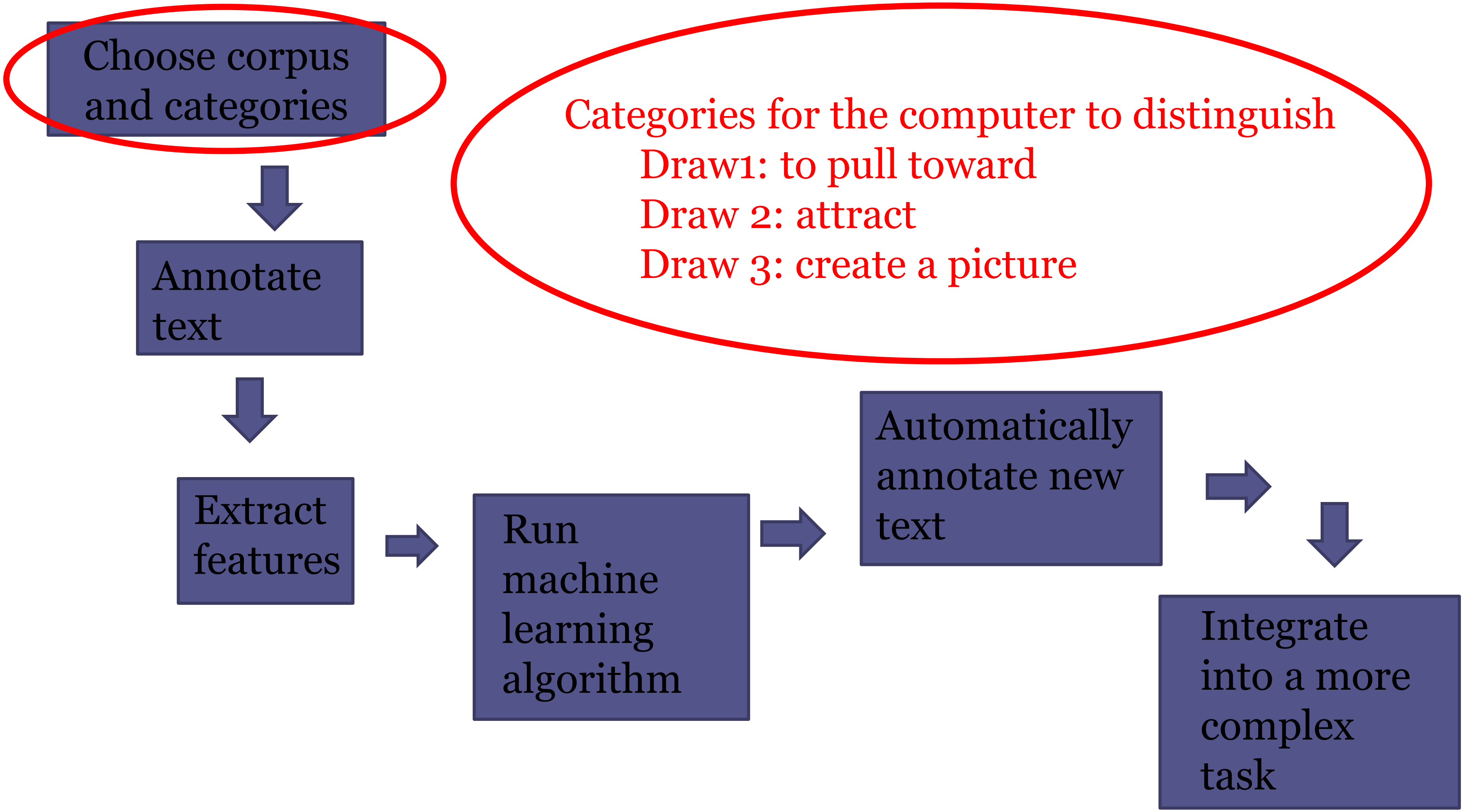
Machine Learning for Semantics in NLP

Supervised machine learning for NLP



Supervised machine learning for NLP

Choose corpus and categories

Annotate text

Label each instance of *draw* in the corpus as draw1, draw2 or draw3

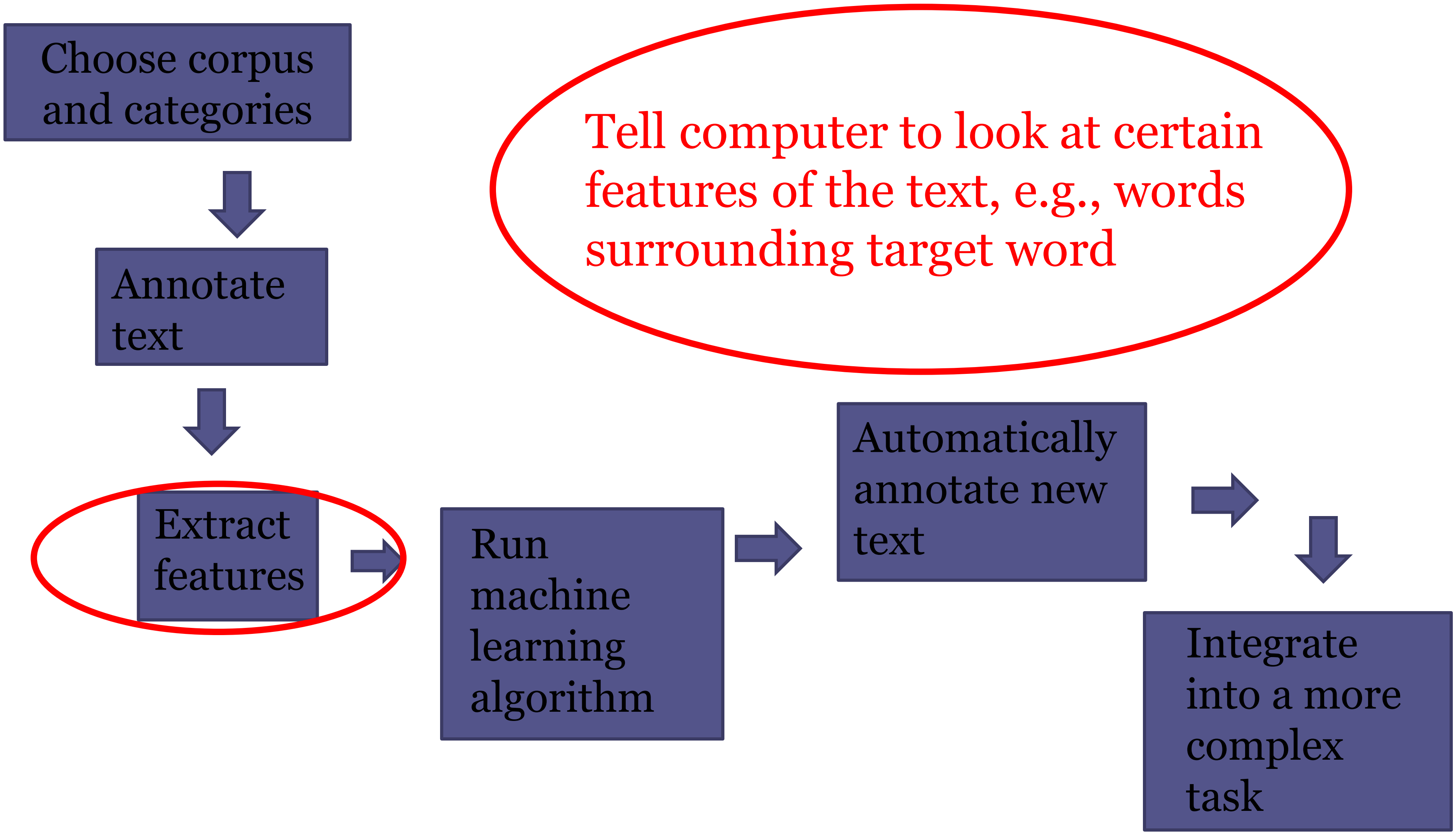
Extract features

Run machine learning algorithm

Automatically annotate new text

Integrate into a more complex task

Supervised machine learning for NLP



Supervised machine learning for NLP

Choose corpus and categories

Annotate text

Extract features

Run machine learning algorithm

Automatically annotate new text

Integrate into a more complex task

Computer correlates features with different senses of *draw*

Supervised machine learning for NLP

Choose corpus and categories

Annotate text

Extract features

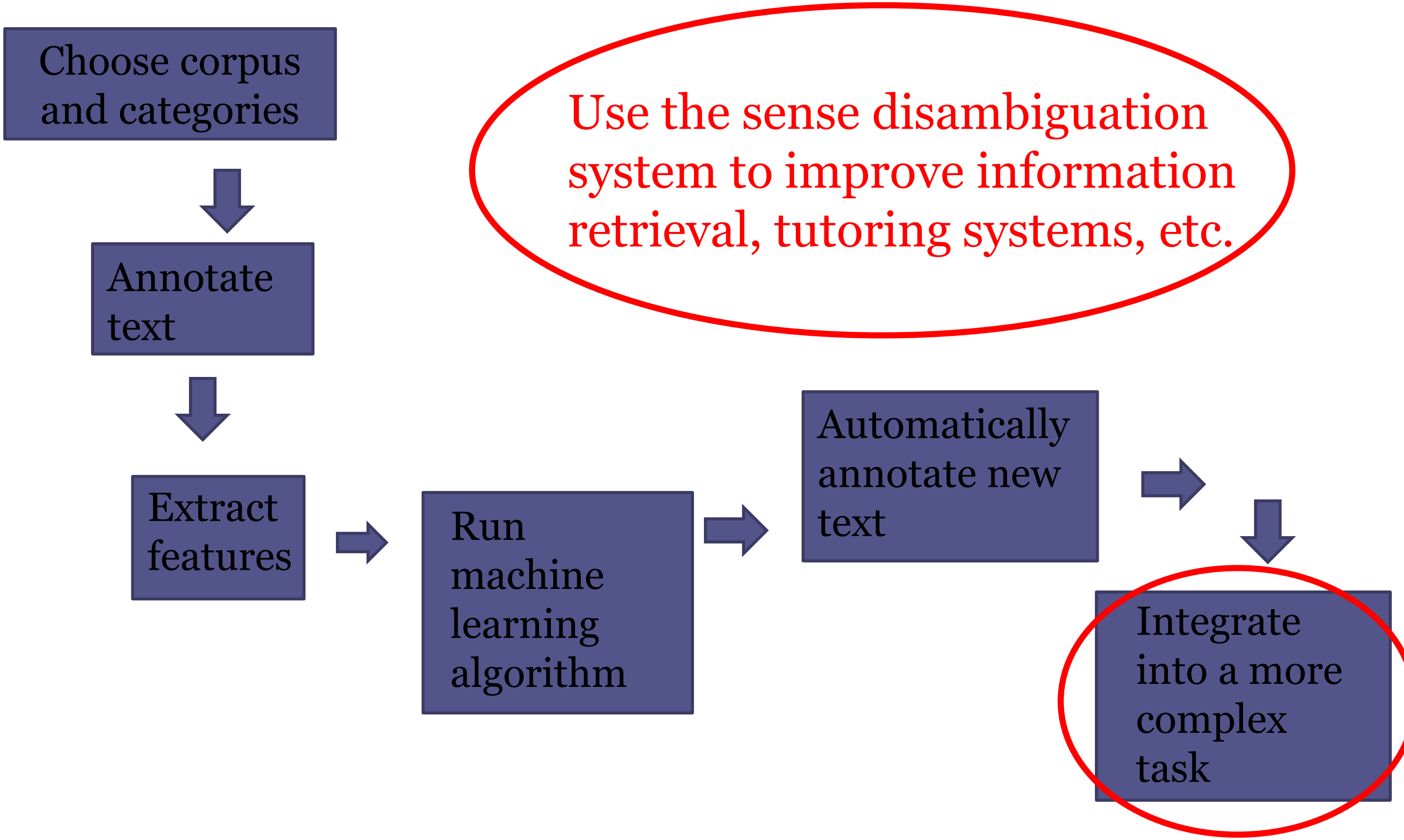
Run machine learning algorithm

Automatically annotate new text

Integrate into a more complex task

When given new text, the system looks at those features to label instances of *draw* with draw1, draw2, or draw3

Supervised machine learning for NLP



Features for WSD

- Lexical
- Syntactic
- Semantic

Lexical

- Words surrounding the target word
  - Choose the window size (2-3 is common)
- All the words in the sentence (and the sentence before and after)
- POS tag for words in the window
- 10,000s of features
- Binary choice for all the words in the corpus
- Most powerful features

Syntactic

- Parse tree labels of the phrase's and its siblings' head words
- For verb sense disambiguation
    - Whether a sentence is passive or active
    - Whether target has subordinate clause
    - Whether target has a PP adjunct
    - Parse tree label of the verb's parent

Semantic

- Named entity type of the word
- Document topic
- For verbs:
  - Synonyms and hypernyms of the arguments (WN)
  - Named entity type of the arguments
  - Dynamic dependency neighbors (object classes)

Features for SRL: Parse-free

- Argument phrase type
  - FN Speaker likely to be a noun phrase
  - FN Topic likely to be a PP or NP
  - FN Medium likely to be a PP
  - [We] talked [about the party] [over the phone.]
- Argument position relative to the target predicate
- Argument order
  - First step is to ID arguments of a sentence
  - Number the arguments

SRL features: Using a parse

- Governing category: Subject or in the VP
- Path through the parse tree from the target predicate to the argument
- Active or passive voice
- Head word of the phrase
  - Lexical feature that needs a parse
- Head word of objects of PPs
  - On Monday
  - On the table

SRL features: to parse or not

- Some languages do not have high-accuracy automatic parsers
- Parsing takes a long time
- Chunking is almost as good (Carreras and Marquez, 2005)
  - NP V NP PP
- Use both to compensate for parser errors

How do you get the features?

- For most realistic assessment of a system, should be done automatically
- The system should be usable on new data
- For example, for syntactic features, use an automatic parser
- Automatic parsers produce errors
  - Lowers a SRL system's F score by 10 points
  - Less impact on WSD

What type of parse to use

- Phrase structure parser (Penn Treebank)
- Combinatorial Categorical Grammar (CCG)
- Lexical Tree Adjoining Grammar (LTAG)
- Dependency parses
- Last 3 more compatible with SRL (Palmer et al., 2010)

Which classifier to use?

- SVM is fast
  - Good for data with a lot of features
  - Good for creating many classifiers (wsd)
- Try different ones out
- SRL
- 2-stage process
  - ID and label individual arguments
  - Finding the best set of roles for an entire sentence
    - Reranking
    - Viterbi search
    - Integer linear programming

Evaluation of WSD systems

- Accuracy
- Percentage correct, as judged against "gold standard" annotation
- Compared to a lower bound, usually the accuracy of a most-frequent-sense method
  - Can be quite high for words with one dominant sense
- Compared to an upper bound: inter-annotator agreement

Evaluation of SRL systems

- System must find the constituents to annotate
- Precision: Percentage of labels output by the system that are correct
- Recall: Percentage of true labels the system identifies

True: [Agent He] ate [Patient the peaches] [Instrument with a spoon.]

System: [Agent He] ate [Patient the peaches] with a spoon.

Precision: 100%; Recall: 66%

F-score

- A way to combine precision and recall into one score
- Harmonic mean of precision (P) and recall (R)

$$F = \frac{2PR}{P + R}$$

Feature evaluation

- Difficult to see which features contributed the most to defining the categories, especially when using SVM
- Run the system (train and test) using only one feature or one type of feature
- Add in another feature and run again
- Compare the results. Did the new feature help?
  - Simple comparison: Is one score higher?
  - Significance tests: Is one score significantly different from the other?

VerbNet classifier

- Treated as a verb sense disambiguation task
- One classifier per verb
- Semlink corpus used for training and test data
- 344 multiclass verbs
  - average 2.7 classes
  - average of 133 instances
  - Includes verbs labeled in the corpus with one VerbNet class and “No appropriate class”

Features

- Lexical
  - Neighbor words and their POS
- Syntactic
  - Passive/active
  - Types of phrases and clauses
  - Heads of phrases
- Semantic
  - Synonyms and hypernyms of arguments
  - Named entity features
  - Dynamic dependency neighbors

Overall Results

- Accuracy, using 5-fold cross validation: 88.67%
- Baseline (most frequent class): 77.78%
- Error reduction: 49%

Feature Experiments

- Developed several different models, each with a different combination of features
- Created a dedicated test set using 30% of the Semlink corpus

Model
Lexical features only
Lexical + syntactic
Lexical + semantic
All but DDN
Lexical + syntactic + DDN
All features

Feature Results

Model	Baseline (%)	Accuracy (%)	Error Reduction (%)
Lexical features only	77.78	83.07	23.81
Lexical + syntactic	77.78	84.44	29.97
Lexical + semantic	77.78	83.75	26.87
All but DDN	77.78	84.12	28.53
Lexical + syntactic + DDN	**77.78**	**84.89**	**32.00**
All features	77.78	84.65	30.92

- DDNs added significantly more than the other semantic features, resulting in the best-performing model

Tools you can use: WEKA Explorer

- Open source
- Implemented in Java
- Graphical user interface
- Preprocessing tools
- Multiple algorithms
  - K-nearest neighbor
  - Naïve Bayes
  - Perceptrons, including SVM
- Visualization tools
- Hands-on tutorial next week
- http://www.cs.waikato.ac.nz/ml/weka/

Tools you can use: RapidMiner

- Open source
- Implemented in Java
- Graphical user interface
- Preprocessing tools
- Multiple algorithms
  - K-nearest neighbor
  - Naïve Bayes
  - Perceptrons, including SVM
- Visualization tools
- http://rapid-i.com/content/view/181/190/