Supervised Machine Learning

Session Overview

- Introduction to machine learning
  - Terminology
  - Development cycle
- Classification practical session
  - Feature preparation
  - Training and application
  - Corpus QA classification metrics
  - Evaluating with cross-validation
- Chunking practical session
  - Training and application
  - Evaluating with Corpus QA and Annotation Diff
- Deep learning demo

Introduction to machine learning

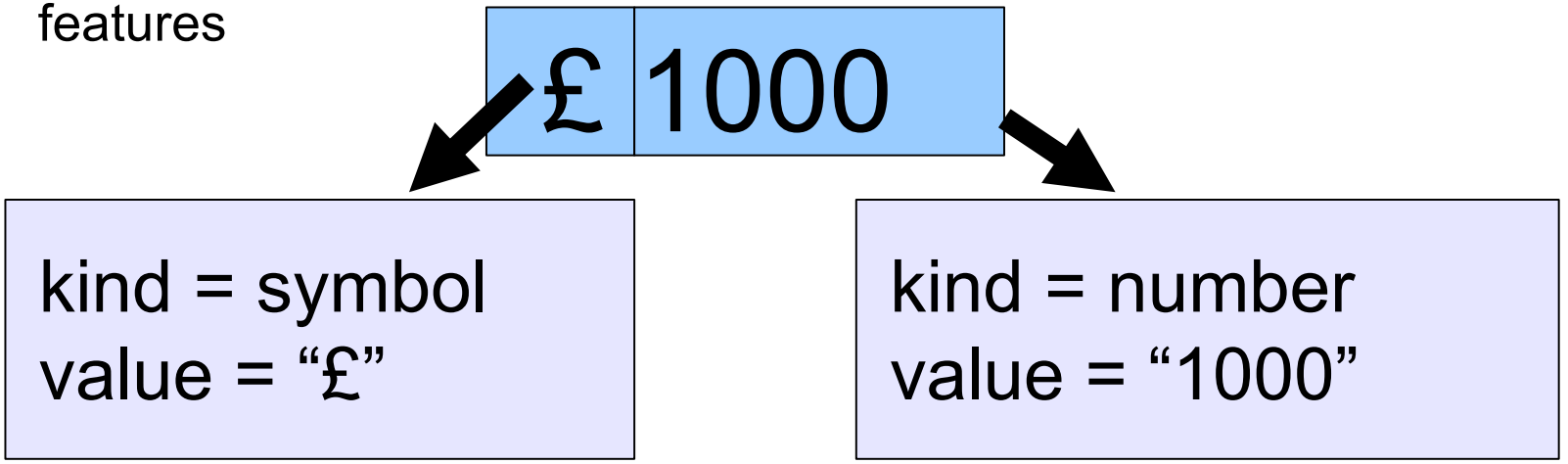
Introduction to ML

- We will introduce ML by providing an overview of terminology only
- We cannot provide a tutorial on ML today due to limited time. However we'll introduce basic concepts in the context of NLP
- For a deeper insight, try:
  - Playing with Weka and reading the Weka book http://www.cs.waikato.ac.nz/ml/weka/index.html
  - Andrew Ng's course:

    https://www.coursera.org/course/ml

Learning a pattern

- Machine learning means automating the process of inferring new data from existing data
- In GATE, that means creating annotations by learning how they relate to other annotations
- 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 JAPE 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
- Features
- Classes

Instances

- Instances are cases that may be learned
- Every instance is a decision for the ML algorithm to make
- E.g. for each word in a sentence, what is it? Is it a location? Person? Neither? For each sentence instance, is it in French? English?

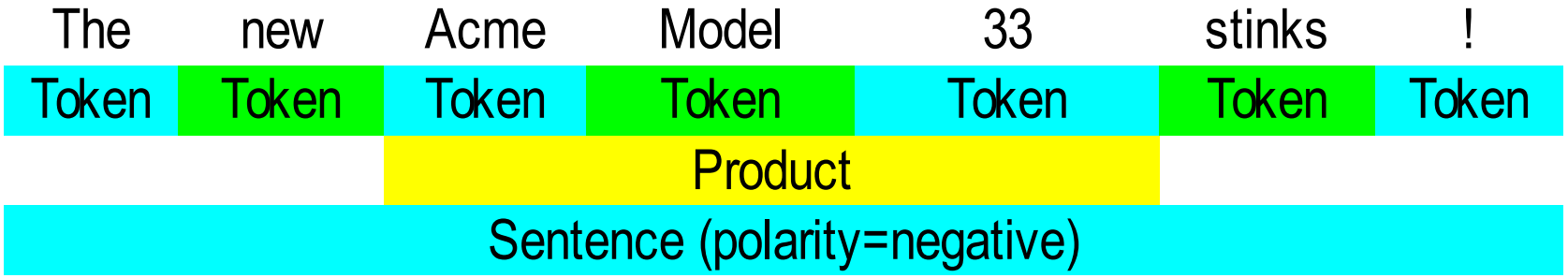
Features

- Features are pieces of information about instances
- They may also be called “attributes”
- For example, the text string, its part of speech, the text of the preceding instance, all the words in the sentence ..
- When you want to identify e.g. the language of a sentence, what information do you use?

Classes

- The class is what we want to learn
- Suppose we want to learn the language of a sentence: for every instance (sentence), the question is “what language is this in?” and the classes might be “English” and “French”
- Sometimes there are many classes, for example many other language possibilities
  - For every instance, the question is “which type from the list does this instance belong to?”

Example: text classification



- instance: Sentence annotation
- features: 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.

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
- Genre detection: classify each document or section as a type of news
- Author identification
- Classifying sentences according to language

Instances, attributes, classes in a chunking task

California Governor Arnold Schwarzenegger proposes deep cuts.

Terminology: Instances, attributes, classes

California Governor Arnold Schwarzenegger proposes deep cuts.

Instances: Any annotation

Tokens are often convenient

Token | Token | Token | Token | Token | Tok | Tok

Terminology: Instances, attributes, classes

California Governor Arnold Schwarzenegger proposes deep cuts.

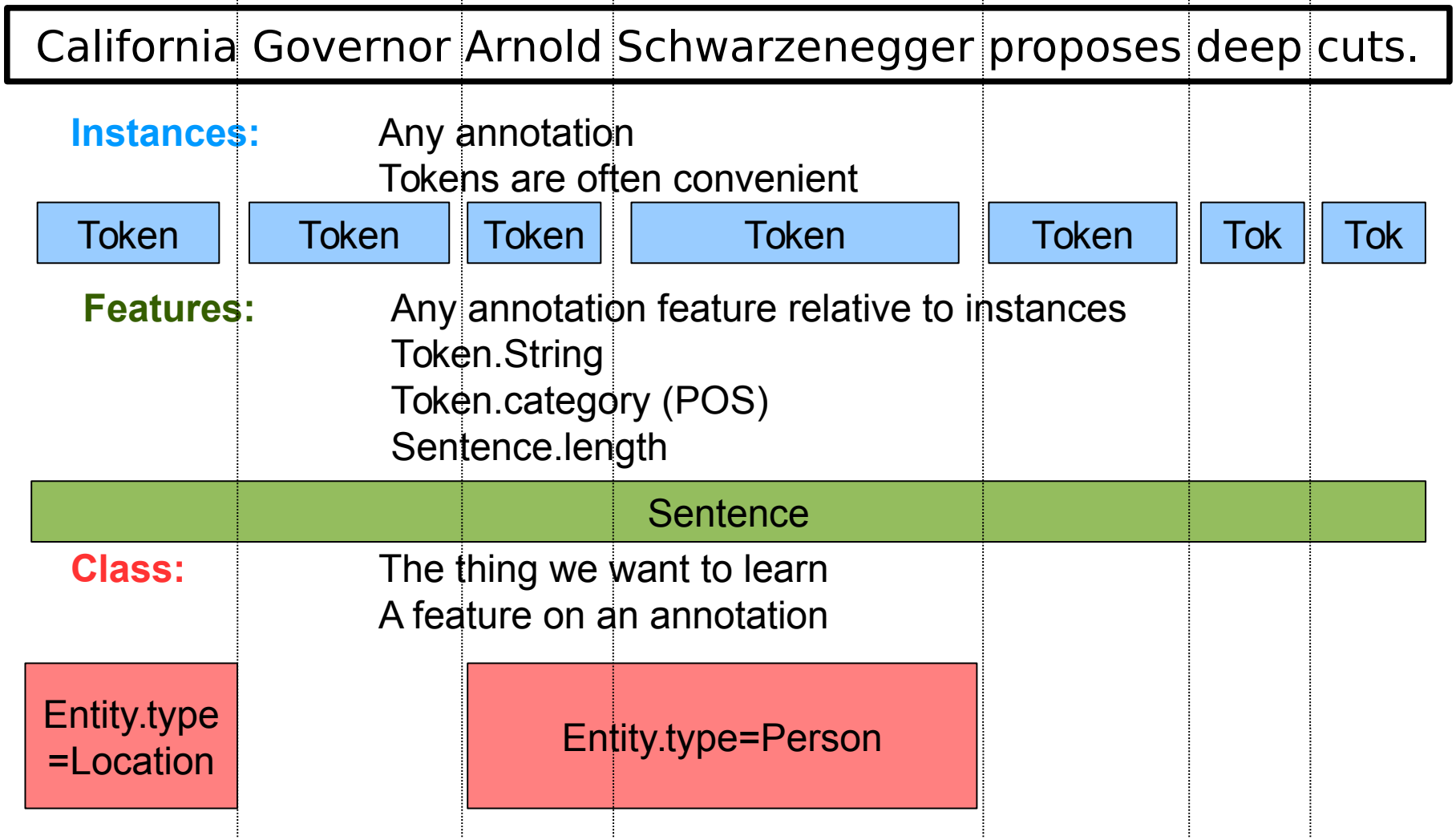
Instances: Any annotation
Tokens are often convenient

Token | Token | Token | Token | Token | Tok | Tok

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

Sentence

Terminology: Instances, attributes, classes



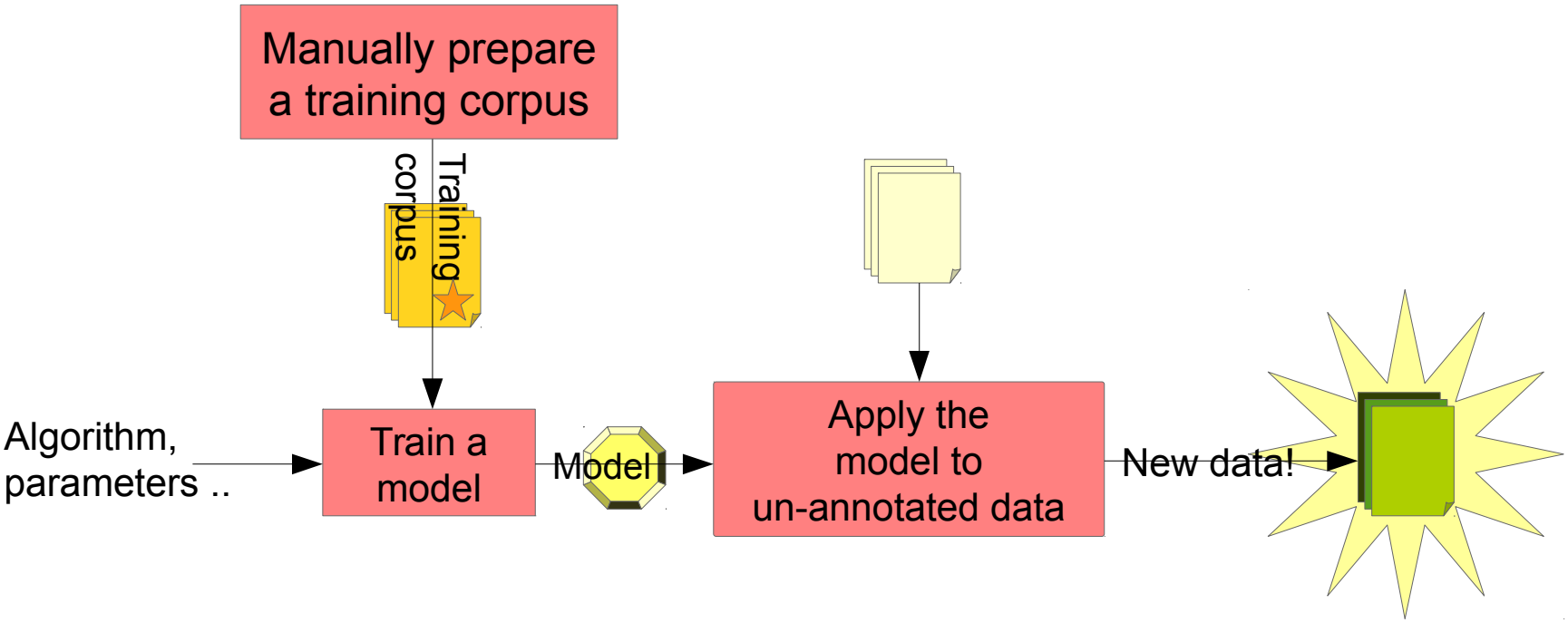
Training

- 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 features
- Models are representations of decision-making processes that allow the machine learner to decide what class the instance has based on the features of the instance

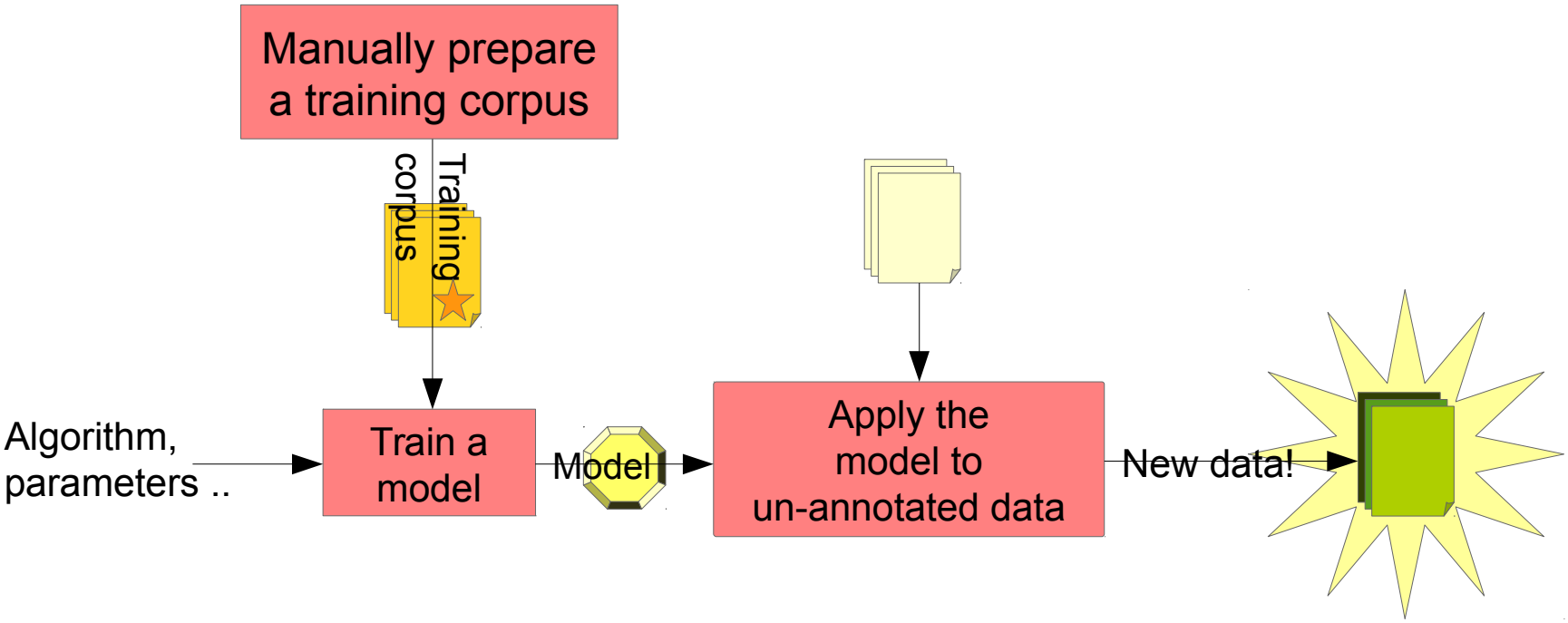
Application

- 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 feature annotations
- The machine learner will work best if the application data is similar to the training data

Development Cycle



Development Cycle

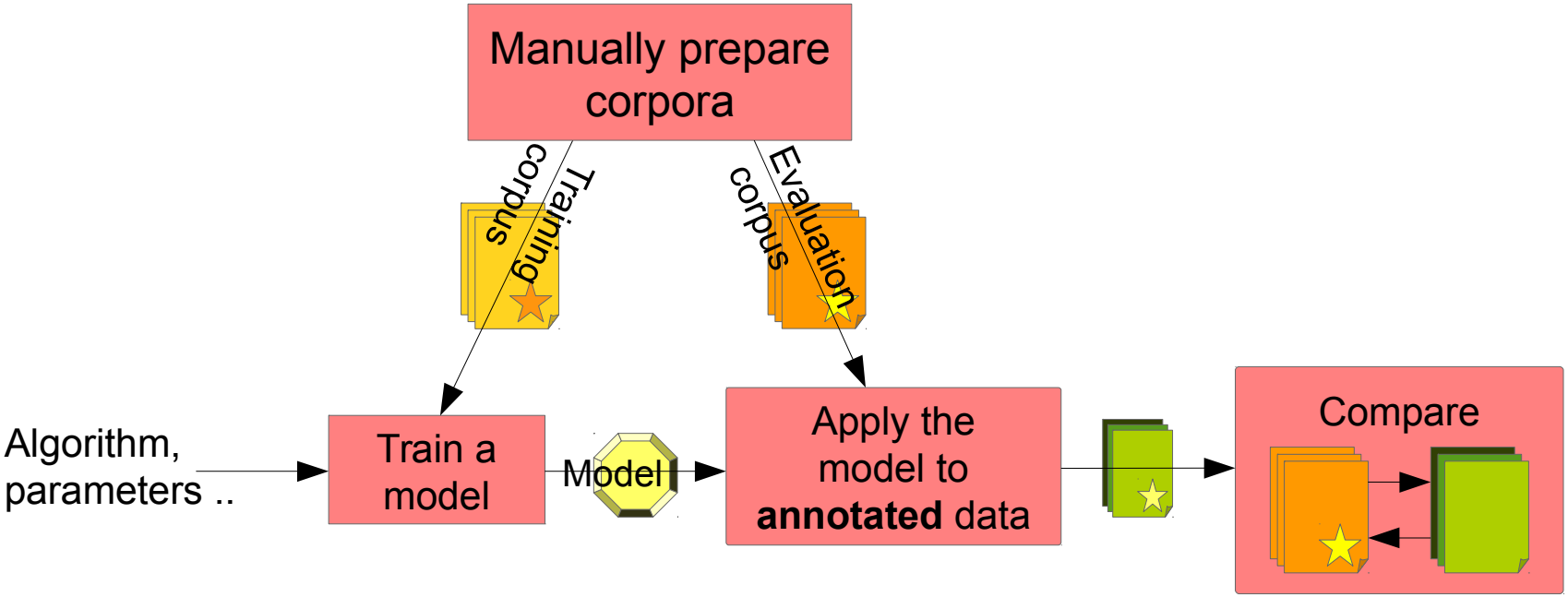


But how do we know how good it is?

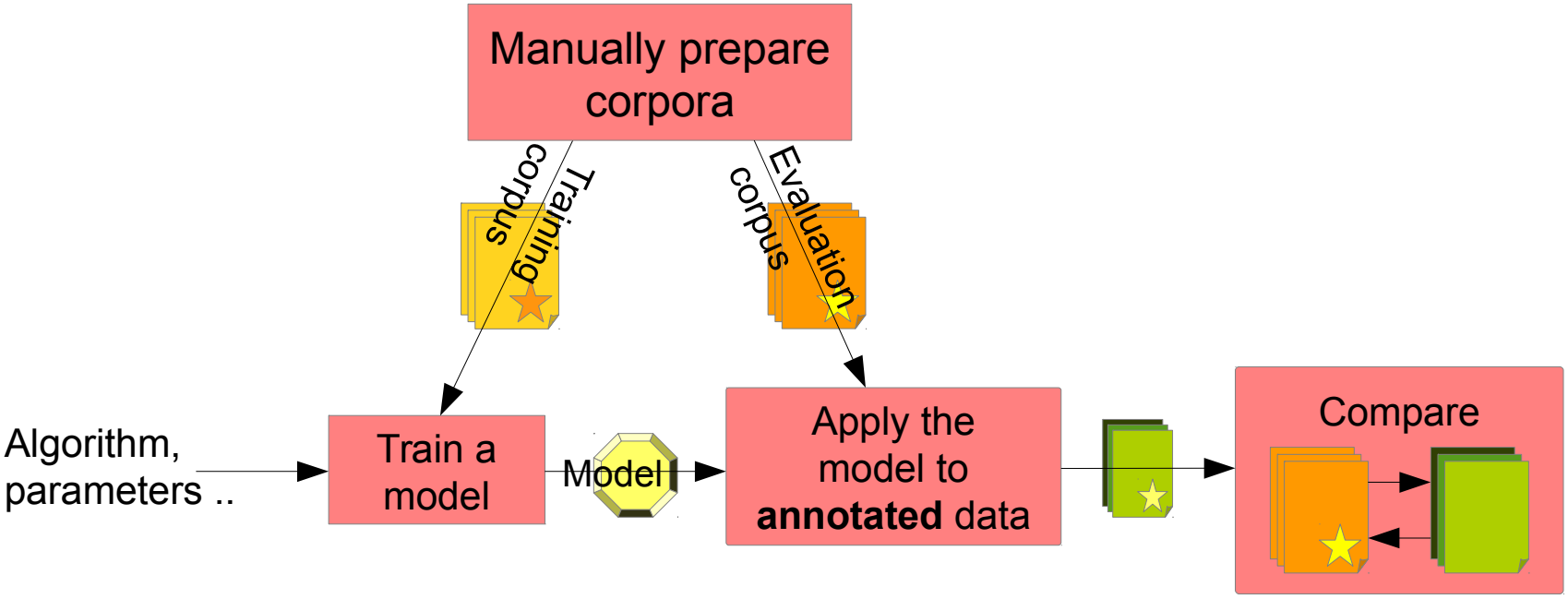
Evaluation

- 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

Development Cycle



Development Cycle

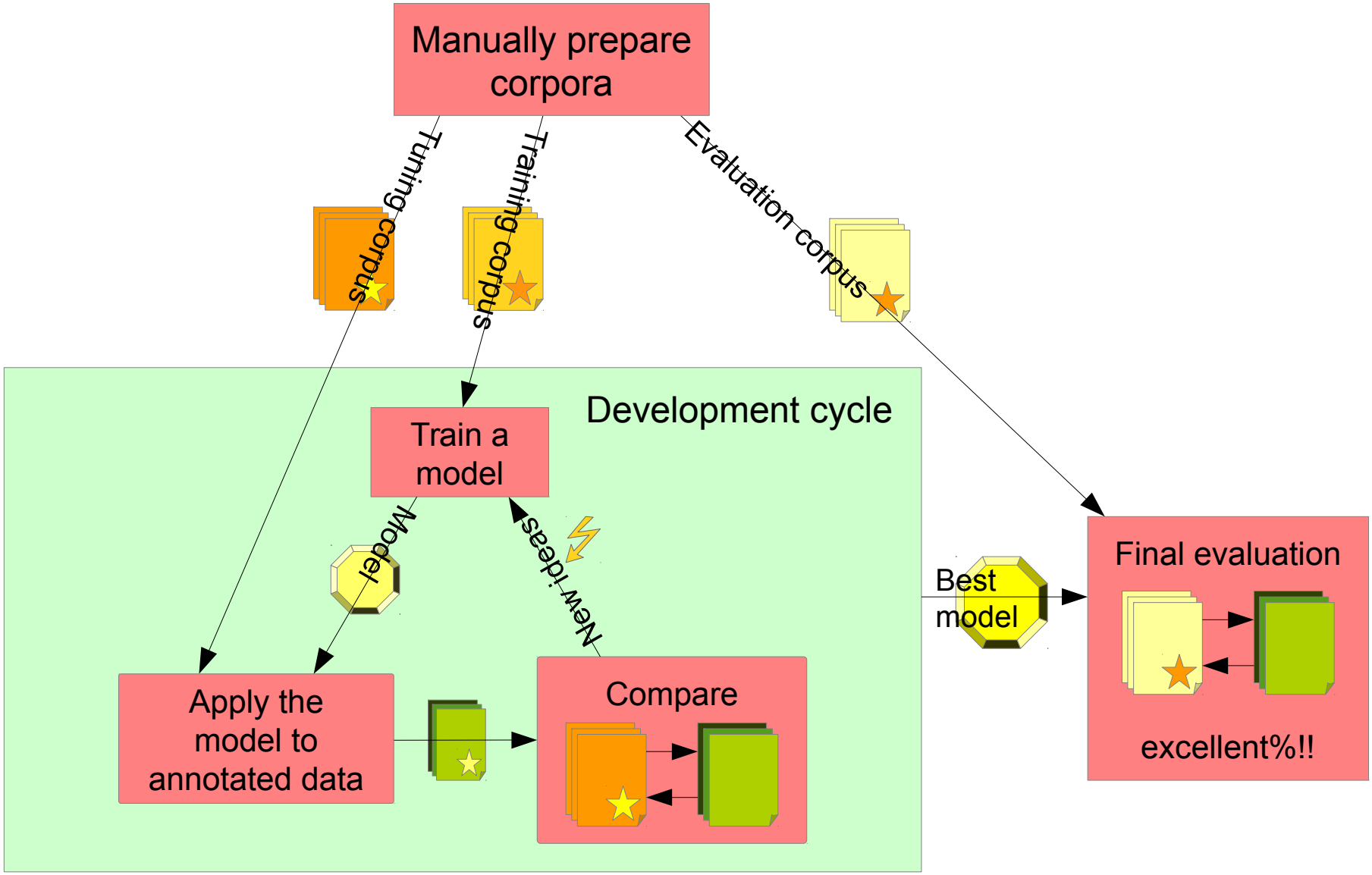


But I don't like that result! I want to make it better!

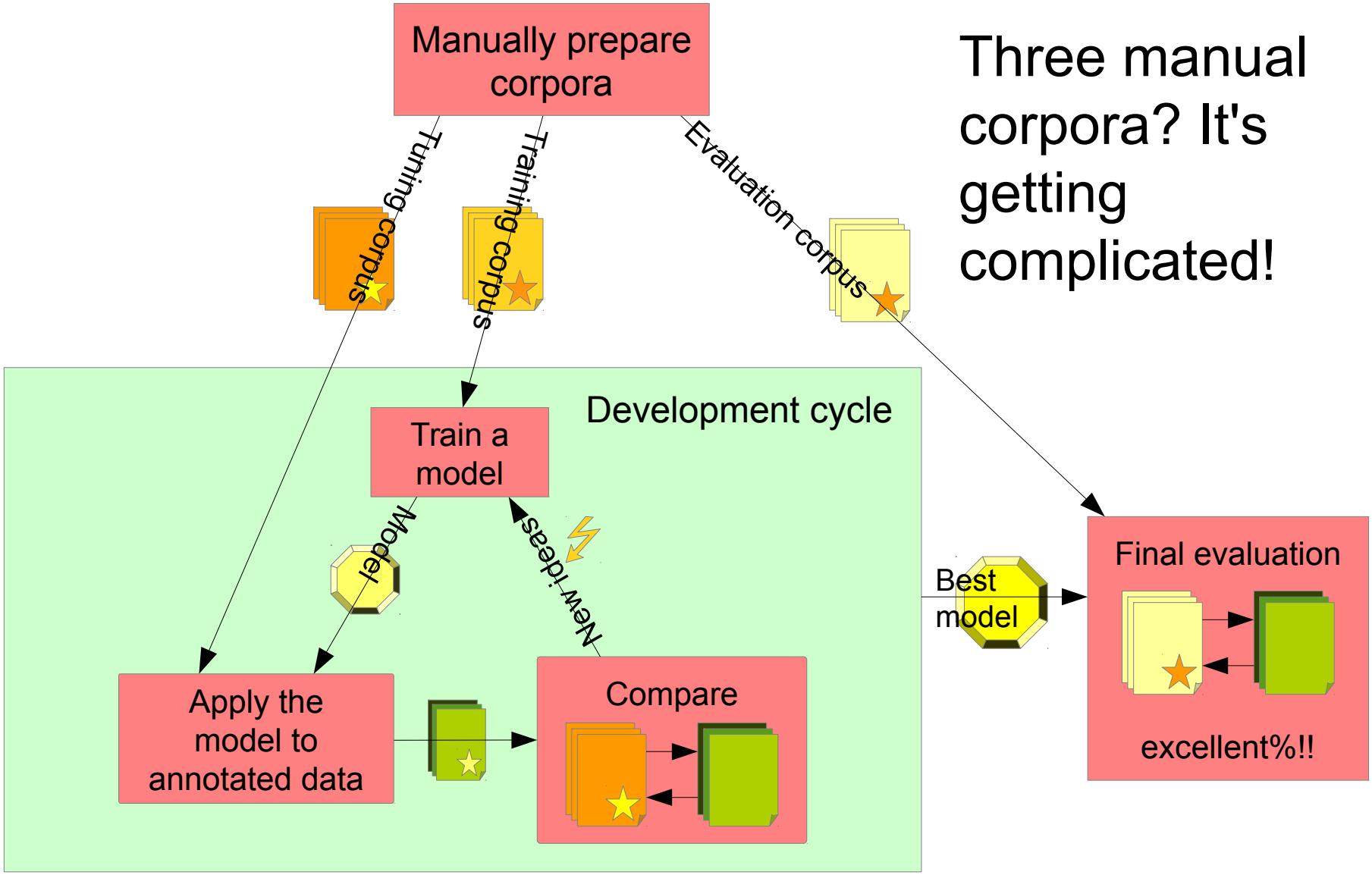
Tuning

- An important part of machine learning work is trying different things to get a good result
- However, be aware that if you tune to get a good result on a corpus, it will be artificially good!
- Some of what you learned transfers to new data, but some of what you learned may be specific to this corpus
- So you need a fresh corpus to get a final result

Development Cycle

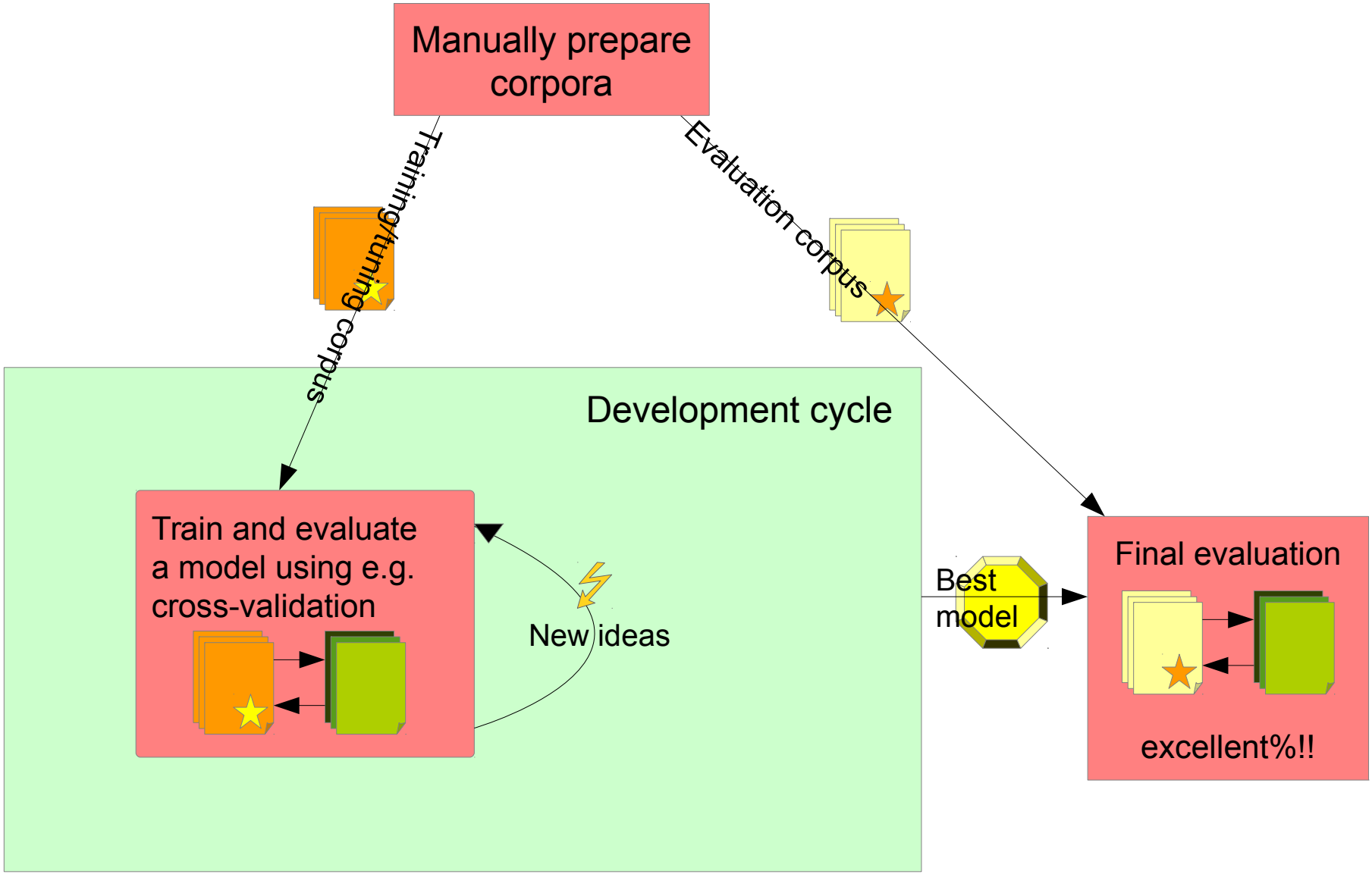


Development Cycle



Three manual corpora? It's getting complicated!

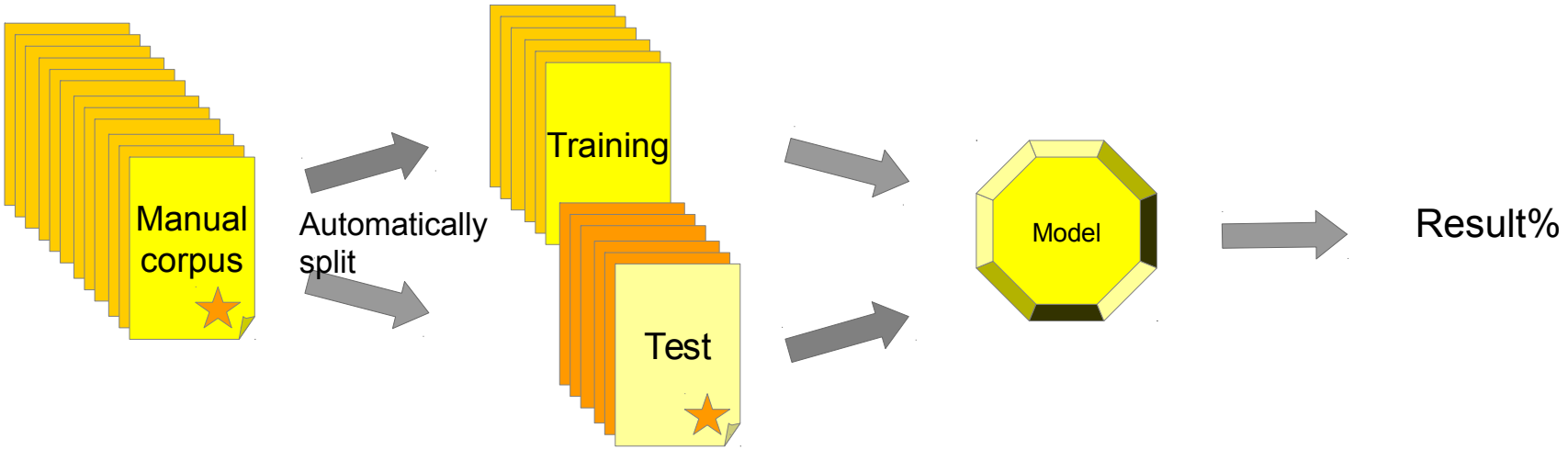
Development Cycle



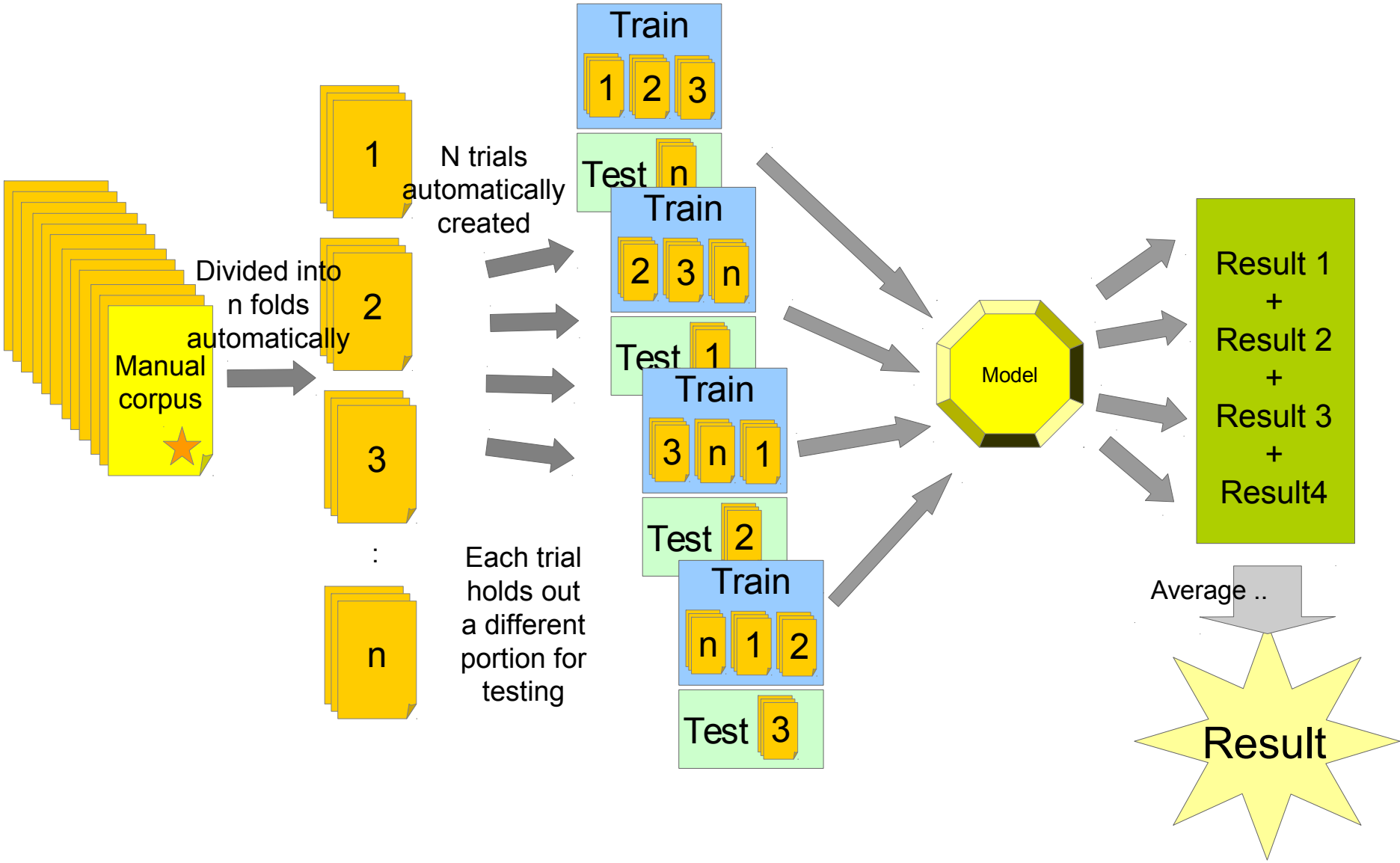
Cross-validation and hold-out evaluation

- The process of splitting a corpus for training and application can be facilitated, so you don't have to split the corpus and run separate training and application steps yourself
- Hold-out evaluation holds back a portion of the corpus for testing
- You can automatically do this a number of times and take an average
- Cross-validation splits the corpus into n portions (“n-fold cross-validation) and in turn, holds each out for testing, then averages all the results
- You could hold out just a single instance each time, maximizing your training portion! The more folds, the longer it takes though
- All you have to do is select which you want, and everything is done automatically

Hold-out evaluation



N-fold cross-validation

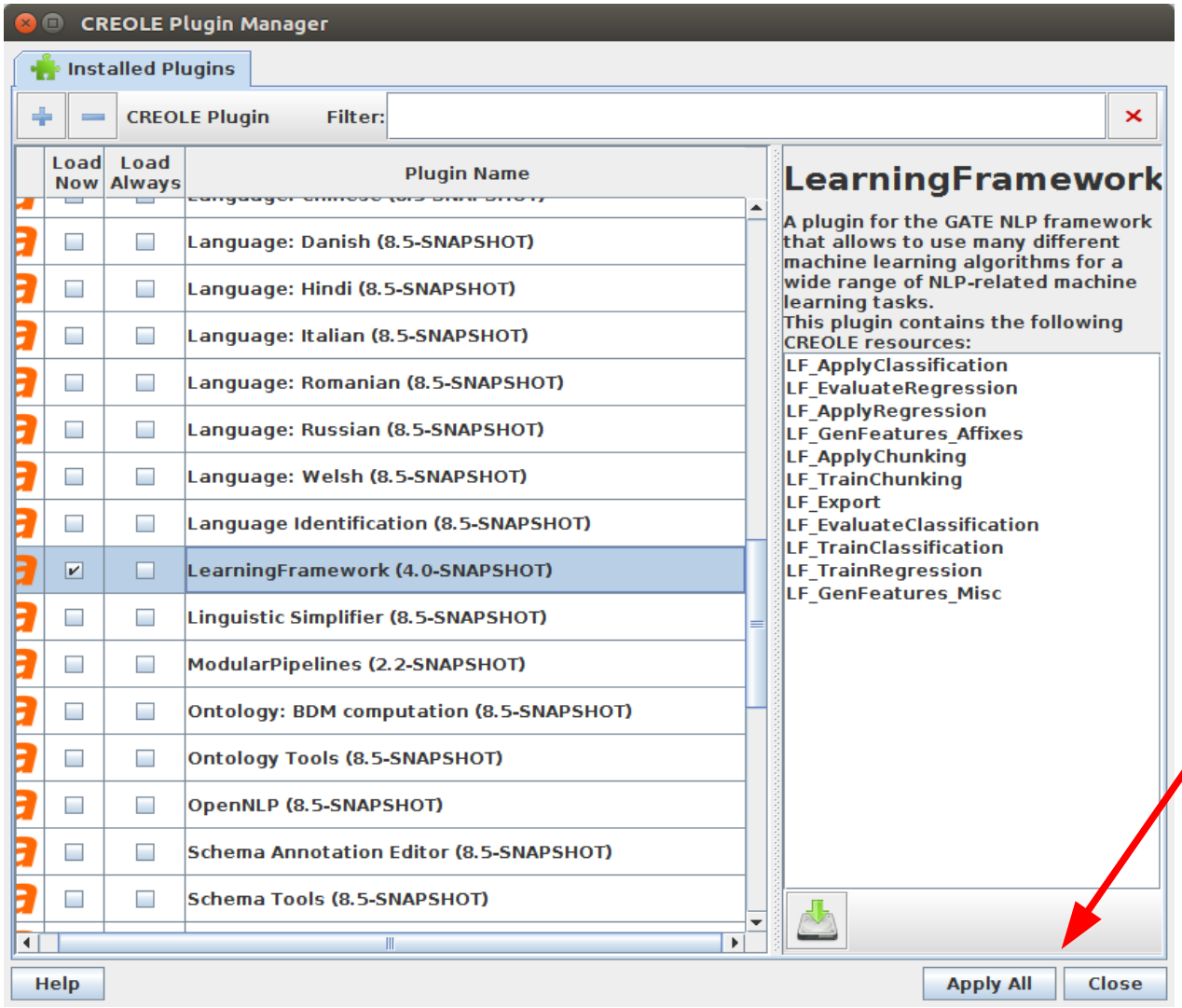


Machine Learning in GATE

- GATE supports machine learning in several ways. Some of the standard PRs are ML-based e.g. Stanford parser
- **Learning Framework**
  - Includes many algorithms
  - Includes deep learning
  - Extensive, actively developed and supported

  `https://github.com/GateNLP/gateplugin-LearningFramework`
- **Batch Learning PR** and **Machine Learning PR**: Old and older(!) GATE ML PRs, no longer supported.
- In this session we will learn to use the Learning Framework and explore its possibilities

Getting the Learning Framework Plugin

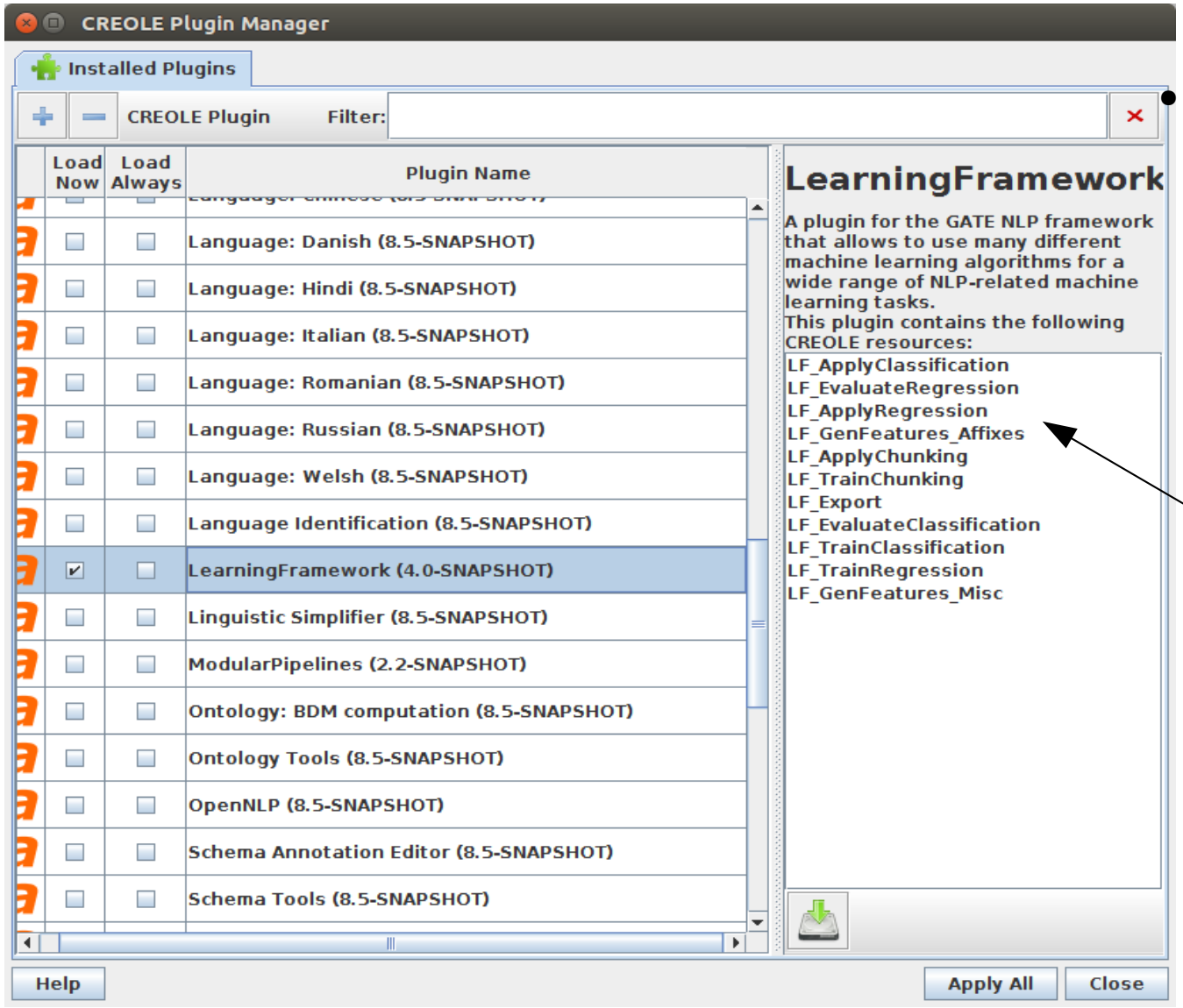


Load the Learning Framework plugin

Load the Tools plugin and the Groovy plugin, while you're there!

Don't forget to apply!

PRs in the plugin



In the plugin manager you might have noticed that the Learning Framework plugin contains 11 PRs

ML Tasks in the Learning Framework

- The Learning Framework supports 3 types of ML tasks:
  - Chunking (named entity recognition, finding NPs)
  - Classification (sentiment classification, POS tagging)
  - Regression (assigning a number rather than a type, for example ranking candidates for named entity linking)
- Separate PRs for training and application facilitate each of these tasks

ML Tasks in the Learning Framework

- The Learning Framework supports 3 types of ML tasks:
  - Chunking (named entity recognition, finding NPs)
  - Classification (sentiment classification, POS tagging)
  - Regression (assigning a number rather than a type, for example ranking candidates for named entity linking)
- Separate PRs for training and application facilitate each of these tasks

PRs in the Plugin

- Evaluate Classification PR provides an accuracy figure for classification evaluation (cross-validation and hold-out)
  - Can be used to evaluate the classification aspect of chunking—more on this later
  - Evaluate Chunking PR is forthcoming .. But in the mean time you can use the normal GATE evaluation tools
- Export—data are exported for use in ML tools outside of GATE
- GenFeatures—some common feature preparation tasks are facilitated in these PRs

Documentation

- The documentation for the plugin is available here:

  https://github.com/GateNLP/gateplugin-LearningFramework/wiki

- You can find advice about algorithm parameters, feature specification and so on
- In today's course you will be asked to use your initiative at times, and may find it useful to consult this wiki!