Decision Tree for Text Categorization

School of Information Studies
Syracuse University

Text Categorization

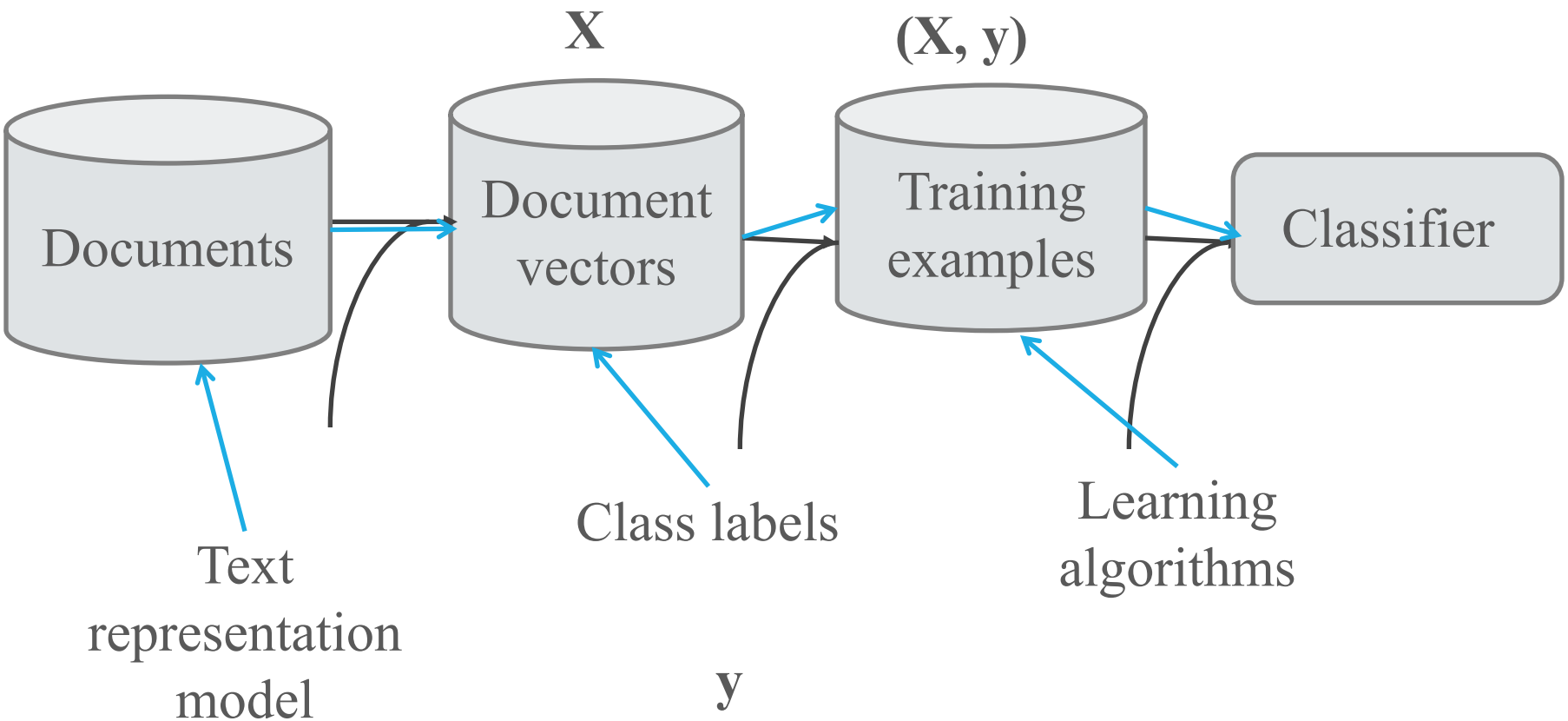
Two steps: training and testing

Step 1: Training

- Goal: build a prediction model (a "classifier") that assigns documents to pre-defined categories (e.g., positive, negative, and neutral comments)
- Input: a collection of training documents and a computer algorithm

	"happy"	"sad"	"mad"	…	Category
Doc1	1	0	0		positive
Doc2	0.1	0.3	0.6		negative
Doc3	0.1	0.1	0.1		neutral

Training a Text Classifier



Step 2: Testing

Goal: use the classifier to predict the category of new documents

Input: a trained classification model and a collection of testing documents with unknown category labels

	"happy"	"sad"	"mad"	…	Category
Doc1	0.8	0.1	0.1		?
Doc2	0.5	0.3	0.2		?
Doc3	0.2	0.4	0.4		?

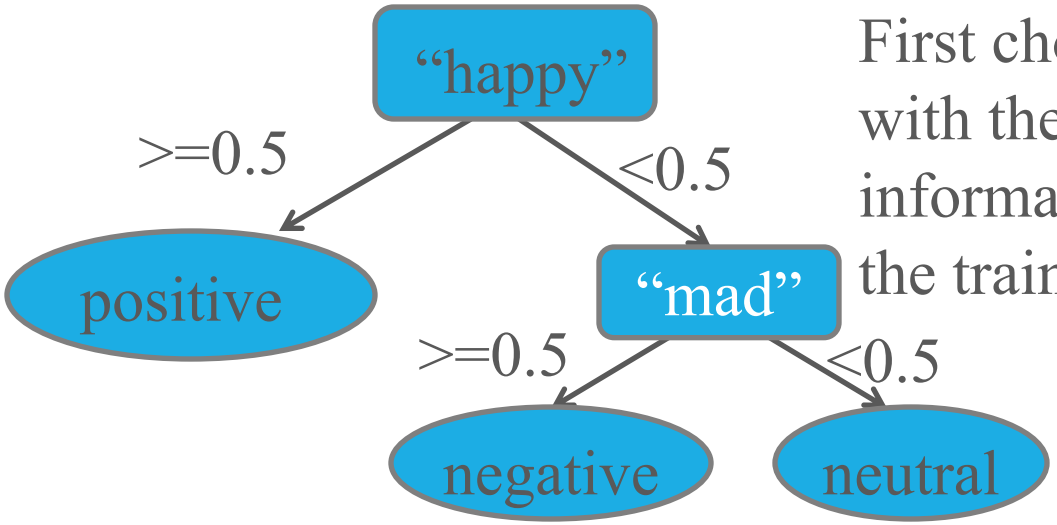
How to Train a Text "Classifier"?

Many candidate algorithms

- Decision tree
- Naïve Bayes
- Support vector machines
- K-nearest neighbor
- Neural network
- …

Train Decision Tree Model

	"happy"	"sad"	"mad"	…	Category
Doc1	1	0	0		positive
Doc2	0.1	0.3	0.6		negative
Doc3	0.1	0.1	0.1		neutral

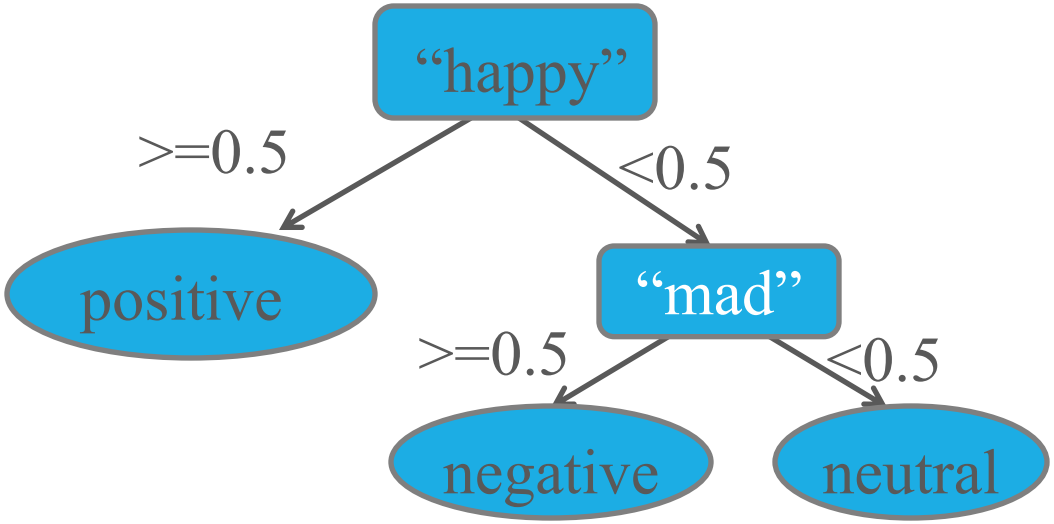


First choose the attribute with the highest information gain to split the training data

Use Decision Tree Model for Prediction

What is the sentiment of this text document?

- "*happy happy happy happy mad mad mad sad sad sad*"
- {"happy" = 0.4, "mad" = 0.3, "sad" =0.3}



Decision Tree Is Not Commonly Used in Text Categorization

A few problems

- Black and white decision: mixed sentiment?
- The tree may become very big: too many word features!