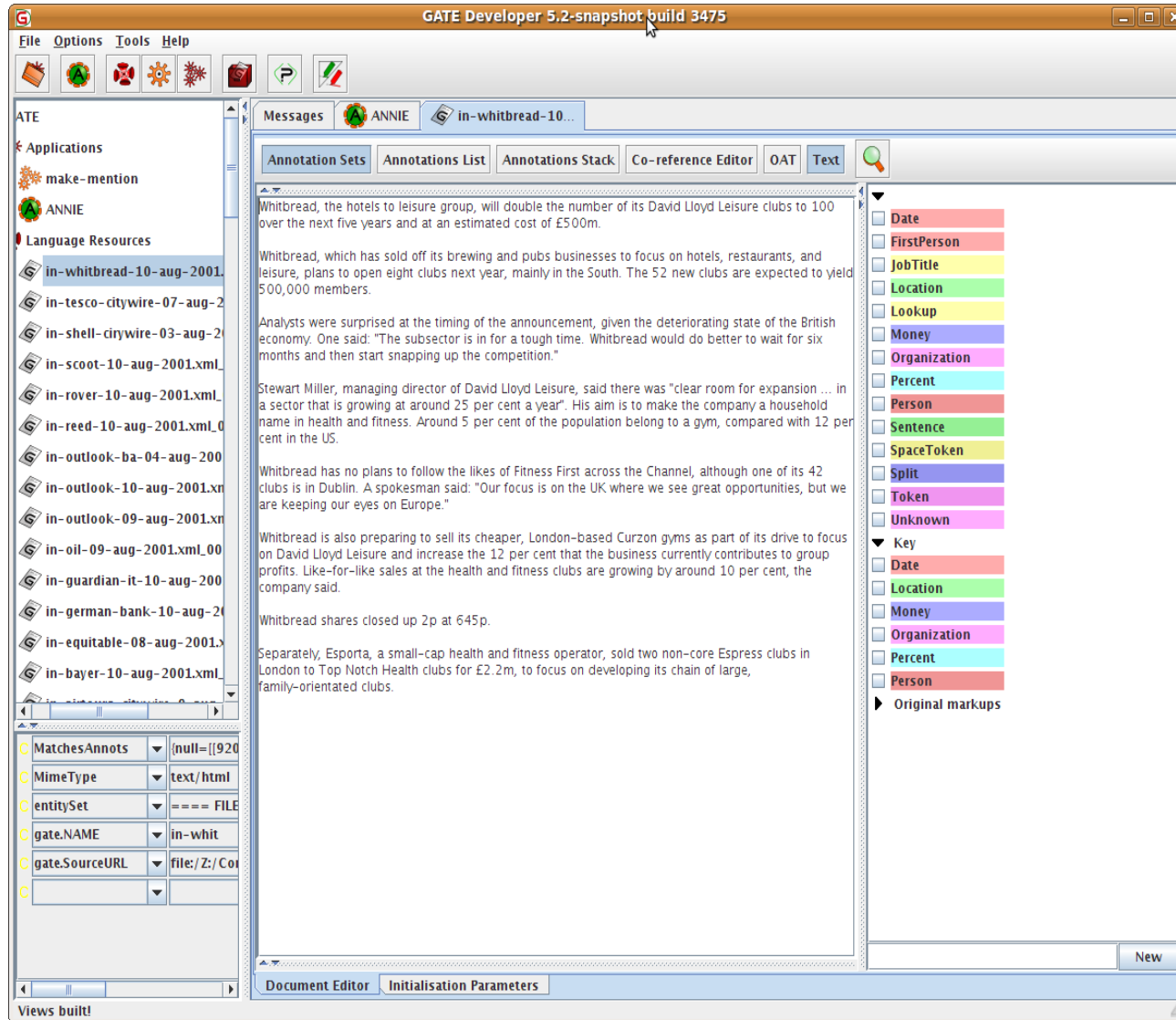
- We are going to train a machine learner to annotate corpora with these entity types
- We need a training corpus and a test corpus
- The training corpus will be used by the machine learner to deduce relationships between attributes and entity types (classes)
- The test corpus will be used to find out how well it is working, by comparing annotations created by the learner with the correct annotations that are already there
- In *Evaluation* mode, which we will try first, the ML PR automatically splits one corpus up into training and test sets



Instances and Attributes

- This corpus so far contains only the class annotations
- There is not much in this corpus to learn from
- What would our instances be?
- What would our attributes be?
- If we run ANNIE over the corpus, then we can use “Token” annotations for instances, and we would have various options for attributes
- **Load ANNIE**
- **Check that the document reset PR's setsToKeep parameter includes “Key”!**
- **Run ANNIE over your corpus**

Running ANNIE on the corpus

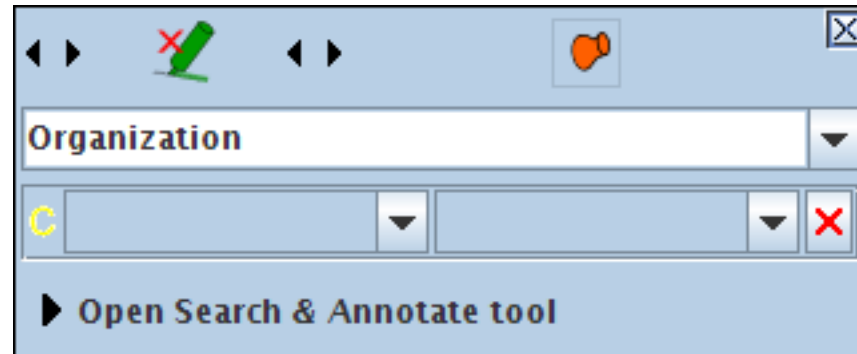


The screenshot shows the GATE Developer 5.2-snapshot build 3475 interface. The main window displays a document titled "in-whitbread-10..." with several paragraphs of text. The text is annotated with various entities and relations. The interface includes a menu bar (File, Options, Tools, Help), a toolbar, and a sidebar with a tree view of applications and language resources. The main window has tabs for "Messages", "ANNIE", and "in-whitbread-10...". Below the text, there are tabs for "Annotation Sets", "Annotations List", "Annotations Stack", "Co-reference Editor", "OAT", and "Text". A list of annotation types is visible on the right side of the main window, including Date, FirstPerson, JobTitle, Location, Lookup, Money, Organization, Percent, Person, Sentence, SpaceToken, Split, Token, Unknown, and Key. The bottom of the window shows a "Document Editor" and "Initialisation Parameters" tab.

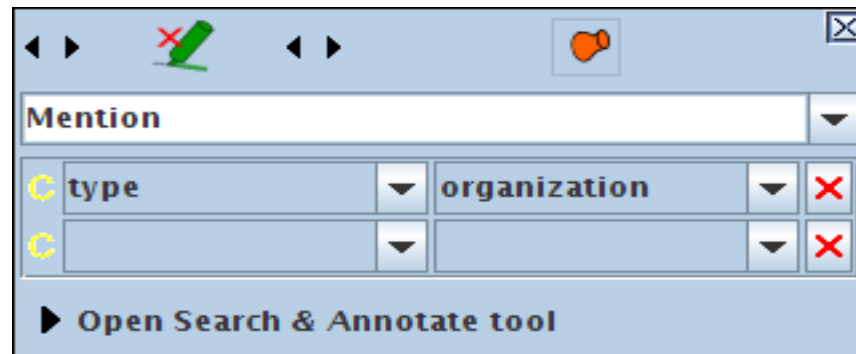
- Having run ANNIE on the corpus, we have more annotations to work with

Preparing the corpus: Classes

- What we have:



- What we need:





Preparing the corpus: Classes

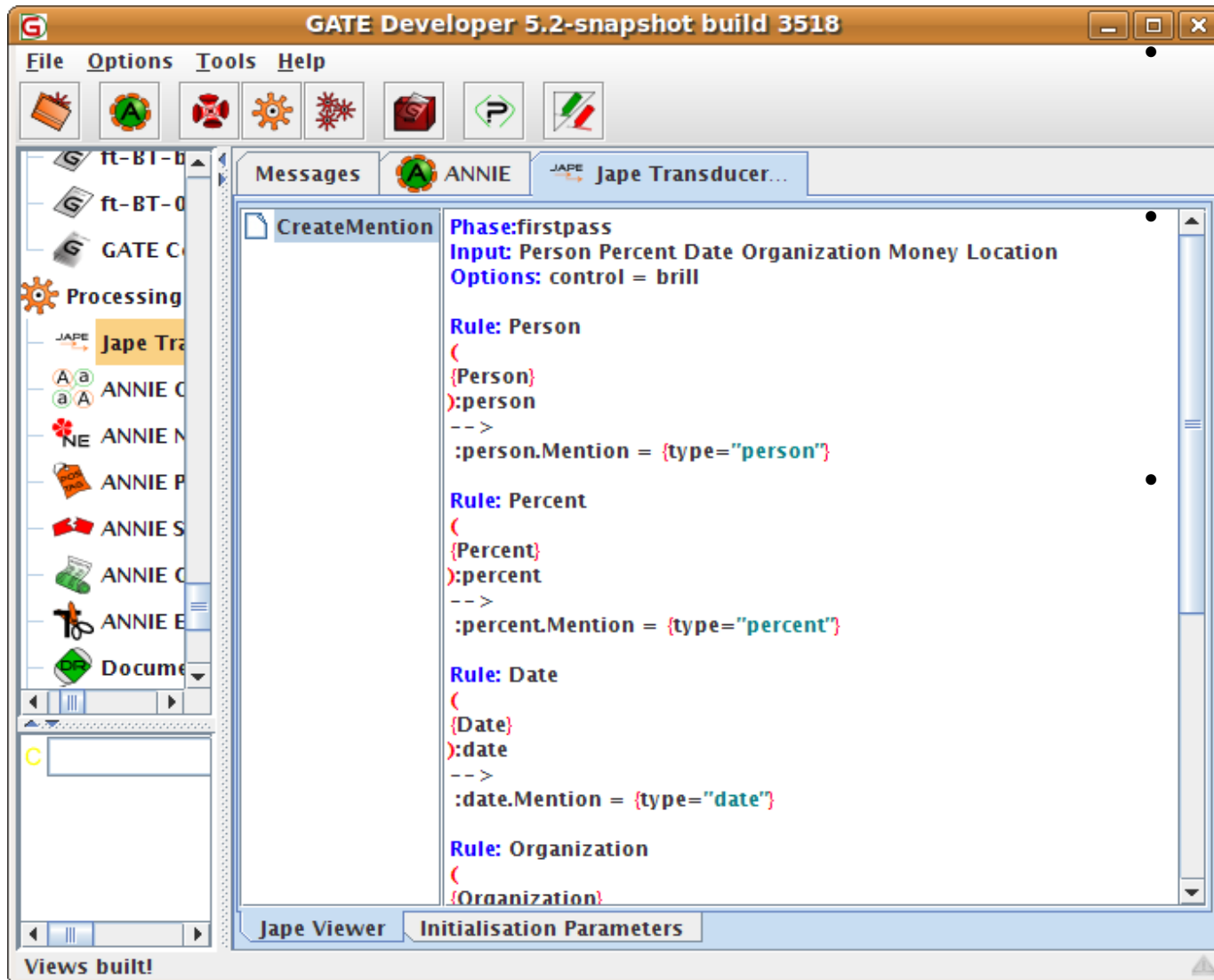
- Currently each class has its own annotation type (Date, Person, Percent etc.)
- But the ML PR expects the class (ML term) to be a feature value, not an annotation type
- So we need to make a new annotation type for the ML to learn from: “Mention” (it doesn't matter what it's called as long as we're consistent and configure the PR to match)



Making class annotations

- **Load a JAPE transducer from the ner/CreateMention.jape grammar**
- **Look at the grammar in GATE**

The CreateMention.jape grammar



The screenshot shows the GATE Developer 5.2 interface. The main window displays the 'CreateMention.jape' grammar in the 'Jape Viewer' tab. The grammar is defined as follows:

```

Phase: firstpass
Input: Person Percent Date Organization Money Location
Options: control = brill

Rule: Person
(
  {Person}
):person
-->
:person.Mention = {type="person"}

Rule: Percent
(
  {Percent}
):percent
-->
:percent.Mention = {type="percent"}

Rule: Date
(
  {Date}
):date
-->
:date.Mention = {type="date"}

Rule: Organization
(
  {Organization}

```

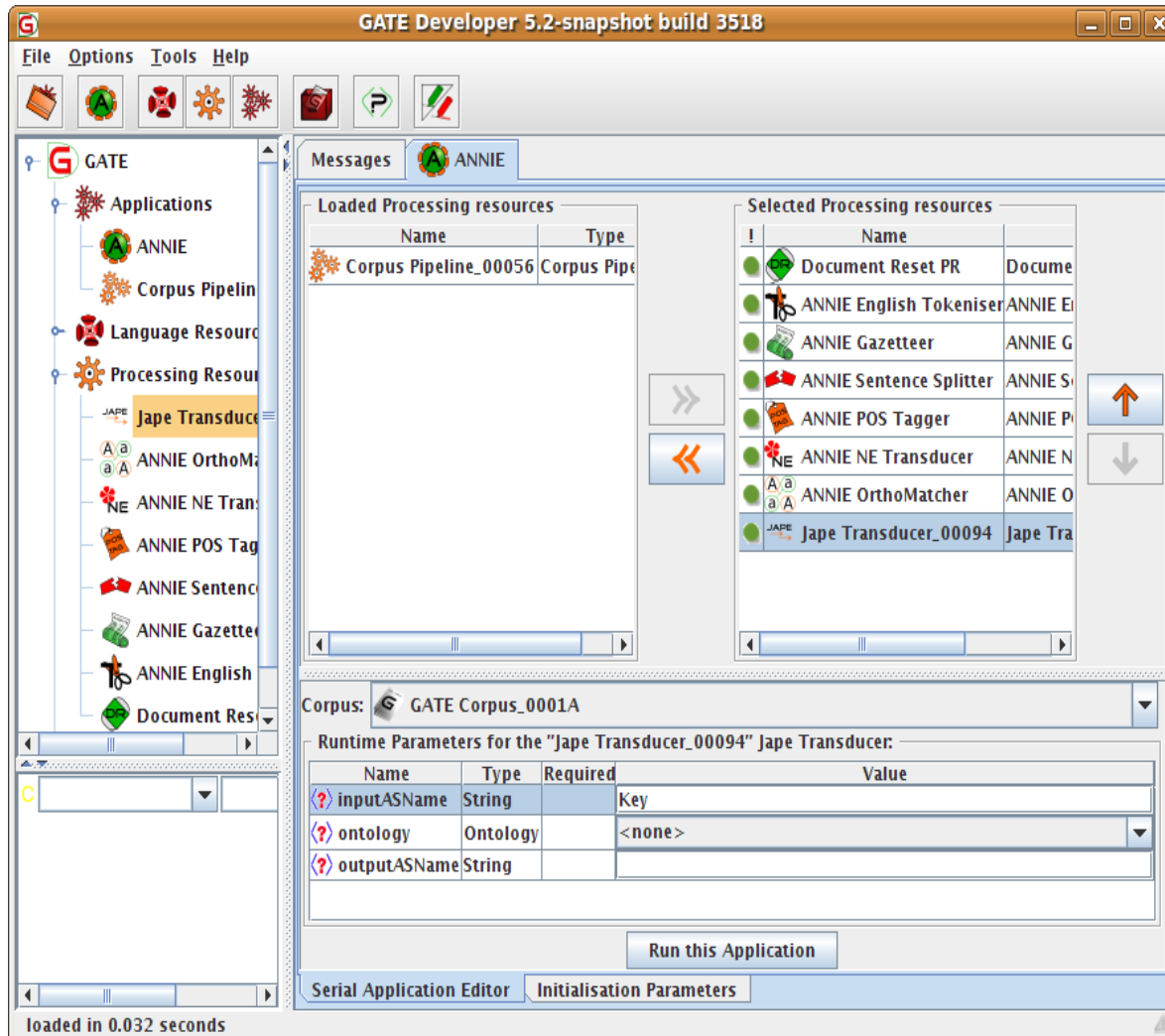
The interface also shows a 'Messages' tab with 'ANNIE' and 'Jape Transducer...' tabs. The bottom status bar indicates 'Views built!'.

This grammar makes a new annotation type called "Mention"

It makes the previous annotation type into a feature of the "Mention" annotation

- Feature name is "type" because "class" is reserved for ontology use

Applying the grammar to the corpus



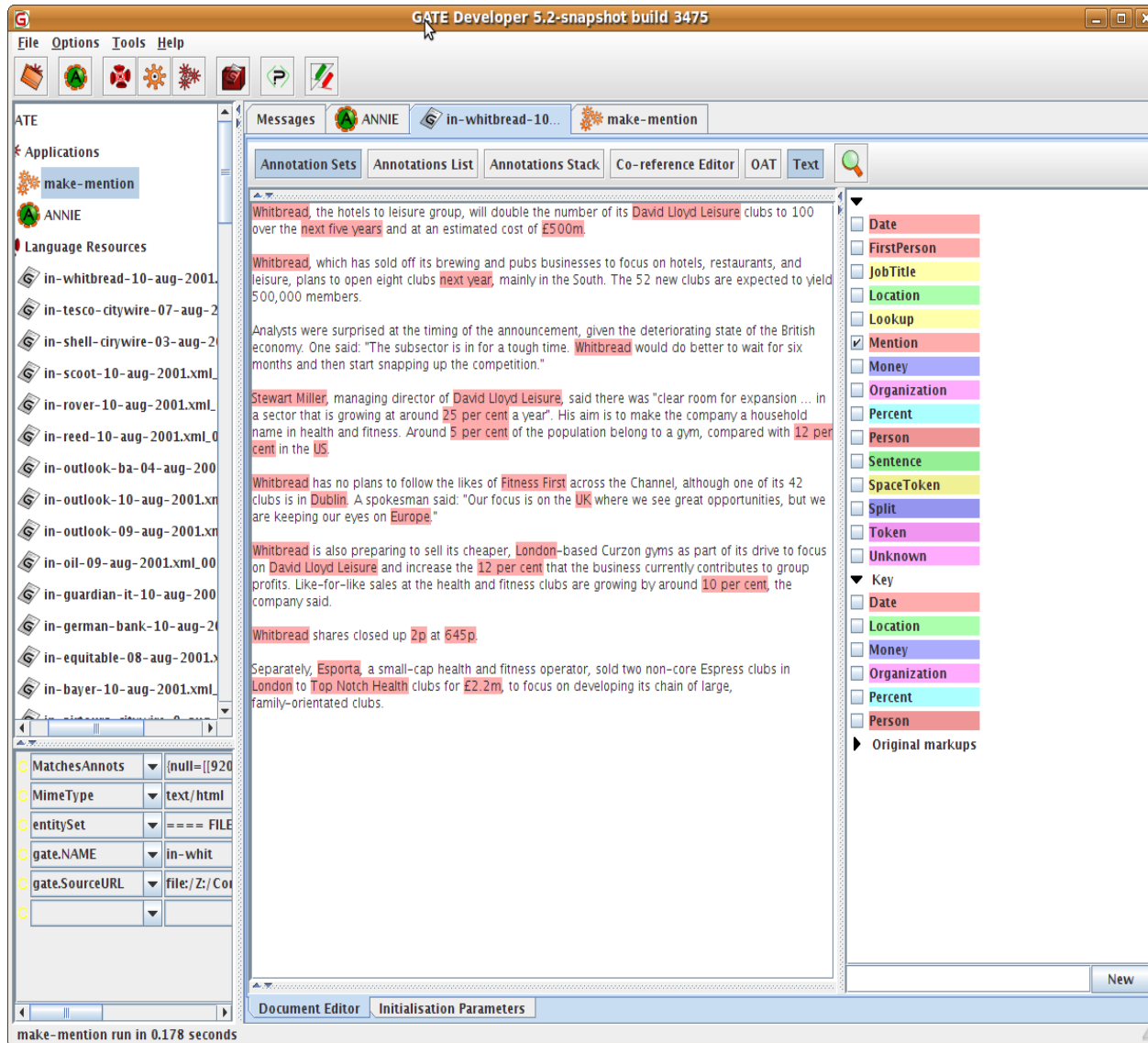
The screenshot shows the GATE Developer interface. On the left, a tree view shows the application structure under 'ANNIE', with 'Jape Transducer' selected. The main window displays the 'Messages' tab for the 'ANNIE' application, showing 'Loaded Processing resources' and 'Selected Processing resources'. The 'Selected Processing resources' list includes 'Jape Transducer_00094'. Below this, the 'Runtime Parameters for the "Jape Transducer_00094" Jape Transducer:' are shown in a table.

Name	Type	Required	Value
inputASName	String	Key	
ontology	Ontology	<none>	
outputASName	String		

At the bottom, there is a 'Run this Application' button and a 'Serial Application Editor' tab. The status bar at the bottom left indicates 'loaded in 0.032 seconds'.

- Add the JAPE transducer at the end of your ANNIE application
- Set the inputASName to “Key”
- Leave the outputASName blank (default)

Check the “Mention” annotations



GATE Developer 5.2-snapshot build 3475

File Options Tools Help

ATE

Applications

- make-mention
- ANNIE

Language Resources

- in-whitbread-10-aug-2001.xml
- in-tesco-citywire-07-aug-2001.xml
- in-shell-citywire-03-aug-2001.xml
- in-scoot-10-aug-2001.xml
- in-rover-10-aug-2001.xml
- in-reed-10-aug-2001.xml
- in-outlook-ba-04-aug-2001.xml
- in-outlook-10-aug-2001.xml
- in-outlook-09-aug-2001.xml
- in-oil-09-aug-2001.xml
- in-guardian-it-10-aug-2001.xml
- in-german-bank-10-aug-2001.xml
- in-equitable-08-aug-2001.xml
- in-bayer-10-aug-2001.xml

Messages

ANNIE in-whitbread-10-aug-2001.xml make-mention

Annotation Sets Annotations List Annotations Stack Co-reference Editor OAT Text

Whitbread, the hotels to leisure group, will double the number of its David Lloyd Leisure clubs to 100 over the next five years and at an estimated cost of £500m.

Whitbread, which has sold off its brewing and pubs businesses to focus on hotels, restaurants, and leisure, plans to open eight clubs next year, mainly in the South. The 52 new clubs are expected to yield 500,000 members.

Analysts were surprised at the timing of the announcement, given the deteriorating state of the British economy. One said: "The subsector is in for a tough time. Whitbread would do better to wait for six months and then start snapping up the competition."

Stewart Miller, managing director of David Lloyd Leisure, said there was "clear room for expansion ... in a sector that is growing at around 25 per cent a year". His aim is to make the company a household name in health and fitness. Around 5 per cent of the population belong to a gym, compared with 12 per cent in the US.

Whitbread has no plans to follow the likes of Fitness First across the Channel, although one of its 42 clubs is in Dublin. A spokesman said: "Our focus is on the UK where we see great opportunities, but we are keeping our eyes on Europe."

Whitbread is also preparing to sell its cheaper, London-based Curzon gyms as part of its drive to focus on David Lloyd Leisure and increase the 12 per cent that the business currently contributes to group profits. Like-for-like sales at the health and fitness clubs are growing by around 10 per cent, the company said.

Whitbread shares closed up 2p at 645p.

Separately, Esports, a small-cap health and fitness operator, sold two non-core Espress clubs in London to Top Notch Health clubs for £2.2m, to focus on developing its chain of large, family-orientated clubs.

Annotations List

- Date
- FirstPerson
- JobTitle
- Location
- Lookup
- Mention
- Money
- Organization
- Percent
- Person
- Sentence
- SpaceToken
- Split
- Token
- Unknown
- Key
 - Date
 - Location
 - Money
 - Organization
 - Percent
 - Person
- Original markups

Document Editor Initialisation Parameters

make-mention run in 0.178 seconds

- Rerun the application
- Check that you have some “Mention” annotations
- Check that they have a feature “type” and that the values look right



Check the “Mention” annotations

- **Check that you have some “Mention” annotations**
- **Check that they have a feature “type” and that the values look right**
- **These Mention annotations are derived from the named entities in the “Key” AS, and may not always match the NEs (Person, Location, etc.) generated by ANNIE in the default AS.**



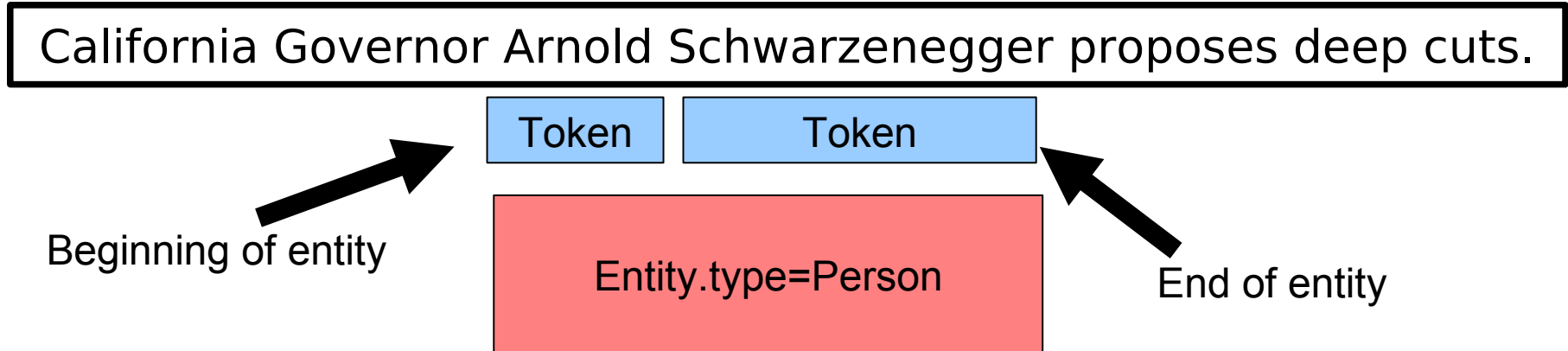
The Configuration File



Looking at the configuration file

- In the configuration file, we tell the machine learning PR what we want it to do
- You will find a configuration file in your hands-on materials, called ner/ner-config-file.xml
- **Open it using a text editor**

<SURROUND value="true"/>



- The class to be learned covers more than one instance (chunking problem)
- The PR has to learn the boundaries (chunking problem)
- So we tell the PR to use *surround mode*



Confidence Thresholds

```
<PARAMETER name="thresholdProbabilityEntity" value="0.2"/>
```

```
<PARAMETER name="thresholdProbabilityBoundary" value="0.4"/>
```

- Classifiers provide confidence ratings—how likely a result is to be correct
- We must determine how certain is good enough
- Depending on the application we might prefer to include or exclude annotations for which the learner is not too sure
- `thresholdProbabilityBoundary` is a threshold for the beginning and end instances
- `thresholdProbabilityEntity` is a threshold for beginning and end instances combined

<multiClassification2Binary method="one-vs-others"/>

California Governor Arnold Schwarzenegger proposes deep cuts.

Entity.type
=Location

Entity.type=Person

- Many algorithms are binary classifiers (e.g. yes/no)
- We have several classes (Person, Location, Organization etc.)
- Therefore the problem must be converted to a set of binary problems, so we can use binary algorithms
- **one-vs-others**
 - LOC vs PERS+ORG / PERS vs LOC+ORG / ORG vs LOC+PERS
- **one-vs-another**
 - LOC vs PERS / LOC vs ORG / PERS vs ORG

<multiClassification2Binary method="one-vs-others"/>



- With more than 3 classes, **one-vs-another** becomes very computationally expensive!
- **one-vs-others**: N classes => N classifiers
 - A vs B+C+D, B vs A+C+D, C vs A+B+D, D vs A+B+C
- **one-vs-another**: N classes => $N \times (N-1) / 2$ classifiers
 - A vs B, A vs C, A vs D, B vs C, B vs D, C vs D



```
<EVALUATION method="kfold" runs="4"/>
```

- We are going to evaluate our application in two ways today
 - The ML PR can automatically evaluate for us
 - We will also run our own evaluation
- This parameter dictates how the ML PR will work in evaluation mode
- The PR ignores this part of the config file in training and application modes

Evaluation

`<EVALUATION method="kfold" runs="4"/>`

OR

`<EVALUATION method="holdout" ratio="0.66"/>`

- Holdout randomly picks *ratio* documents for training and uses the rest for testing; this is faster than k-fold because it only runs once
- k-fold cross-validation will give you more reliable results and lets you “stretch” your corpus

K-Fold Cross-Validation

- In k-fold cross-validation, the corpus is split into k equal parts, and the learner is trained k times on k-1 parts and evaluated on 1; the results are averaged
- For example, if k=4, the documents are split into groups A, B, C, & D, then:
 - train on A+B+C, test on D;
 - train on A+B+D, test on C;
 - train on A+C+D, test on B;
 - train on B+C+D, test on A;
 - average these 4 results
- This maximises the use of the training data without losing testing accuracy, but takes 4 times as long
- `<EVALUATION method="kfold" runs="4"/>`



<ENGINE nickname="PAUM" ..

- Next we specify what machine learning algorithm we wish to use
- Today we are using the Perceptron with uneven margins ("PAUM")
- We will use the following options: options="-p 50 -n 5 -optB 0.3"
 - Challenge: find out what these options do! (Hint: user guide §18.2)



<INSTANCE-TYPE>...

- Next, we tell the ML PR what our instance annotation is
- The goal of the ML PR is to try to learn how the attributes of every instance relate to its class, so the instance is an important choice
- We have decided that the “Token” is our instance annotation type
 - We have run the tokenizer to ensure we have “Token annotations in our corpus
 - The POS tagger adds category features to the Token annotations

Specifying Attributes

```
<ATTRIBUTELIST>
  <NAME>Form</NAME>
  <SEMTYPE>NOMINAL</SEMTYPE>
  <TYPE>Token</TYPE>
  <FEATURE>category</FEATURE>
  <RANGE from="-2" to="2"/>
</ATTRIBUTELIST>
```

- For every attribute, we create a specification like the one above
- This is the information from which the PR will learn, so it is important to give it some good data
- You can see in the configuration file that there are several attributes (including Lookup.majorType), providing a good range of information
- However, if you have too many attributes it can take a very long time to learn!

Breaking down the attribute specification

- `<NAME>Form</NAME>`
 - This is the name that we choose for this attribute. It can be anything we want, but it will help us later if we make it something sensible!
- `<SEMTYPE>NOMINAL</SEMTYPE>`
 - Is the value of this attribute a number or a name? (Currently only nominal types are supported.)

Breaking down the attribute specification

- `<TYPE>Token</TYPE>`
 - The value of the attribute will be taken from the “Token” annotation
- `<FEATURE>category</FEATURE>`
 - The value of the attribute will be taken from the “category” feature

Breaking down the attribute specification

```
<ATTRIBUTELIST>  
:  
  <RANGE from="-2" to="2"/>  
</ATTRIBUTELIST>
```

- Because this is an “ATTRIBUTELIST” specification, we can specify a “RANGE”
- In this case, we will gather attributes from the current instance and also the preceding and following two; i.e., a window of 5 Token annotations centred on the one in question

Specifying the Class Attribute

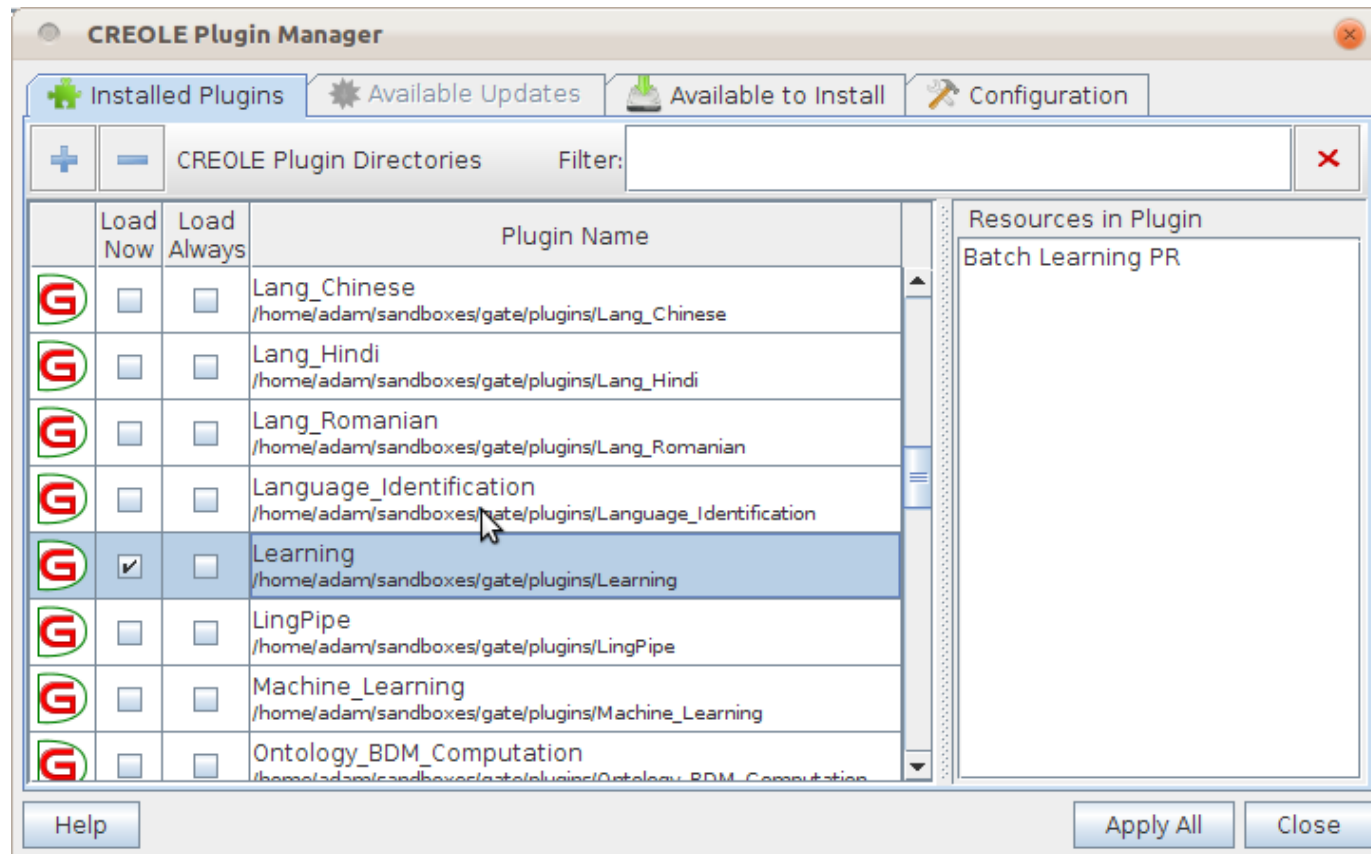
```
<ATTRIBUTE>
  <NAME>Class</NAME>
  <SEMTYPE>NOMINAL</SEMTYPE>
  <TYPE>Mention</TYPE>
  <FEATURE>type</FEATURE>
  <POSITION>0</POSITION>
  <CLASS/>
</ATTRIBUTE>
```

- You can call the class attribute whatever you want, but “Class” is a sensible choice
- Remember that our class attribute is the “type” feature of the “Mention” annotation
- This is an ATTRIBUTE, not an ATTRIBUTELIST, so we have “position”, not “range”
- The <CLASS/> element tells the Batch Learning PR that this is the class attribute to learn.



Running the ML PR in evaluation mode

Loading the Learning plugin



- **Load the “Learning” plugin**
- (We are **not** going to use the “Machine Learning” plugin, which is obsolete and does not have all the functionality we want.)



Creating a learning application

- **Create a “Batch Learning PR” with ner/ner-config.xml as the the configFileURL parameter**
- **Make a new corpus pipeline and put this PR (only!) in it**

Running the application in evaluation mode



GATE Developer 7.2-SNAPSHOT build 4620

File Options Tools Help

Messages ANNIE in-outlook-ba-0... Corpus Pipeline...

Loaded Processing resources

Name	
ANNIE English Tokeniser	ANNIE English T...
ANNIE Gazetteer	ANNIE Gazettee...
ANNIE NE Transducer	ANNIE NE Trans...
ANNIE OrthoMatcher	ANNIE OrthoMa...
ANNIE POS Tagger	ANNIE POS Tagg...
ANNIE Sentence Splitter	ANNIE Sentence...
Document Reset PR	Document Rese...
createMention	JAPE Transduce...

Selected Processing resources

Name
Batch Learning PR_000...

Corpus: GATE Corpus_00011

Runtime Parameters for the "Batch Learning PR_0008D" Batch Learning PR:

Name	Type	Required	
inputASName	String		
learningMode	RunMode	✓	EVALUATION
outputASName	String		
runProtocolDir	URL		

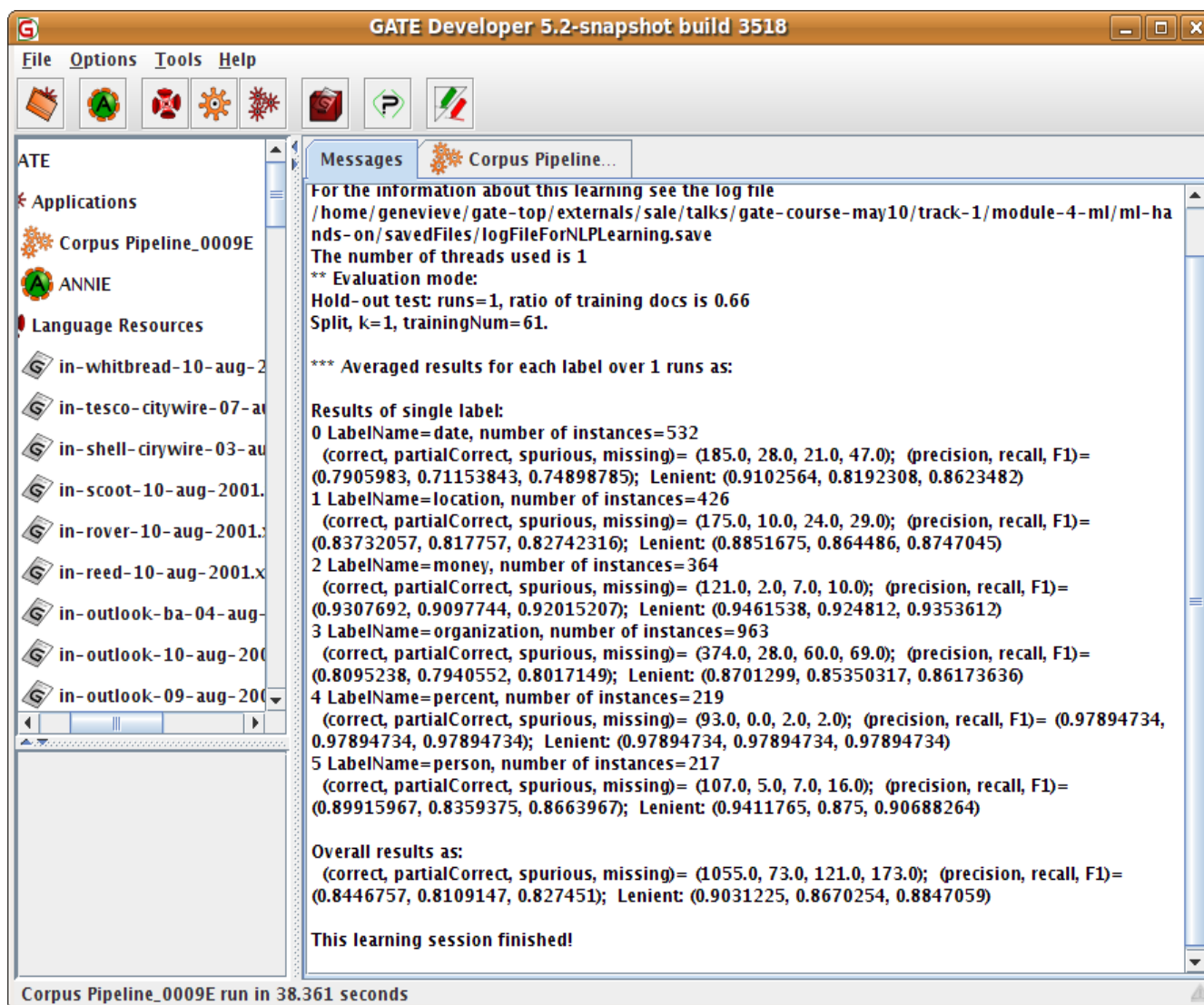
Run this Application

Serial Application Editor Initialisation Parameters

Corpus Pipeline_0008E run in 80.858 seconds

- **Make sure the corpus is selected**
- **The inputASName is blank because the attributes and classes are in the default annotation set**
- **Select "EVALUATION" for the learningMode**
- **OutputASName should be the same as inputASName in evaluation mode**
- **Run the application!**

Inspecting the results



GATE Developer 5.2-snapshot build 3518

File Options Tools Help

Messages Corpus Pipeline...

For the information about this learning see the log file
/home/genevieve/gate-top/externals/sale/talks/gate-course-may10/track-1/module-4-ml/ml-hands-on/savedFiles/logFileForNLPLearning.save
The number of threads used is 1
** Evaluation mode:
Hold-out test: runs=1, ratio of training docs is 0.66
Split, k=1, trainingNum=61.

*** Averaged results for each label over 1 runs as:

Results of single label:

0 LabelName=date, number of instances=532
(correct, partialCorrect, spurious, missing)= (185.0, 28.0, 21.0, 47.0); (precision, recall, F1)= (0.7905983, 0.71153843, 0.74898785); Lenient: (0.9102564, 0.8192308, 0.8623482)

1 LabelName=location, number of instances=426
(correct, partialCorrect, spurious, missing)= (175.0, 10.0, 24.0, 29.0); (precision, recall, F1)= (0.83732057, 0.817757, 0.82742316); Lenient: (0.8851675, 0.864486, 0.8747045)

2 LabelName=money, number of instances=364
(correct, partialCorrect, spurious, missing)= (121.0, 2.0, 7.0, 10.0); (precision, recall, F1)= (0.9307692, 0.9097744, 0.92015207); Lenient: (0.9461538, 0.924812, 0.9353612)

3 LabelName=organization, number of instances=963
(correct, partialCorrect, spurious, missing)= (374.0, 28.0, 60.0, 69.0); (precision, recall, F1)= (0.8095238, 0.7940552, 0.8017149); Lenient: (0.8701299, 0.85350317, 0.86173636)

4 LabelName=percent, number of instances=219
(correct, partialCorrect, spurious, missing)= (93.0, 0.0, 2.0, 2.0); (precision, recall, F1)= (0.97894734, 0.97894734, 0.97894734); Lenient: (0.97894734, 0.97894734, 0.97894734)

5 LabelName=person, number of instances=217
(correct, partialCorrect, spurious, missing)= (107.0, 5.0, 7.0, 16.0); (precision, recall, F1)= (0.89915967, 0.8359375, 0.8663967); Lenient: (0.9411765, 0.875, 0.90688264)

Overall results as:
(correct, partialCorrect, spurious, missing)= (1055.0, 73.0, 121.0, 173.0); (precision, recall, F1)= (0.8446757, 0.8109147, 0.827451); Lenient: (0.9031225, 0.8670254, 0.8847059)

This learning session finished!

Corpus Pipeline_0009E run in 38.361 seconds

- The application may take a few minutes to run
- **When it is finished, switch to the “Messages” tab to examine the results**

How well did we do?

- Here is my previous result:

**(precision, recall, F1) = (0.82714,
0.818435, 0.8225587)**

- These figures look pretty good, but what do they mean?
- Next we will discuss evaluation measures
- Then we will run the PR in different modes
- Then we will see if we can improve these numbers



Evaluation in Machine Learning



Recap of Evaluation in GATE

- Evaluation is an important part of information extraction work
 - We need to find out how good our application is by comparing its annotations to the “right answers” (manually prepared or corrected annotations)
 - Sometimes we need to compare the work of different human annotators, to see how consistent they are
- We use similar functions for both types of evaluation tasks



Evaluation Mode

- We ran the machine learning PR in evaluation mode earlier
- We specified how the PR should run evaluation in the configuration file
- Once we had run the application, we obtained evaluation statistics in the “Messages” tab



What are precision, recall and F1?

- Precision: what proportion of our automatic annotations were correct?
- Recall: what proportion of the correct annotations did our automatic tool create?
- $P = \text{correct} / (\text{correct} + \text{spurious}) = \text{tp} / (\text{tp} + \text{fp})$
- $R = \text{correct} / (\text{correct} + \text{missing}) = \text{tp} / (\text{tp} + \text{fn})$
- where tp = true positives, fp = false positives, fn = false negatives

What are precision, recall and F1?

- F-score is an amalgam of the two measures
 - $F = 1 / (\beta/P + (1-\beta)/R)$
 - The equally balanced F1 ($\beta = 0.5$) is the most common F-measure
 - $F1 = 2PR / (R + P)$
- We can also run our own ML evaluation using the Corpus QA tool—let's do that now

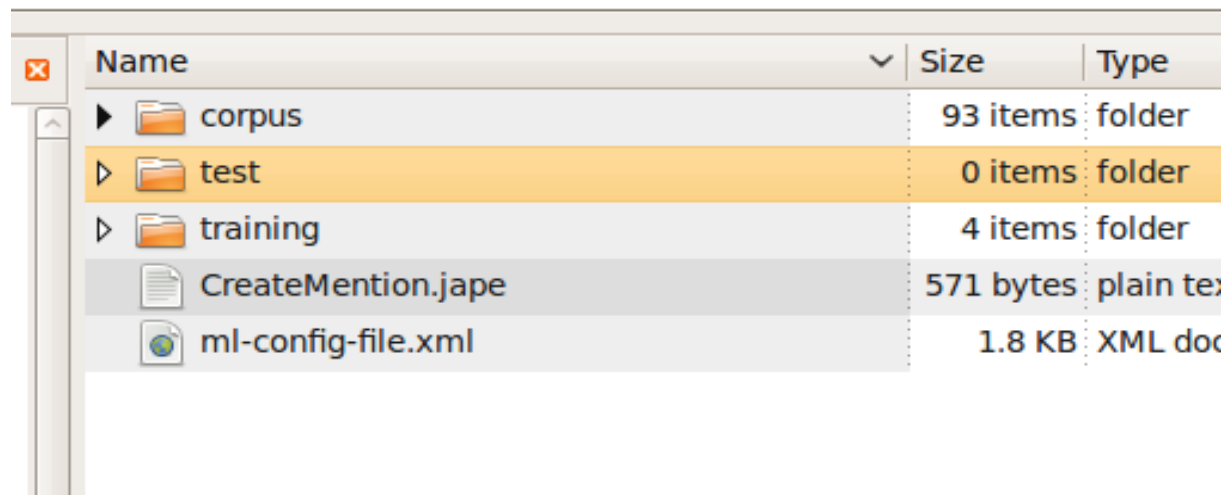


Splitting into training and test corpora

- To tell how well a machine learner is performing, you need to train it and test it on different sets of data
- Evaluation mode does this automatically over “folds” of the corpus
- To see a detailed evaluation, we need to split our corpus into two parts: the training corpus and the test corpus; we will train and apply in separate runs

Saving and splitting the corpus

nl-hands-on

A screenshot of a file manager window titled "nl-hands-on". The window displays a list of files and folders. The "test" folder is highlighted in yellow. The list includes folders for "corpus", "test", and "training", and files "CreateMention.jape" and "ml-config-file.xml".

Name	Size	Type
▶ corpus	93 items	folder
▶ test	0 items	folder
▶ training	4 items	folder
📄 CreateMention.jape	571 bytes	plain text
🌐 ml-config-file.xml	1.8 KB	XML document

- **Create new “training” and “test” directories on your computer (somewhere easy to find)**
- **Right click on your corpus, select “Save as XML”, and save the whole corpus in the “training” directory**
- **Use your file manager to move roughly half the documents from “training” into “test” (try to randomise them a little)**



Tidying up

- **Do not close the Batch Learning PR and its corpus pipeline! (We are going to keep using them.)**
- **Close all your open documents and corpora in GATE Developer**
- **Close the modified ANNIE application recursively**
- **Create new GATE corpora called “training” and “test”**
- **Populate each corpus from the appropriate directory (as before, set the encoding to UTF-8!)**

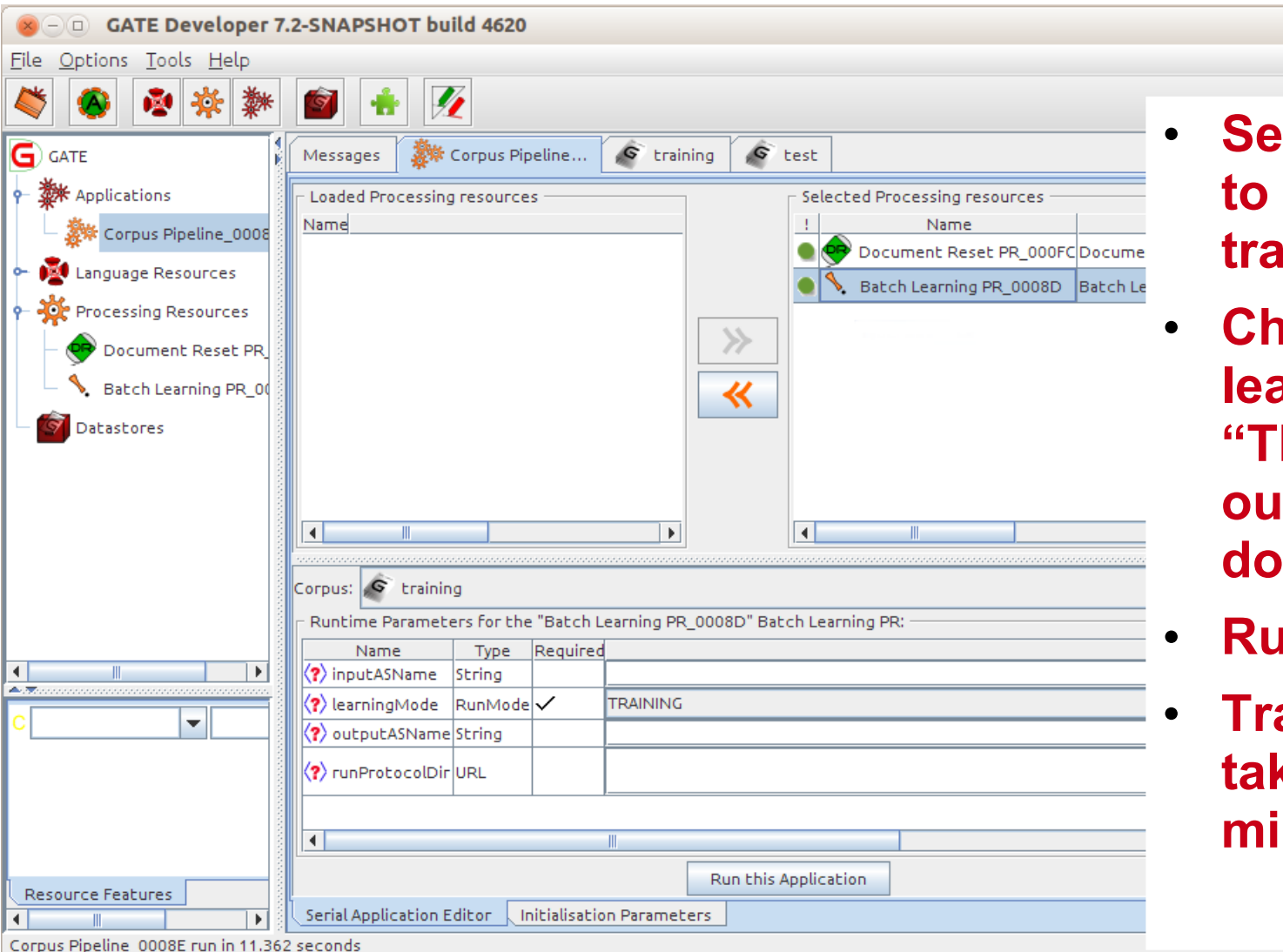


Setting up the application

- **Create a Document Reset PR**
- **Add it to the ML pipeline before the Batch Learning PR**
- **Edit the Document Reset PR's setsToRemove parameter to include just “ML”**
- **Edit the setsToKeep parameter to be an empty list**

Running the ML PR in Training Mode

- **Set your pipeline to run on the training corpus**
- **Change the PR's learningMode to "TRAINING" (the outputASName doesn't matter)**
- **Run the pipeline**
- **Training may take a few minutes**

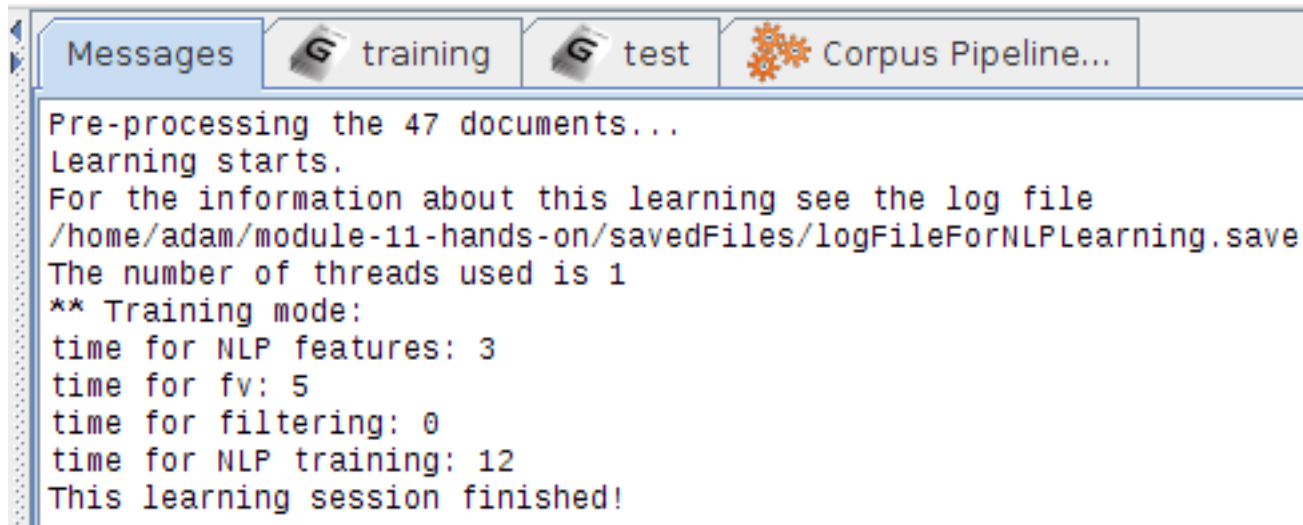


The screenshot shows the GATE Developer interface. The left sidebar displays a tree view with 'Applications' expanded to 'Corpus Pipeline_0008E'. The main window shows the 'Corpus Pipeline...' configuration. The 'Loaded Processing resources' list is empty, and the 'Selected Processing resources' list contains 'Document Reset PR_000FC' and 'Batch Learning PR_0008D'. The 'Corpus' dropdown is set to 'training'. Below, the 'Runtime Parameters for the "Batch Learning PR_0008D" Batch Learning PR:' table is shown.

Name	Type	Required	Value
inputASName	String		
learningMode	RunMode	✓	TRAINING
outputASName	String		
runProtocolDir	URL		

At the bottom, there is a 'Run this Application' button and a 'Serial Application Editor' window showing 'Initialisation Parameters'. The status bar at the bottom indicates 'Corpus Pipeline_0008E run in 11.362 seconds'.

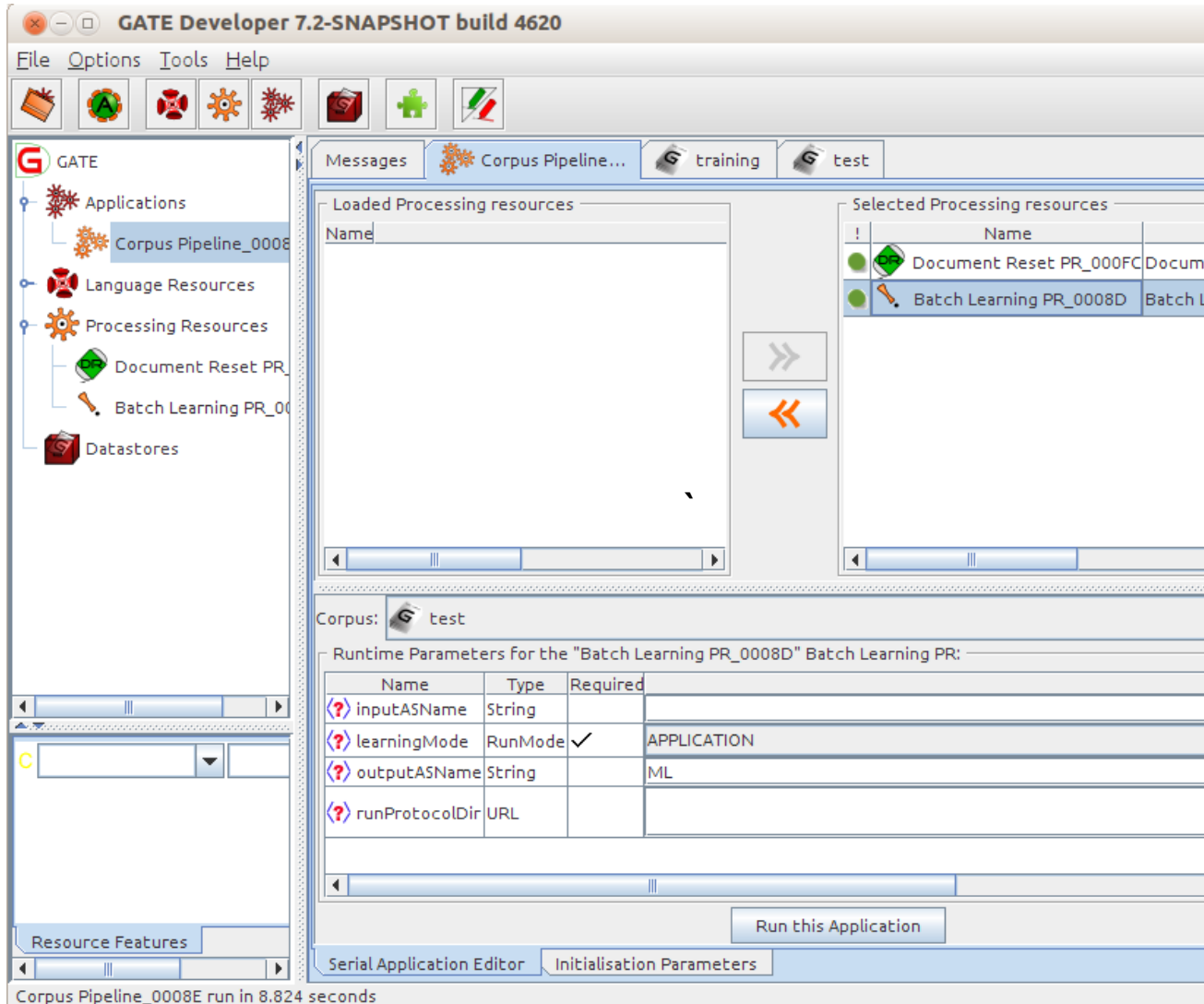
Finished Training!

A screenshot of the GATE software's Messages window. The window has a title bar with tabs for "Messages", "training", "test", and "Corpus Pipeline...". The "Messages" tab is active, showing a log of training progress. The text in the window reads: "Pre-processing the 47 documents... Learning starts. For the information about this learning see the log file /home/adam/module-11-hands-on/savedFiles/logFileForNLPlearning.save The number of threads used is 1 ** Training mode: time for NLP features: 3 time for fv: 5 time for filtering: 0 time for NLP training: 12 This learning session finished!"

```
Messages | G training | G test | Corpus Pipeline...
Pre-processing the 47 documents...
Learning starts.
For the information about this learning see the log file
/home/adam/module-11-hands-on/savedFiles/logFileForNLPlearning.save
The number of threads used is 1
** Training mode:
time for NLP features: 3
time for fv: 5
time for filtering: 0
time for NLP training: 12
This learning session finished!
```

- Because we saved the documents after running our modified ANNIE, they already have the instances and attributes for ML
- This time there are no evaluation results in the messages tab (because we are only training the model)
- Note the “savedFiles” directory beside the XML configuration file
 - Training mode saves the model there
 - Application mode reads it from there
 - The runProtocolDir parameter (in recent versions of GATE) can change the location

Running the ML PR in Application Mode



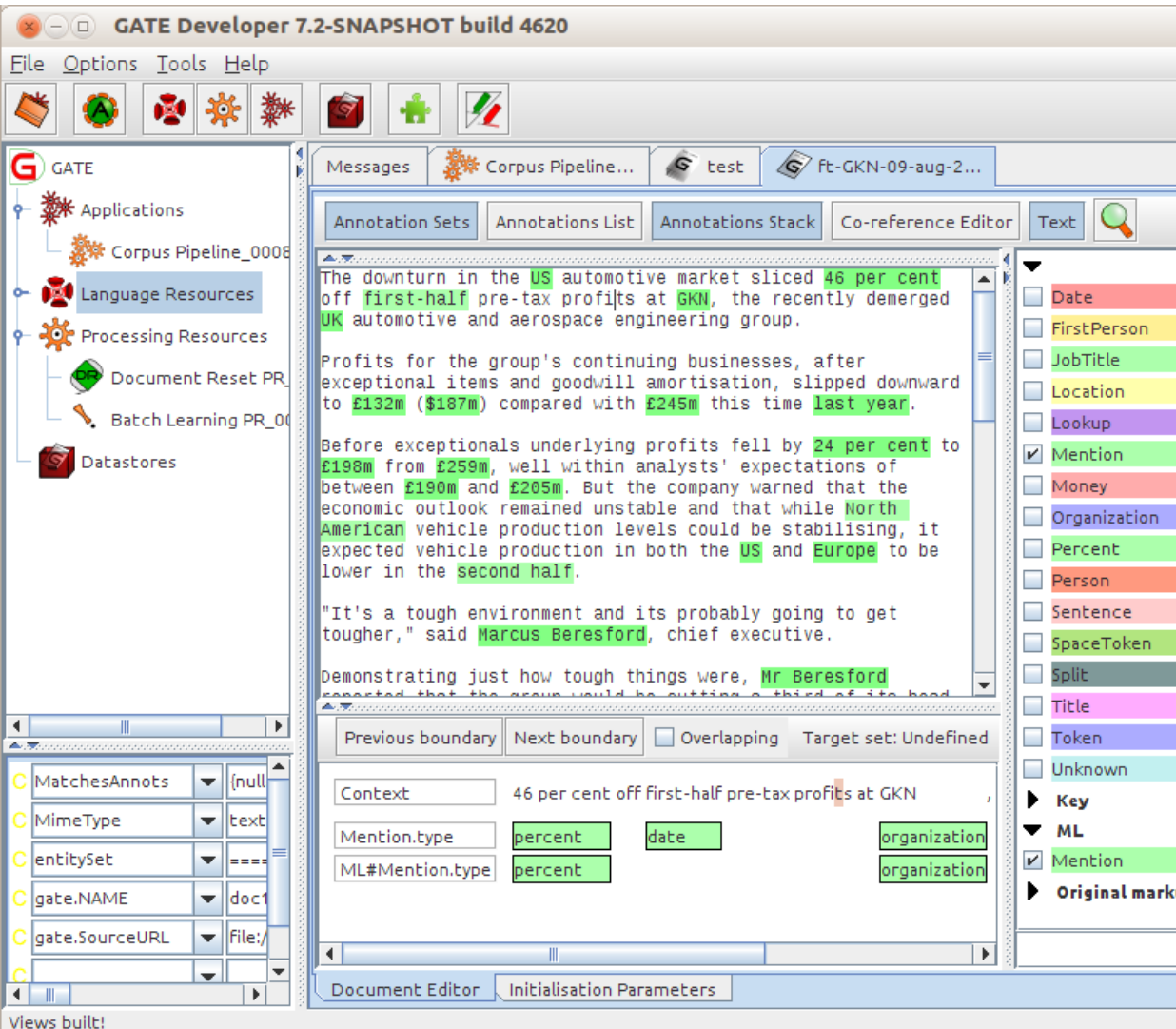
The screenshot shows the GATE Developer interface with the following configuration:

- Corpus:** test
- Runtime Parameters for the "Batch Learning PR_0008D" Batch Learning PR:**

Name	Type	Required	Value
inputASName	String		
learningMode	RunMode	<input checked="" type="checkbox"/>	APPLICATION
outputASName	String		ML
runProtocolDir	URL		
- Buttons:** Run this Application
- Bottom Bar:** Serial Application Editor, Initialisation Parameters
- Status Bar:** Corpus Pipeline_0008E run in 8.824 seconds

- **Change corpus to "test"**
- **Change learningMode to "APPLICATION"**
- **Set outputASName to "ML": your new (automatic) annotations will go here so they don't get mixed up with the existing ones**
- **Application mode is faster than training mode**

Examining the results of application



The screenshot shows the GATE Developer interface. The main window displays a document with several sentences. The text is annotated with various entities and mentions. The 'Annotations Stack' panel on the right shows a list of annotation sets, with 'Mention' selected. The 'Document Editor' panel at the bottom shows the current document content.

Document text:

The downturn in the US automotive market sliced 46 per cent off first-half pre-tax profits at GKN, the recently demerged UK automotive and aerospace engineering group.

Profits for the group's continuing businesses, after exceptional items and goodwill amortisation, slipped downward to £132m (\$187m) compared with £245m this time last year.

Before exceptionals underlying profits fell by 24 per cent to £198m from £259m, well within analysts' expectations of between £190m and £205m. But the company warned that the economic outlook remained unstable and that while North American vehicle production levels could be stabilising, it expected vehicle production in both the US and Europe to be lower in the second half.

"It's a tough environment and its probably going to get tougher," said Marcus Beresford, chief executive.

Demonstrating just how tough things were, Mr Beresford reported that the group would be cutting a third of its head

Annotations Stack:

- Date
- FirstPerson
- JobTitle
- Location
- Lookup
- Mention
- Money
- Organization
- Percent
- Person
- Sentence
- SpaceToken
- Split
- Title
- Token
- Unknown
- Key
- ML
 - Mention
 - Original marku

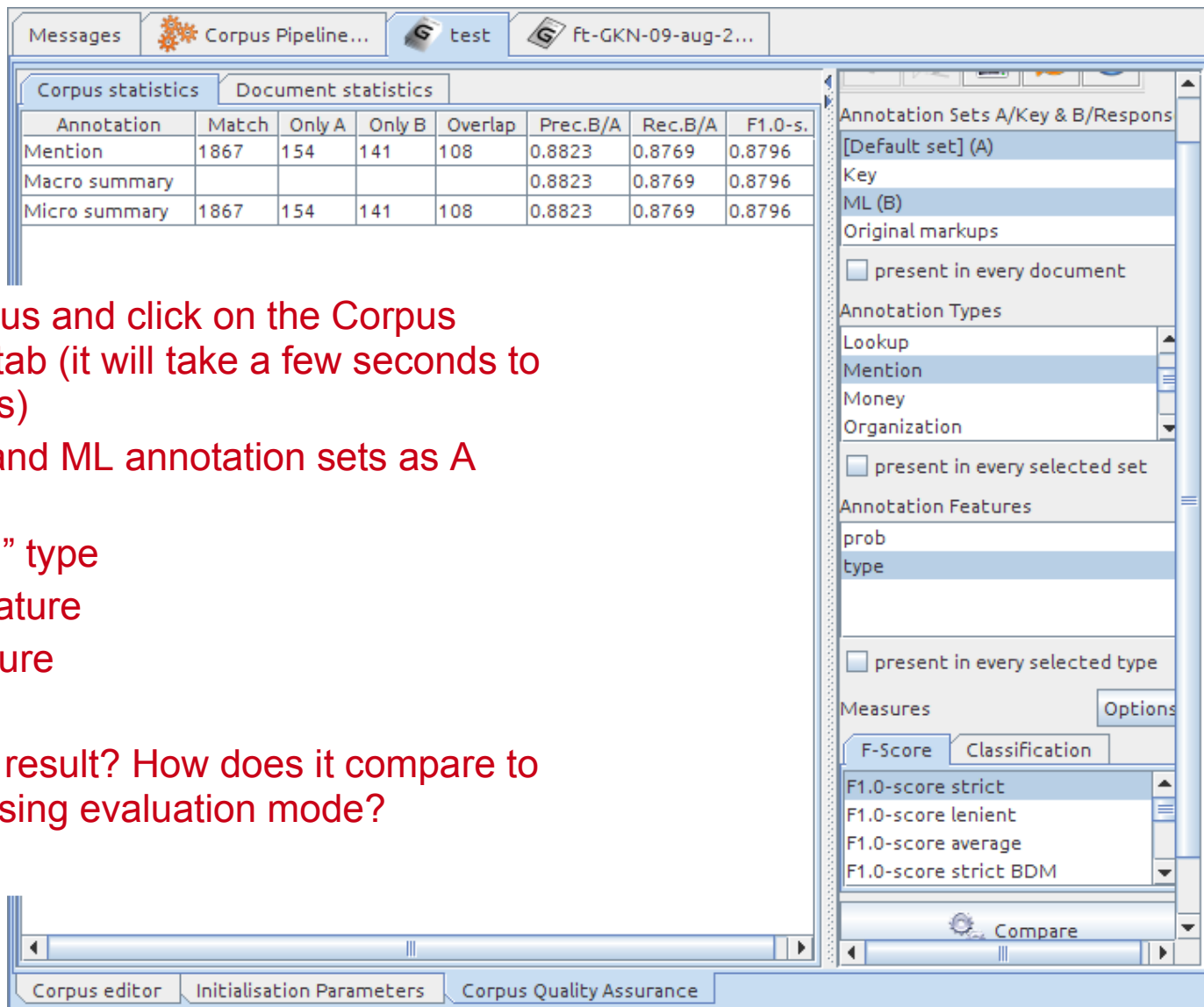
Document Editor:

Context: 46 per cent off first-half pre-tax profits at GKN

Mention.type	percent	date	organization
ML#Mention.type	percent		organization

- **Examine a document from the test corpus**
- You should have a new "ML" AS with Mention annotations
- The original Mention annotations (in the default AS) are similar but not always identical!
- The Annotations Stack is good for comparing them
- How similar do they appear to be? Do you think you will get a good result?

Comparing the Sets with Corpus QA



The screenshot shows the GATE Corpus QA interface. The main window displays a table of statistics for the 'test' corpus. The table has columns for Annotation, Match, Only A, Only B, Overlap, Prec.B/A, Rec.B/A, and F1.0-s. The rows are Mention, Macro summary, and Micro summary. The statistics are as follows:

Annotation	Match	Only A	Only B	Overlap	Prec.B/A	Rec.B/A	F1.0-s.
Mention	1867	154	141	108	0.8823	0.8769	0.8796
Macro summary					0.8823	0.8769	0.8796
Micro summary	1867	154	141	108	0.8823	0.8769	0.8796

The right-hand pane shows the configuration for the 'Annotation Sets A/Key & B/Respons' comparison. The 'Default set' (A) is selected, and the 'ML (B)' set is also selected. The 'Mention' annotation type is selected, and the 'type' feature is selected. The 'F-Score' measure is selected, and the 'F1.0-score strict' option is chosen. The 'Compare' button is visible at the bottom of the configuration pane.

- Select the test corpus and click on the Corpus Quality Assurance tab (it will take a few seconds to scan the documents)
- Select the Default and ML annotation sets as A and B, respectively
- Select the “Mention” type
- Select the “type” feature
- Choose an F-measure
- Click on Compare
- Did you get a good result? How does it compare to the result you got using evaluation mode?

Using Annotation Diff to examine performance

Annotation Difference

Key doc: ft-BT-briefing-02-a... Key set: [Default set] Type: Mention Weight: 1.0

Resp. doc: ft-BT-briefing-02-a... Resp. set: ML-results Features: all some none 1.0

Compare

Start	End	Key	Features	=?	Start	End	Response	Features
1517	1519	BT	{class=organization}	=	1517	1519	BT	{class=organization, prob=1.0}
171	173	2p	{class=money}	=	171	173	2p	{class=money, prob=1.0}
1956	1972	Deutsche · Telekom	{class=organization}	=	1956	1972	Deutsche · Telekom	{class=organization, prob=1.0}
46	55	yesterday	{class=date}	=	46	55	yesterday	{class=date, prob=1.0}
1322	1327	Oftel	{class=organization}	=	1322	1327	Oftel	{class=organization, prob=1.0}
867	882	January · 22 · 2001	{class=date}	=	867	882	January · 22 · 2001	{class=date, prob=1.0}
1198	1203	Scoot	{class=organization}	=	1198	1203	Scoot	{class=organization, prob=1.0}
514	524	Amazon.com	{class=organization}	~	514	520	Amazon	{class=organization, prob=1.0}
1753	1761	Scoot · UK	{class=organization}	-?				
1181	1195	late · last · year	{class=date}	-?				
1007	1017	Air · Canada	{class=organization}	-?				
1924	1926	DT	{class=organization}	-?				
				?-	1499	1511	0800 · 192 · 192	{class=money, prob=1.0}
482	488	Amazon	{class=organization}	<>	482	488	Amazon	{class=location, prob=0.99999946}
800	806	Amazon	{class=organization}	<>	800	806	Amazon	{class=location, prob=0.99999905}
756	762	Amazon	{class=organization}	<>	756	762	Amazon	{class=location, prob=1.0}

93 documents loaded

Correct: 36 Recall Precision F-measure

Partially correct: 1 Strict: 0.82 0.88 0.85


Missing: 7 Lenient: 0.84 0.90 0.87

False positives: 4 Average: 0.83 0.89 0.86

Statistics Adjudication

Show document

Export to HTML

- Switch to the “Document statistics” tab
- Choose a document
- Click on the Annotation Diff icon 
- What kind of mistakes did your application make?



Using Annotation Diff...

- “Correct”: the response annotation has the right feature and span
- “Partially correct”: response has the right feature and overlapping but not exactly matched span; this counts as correct in the “lenient” scoring
- “Missing”: key annotation+feature is missing from the response (a.k.a. “false negative”)
- “False positive”: response annotation+feature shouldn't be there (a.k.a. “spurious”)



Varying the configuration file

- Now we are going to experiment with varying the configuration file to see if we can produce varied results
- You can edit the configuration file in your favourite text editor
- Make sure you save your changes then **reinitialise the PR** (this reads the file again and updates the configuration used inside GATE)



Exercises

- **Spend some time working on your exercise sheet**
- **Feel free to ask questions**



Confidence Thresholds

```
<PARAMETER name="thresholdProbabilityEntity" value="0.2"/>  
<PARAMETER name="thresholdProbabilityBoundary" value="0.42"/>  
<PARAMETER name="thresholdProbabilityClassification" value="0.5"/>
```

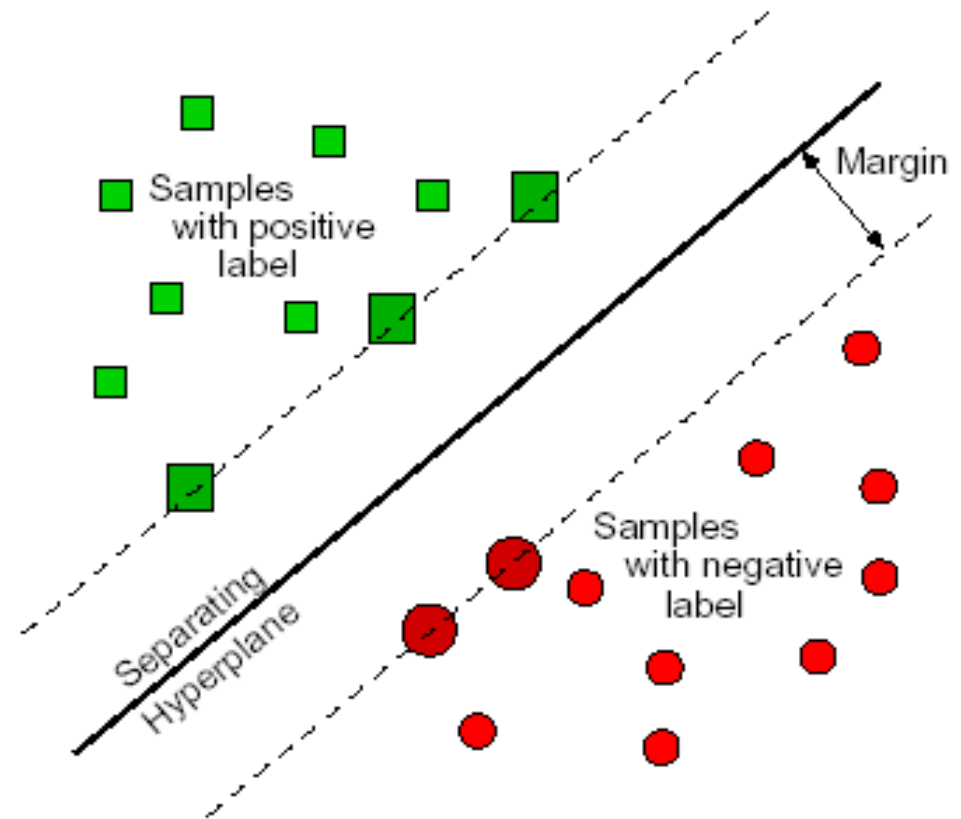
- Each classifier will provide confidence ratings—how likely is a result to be correct; we must determine how certain is good enough
- Depending on the application we might prefer to include or exclude annotations for which the learner is not too sure
- `thresholdProbabilityBoundary` and `thresholdProbabilityEntity` are thresholds for chunk learning
- `thresholdProbabilityClassification` applies to classification tasks, such as sentiment or genre detection, author identification, language identification



Engines and Algorithms

Support Vector Machines

- Attempt to find a hyperplane that separates data
- Goal: maximize margin separating two classes
- Wider margin = greater generalisation





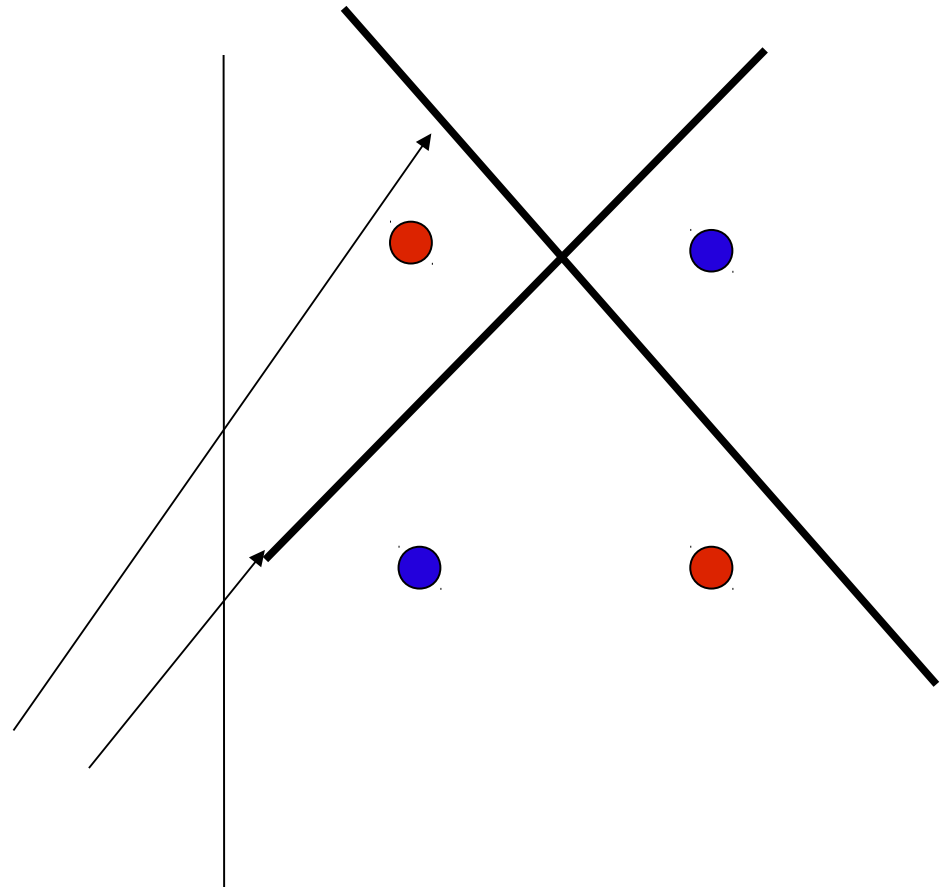
Support Vector Machines

- Points near decision boundary: support vectors (removing them would change boundary)
- Points far from boundary not important for decision
- What if data doesn't split?
 - Soft boundary methods exist for imperfect solutions
 - However linear separator may be completely unsuitable

Support Vector Machines

- What if there is no separating hyperplane?
- See example:
- Or class may be a globule

They do not work!



Kernel Trick

- Map data into different dimensionality
- <http://www.youtube.com/watch?v=3liCbRZPrZA>
- As shown in the video, due to polynomial kernel elliptical separators can be created nevertheless.
- Now the points are separable!





Kernel Trick in GATE and NLP

- Binomial kernel allows curved and elliptical separators to be created
- These are commonly used in language processing and are found to be successful
- Linear and polynomial kernels are implemented in Batch Learning PR's SVM



Support Vector Machines

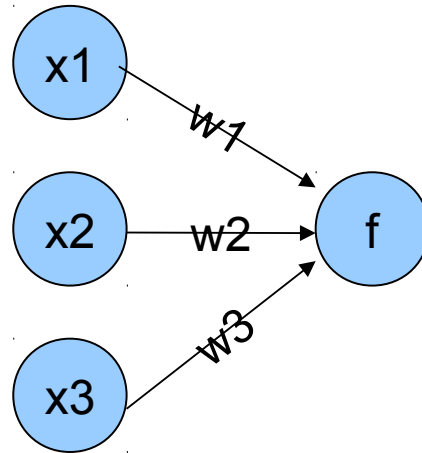
- SVMs combined with kernel trick provide a powerful technique
- Multiclass methods simple extension to two class technique (one vs. another, one vs. others)
- Widely used with great success across a range of linguistic tasks



Perceptron and PAUM

- Perceptron is one of the oldest ML methods (invented in the 50s)
- Like SVM, it determines a hyperplane separator between the data points
- Theoretically SVM works better because it calculates the optimal separator, but in practice, however, there is usually little difference, and Perceptron is a lot faster

Perceptron



- You might think of perceptrons as being these things (correct)
- What this is actually calculating is a dot product $w \cdot x$

More perceptron

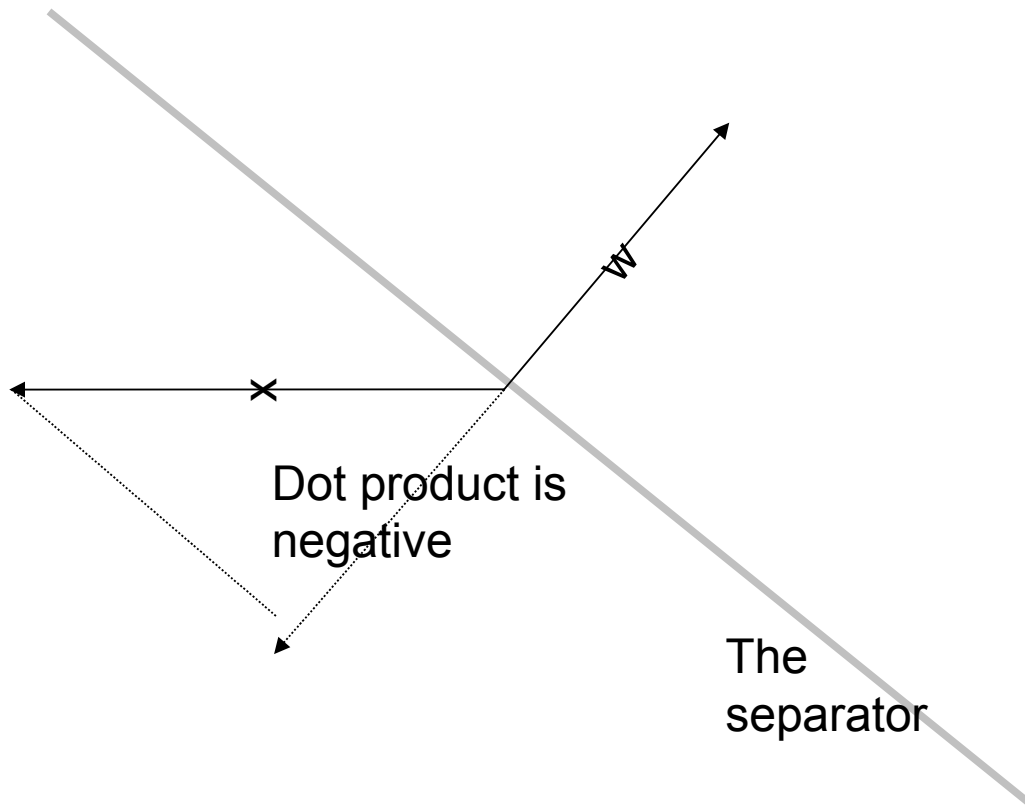
$$f(x) = \begin{cases} 1 & \text{if } \mathbf{w} \cdot \mathbf{x} + b > 0 \\ 0 & \text{otherwise} \end{cases}$$

- x is a datapoint represented as a vector
- w is a vector that defines the separating hyperplane (it is perpendicular to it)
- This function tells you which side of the hyperplane your point lies
- b defines an offset from the origin

More perceptron

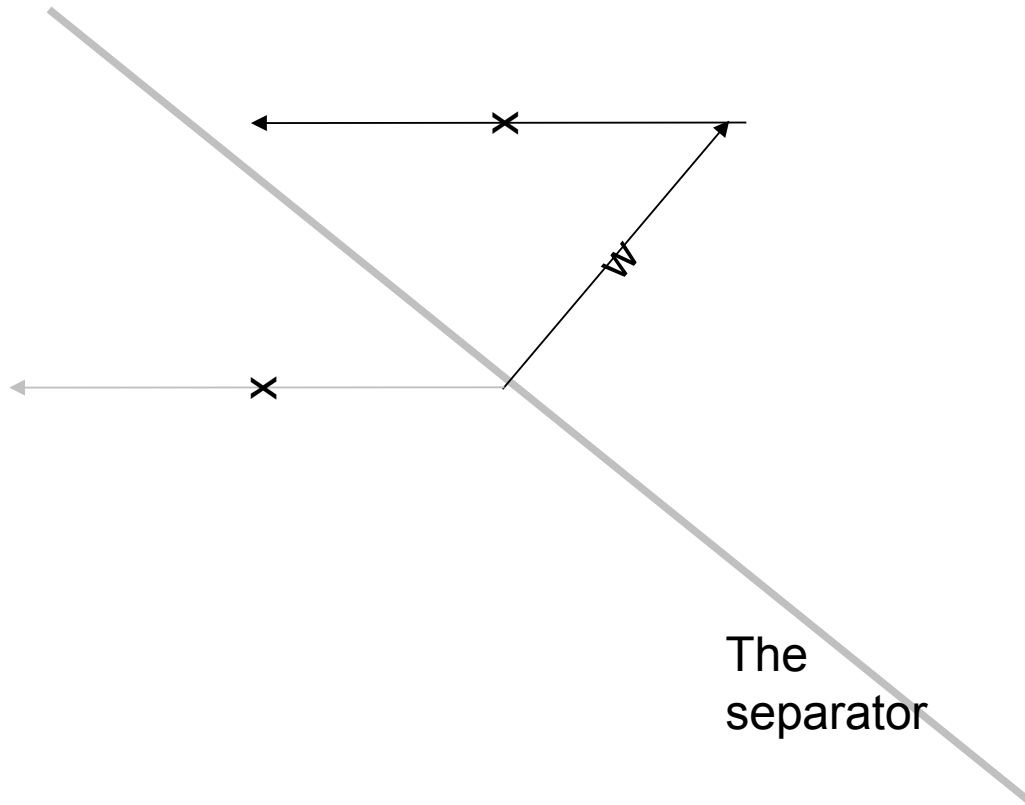
- How does it learn?
 - Each datapoint is annotated with class value 1 or 0
 - Function returns 1 or 0 depending on which side of the separator the point lies
 - Calculate difference between actual and desired output
 - Multiply input vector by this delta and add it to the weight vector
 - Given sufficient iterations the separator will find a solution

Perceptron update



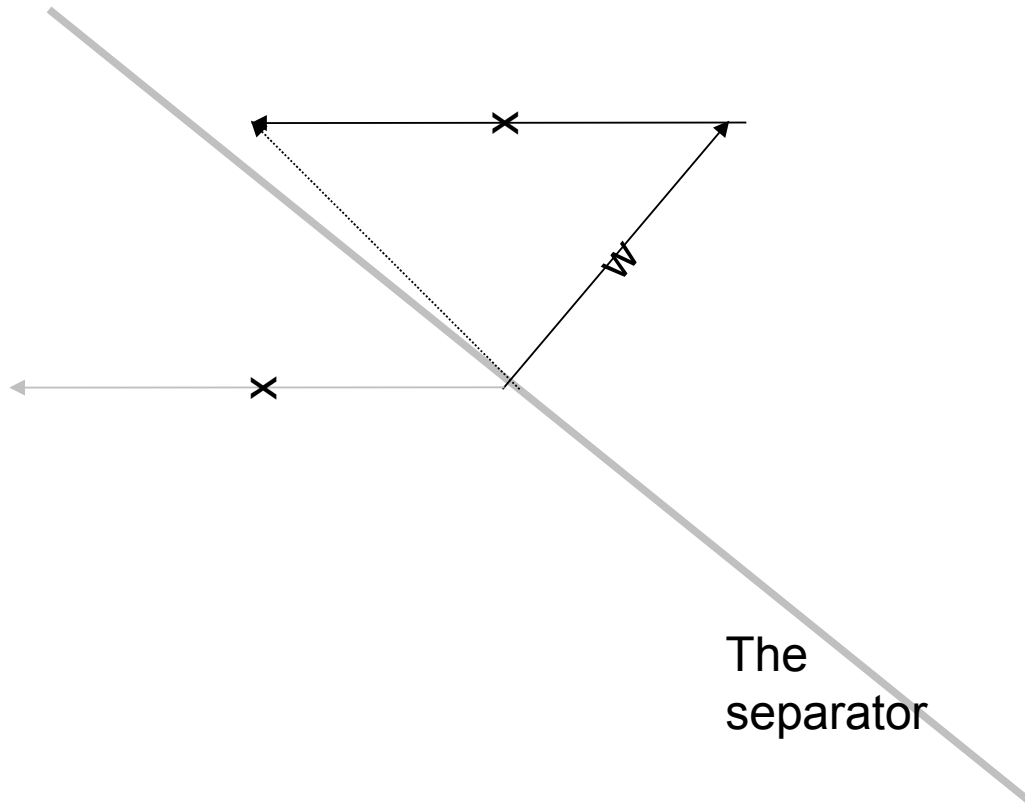
- Dot product is negative, so $f=0$
- But x is a positive example!
- Oh no! Must update

Perceptron update



- x class is 1
- $f(x) = 0$
- $w += (1-0)x$

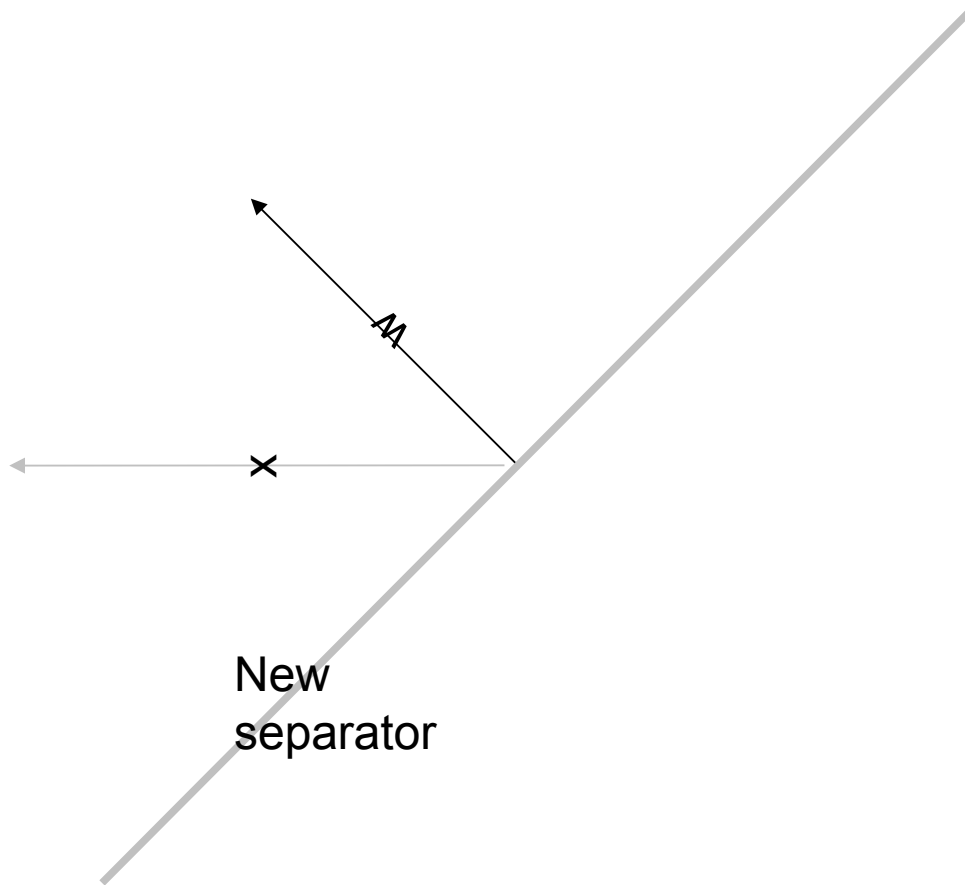
Perceptron update



The
separator

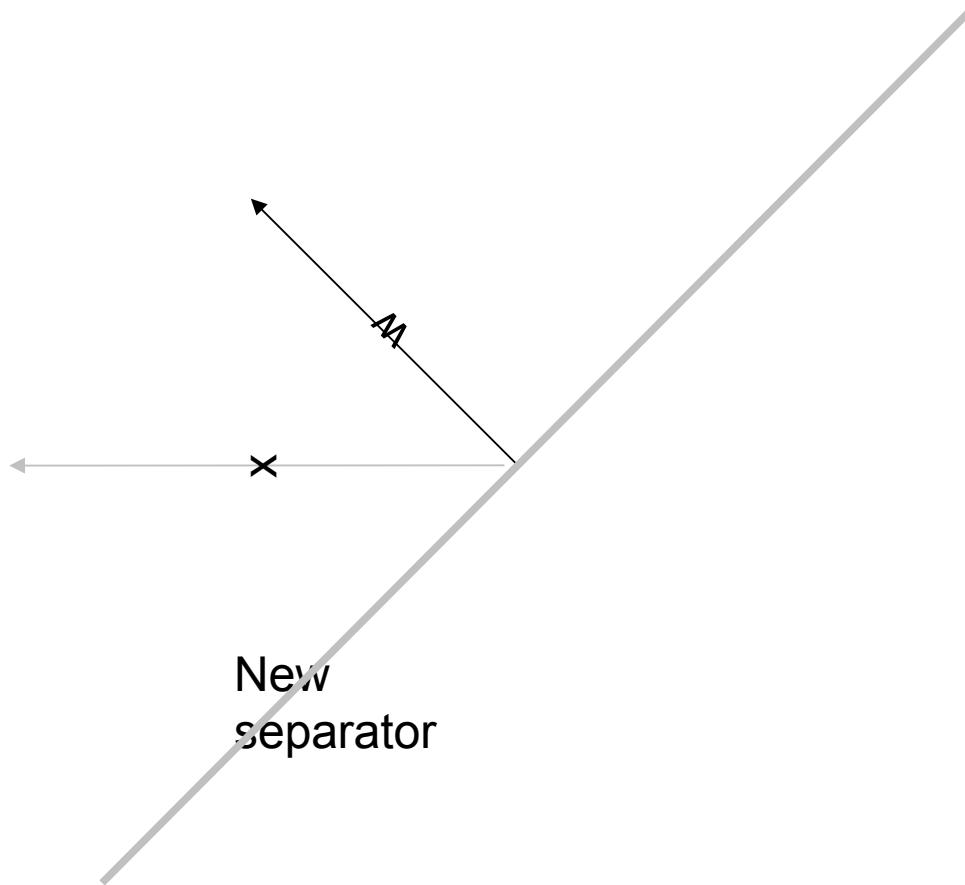
- x class is 1
- $f(x) = 0$
- $w += (1-0)x$

Perceptron update



- x class is 1
- $f(x) = 0$
- $w += (1-0)x$

Perceptron update

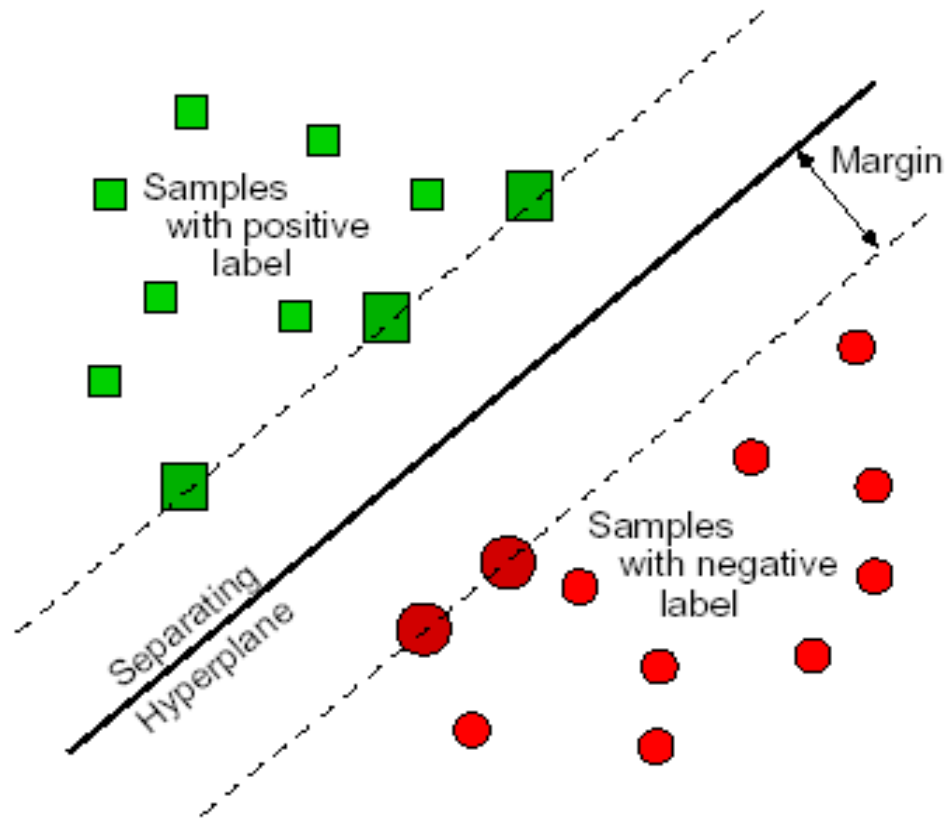


- Now x is on the right side of the separator!

Perceptron with Uneven Margins

- Both Perceptron and SVM implement “uneven margins”
- PAUM = “Perceptron Algorithm with Uneven Margins”
- This means that it doesn't position the separator centred between the points, but more towards one side

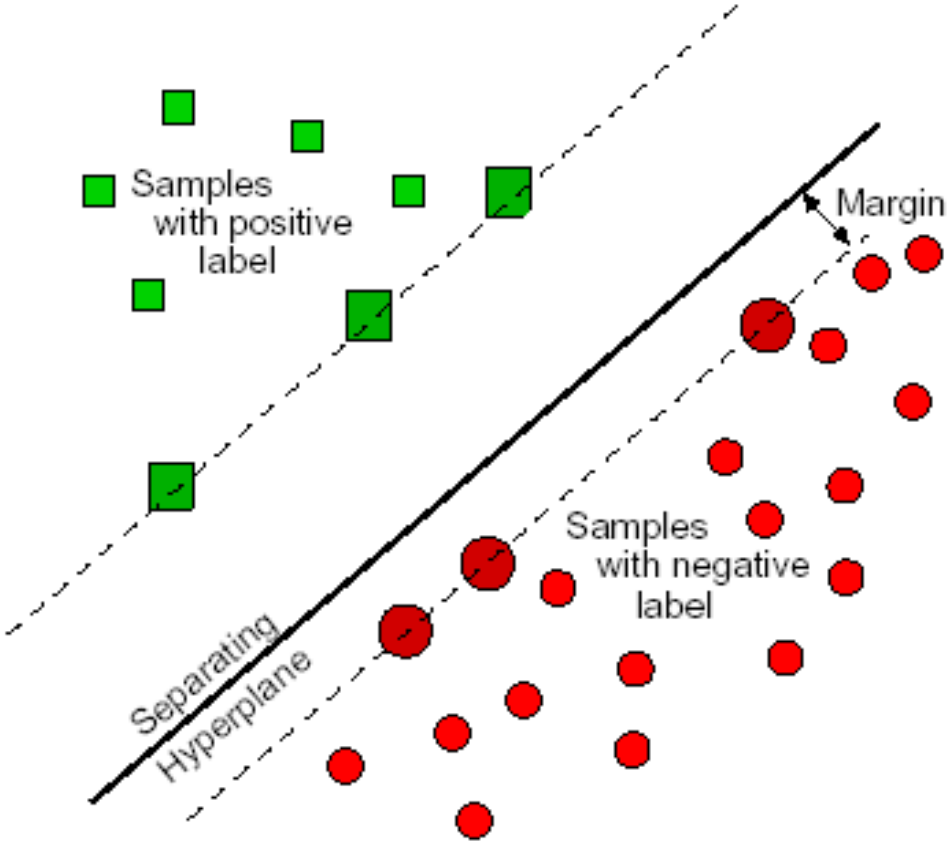
Even Margins



Why Uneven Margins?

- In NLP the datasets are often very imbalanced.
- If you are tagging instances of “Person”, there are a few positive cases mixed with many words that are not persons.
- In opinion mining, you may have a few sentences with opinions but mostly sentences without them.
- So move the margin away from the smaller group of training examples.
- Y. Li, K. Bontcheva, and H. Cunningham. Using Uneven Margins SVM and Perceptron for Information Extraction. CoNLL-2005.

Uneven Margins



Some Other Algorithms

- Batch Learning PR also includes the following from Weka
 - Naïve Bayes
 - Uses Bayes' theorem (probabilities) to determine the most likely class given attributes and training corpus
 - K-Nearest Neighbour
 - Determines class of a point based on k training points positioned geometrically closest to it
 - C4.5 (decision tree)
 - Makes a series of binary decisions that determine the class of a point based on its attribute values (e.g. “is string length > 3?”)



Classification tasks

- Opinion mining
 - Example: the documents contain spans of text (such as individual sentences or longer consumer reviews) which you want to classify as positive, neutral, or negative
 - Module 12 tomorrow will cover this, with hands-on work
- Genre detection: classify each document or section as a type of news
- Author identification



Classification tasks

- `thresholdProbabilityClassification`: the “pickiness” of the classifiers
 - increasing this generally raises precision and reduces recall
 - decreasing this generally increases recall and reduces precision
- `thresholdProbabilityBoundary` and `thresholdProbabilityEntity`: ignored



Classification tasks

- `<SURROUND VALUE="FALSE"/>`
 - the class boundaries are known
- **INSTANCE-TYPE**: type of annotation that covers each span of text to classify (Sentence, p (paragraph), etc.)
- We typically use NGRAM elements as attributes
- The GATE user guide gives examples, and Module 12 will cover this for opinion mining



Hands-on: text classification

- Close open applications, PRs, and LRs in GATE
- If you've closed GATE since the last exercise, you need the ANNIE, Tools, and Learning plugins for this exercise
- If you haven't closed GATE, load the Tools plugin
- Create new empty “training” and “test” corpora
 - Populate them from “language/training-corpus” and “language/test-corpus” directories in the hands-on material
 - Set the encoding to UTF-8 before you click OK
- Inspect the documents: the Key AS contains Sentence annotations with a lang feature
- Very few documents, but many instances (Sentence annotations)
- Task: language identification



Text classification

- Create a new **Conditional** Corpus Pipeline and add the following PRs:
 - Document Reset
 - ANNIE English Tokenizer
 - ANNIE Sentence Splitter
 - Annotation Set Transfer
 - Batch Learning PR with “language/ml-language.xml” as the config file
- Examine this config file in an editor and notice how it differs from the NER file



Text classification config file

- Note the changes for text classification:
- `<SURROUND value="false"/>`
- `thresholdProbabilityClassification` is used
- `INSTANCE-TYPE` is `Sentence`



Text classification example

- Training
- We use the Sentence annotations as instances, lang features as ML classes, and the tokenizer's output as attributes
- Check that Document Reset will keep the “Key” AS
- Switch the Sentence Splitter off (red signal light)
- Configure the AS Transfer PR to copy all annotations from “Key” to the default AS
- Set the Batch Learning PR to TRAINING mode
- Set the pipeline to run on the training corpus
- Run the pipeline



Text classification example

- Testing
- Here we create our own Sentence annotations and use ML to classify them
- Switch the Sentence Splitter on (green light)
- Switch the AS Transfer PR off (red light)
- Set the Batch Learning PR to APPLICATION mode
 - Leave the inputASName blank (default AS)
 - Set the Batch Learning PR's output AS to "Output"
- Set the pipeline to run on the test corpus
- Run the pipeline



Text classification example

- Inspect the test corpus with Corpus QA:
 - A = Key, B = Output
 - select “Sentence” annotations and the “lang” feature
 - select “F1 strict” and click “Compare”
- My results:
 - 411 matches, 16 “only A”, 16 “only B”
 - so all sentences were classified, and only 16 were classified incorrectly

Text classification example

- In Corpus QA, try Classification → Observed Agreement, click Compare, and look at the “Confusion Matrices” tab

- I get a table like this:

	de	en	fr
de	294	0	0
en	8	86	0
fr	8	0	31

- This shows that 8 English & 8 French sentences were misclassified as German
- Module 12 will include ML text classification examples for opinion mining



Further tinkering

- Try lower or higher threshold values
- Try different combinations of attributes