Text Classification

- What kind of learning (from Ch. 18) is this?
- What about our class project?

- Choose the most likely category for a given text
- Each text belongs to exactly one category
- Predisdom set of categories

Single-Class Text Classification

- News stories into topics
- Divides texts into categories

Decision Trees

- Consistent, easy-to-understand, accurate, fast
- Learn a function f from examples (x, f(x))

Inductive Learning

Learning Algebras

Learning from Observations (Chapter 18, AIMA)

Review

TEXT CLASSIFICATION

- Algorithms - back to decision trees
- Feature Selection
- Features
- Training Classifiers
- The Classifier Interface
- Labelled Texts
- Using Classifiers
- Introduction

Announcements (Talk tomorrow, project questions)
Let’s begin with:

Examining a training corpus

How do we get ideas for this?

Which aspects of a sentence are relevant to our classification task?

<table>
<thead>
<tr>
<th>Features</th>
</tr>
</thead>
</table>

What do we do first?

Why is this text classification?

- questions
- imperatives (commands)
- statements

Categorise sentences as

| Task |

Using classifiers (Ch. 6)
What next?
See code examples
What if we wanted to switch classification algorithm?
Use the ClassifierTrainer to train a new classifier using the training corpus.
Build a ClassifierTrainer using the FeatureDetector list.

Training the Classifier

Training Corpus

What next?
See code examples
What about our project?
load and label

Using NLTK

The length of the sentence
The first word of the sentence
Which words are present in the sentence (bag of words)
Example: Features (from Tutorial)
See code examples.

*Tab эта  returns the list of category labels.
*classifyfY returns a labeled text token.

Requires two methods:

- `ClassifierInterface` for single category text classifiers.
- `LabeledTexts` for classes representing categorized text types.

**Classifying New Texts**

- Use the classifier we built to classify new texts.

**Tagging**

- **Cueing**
- **Multi-category classification**

**Relationship to Other Tasks**

- "Obvious number function; type/token distinction"
- "Depends by `classifier` module"
See code examples

device can then be used
created from Python function using the constructor

Python function needs to be provided in implementations

Features:

Returns: feature value
Input: labeled text
Single method: detector

Feature detectors for implementing feature detectors functions

Feature detectors

What are the values returned?

See code example

Map label/detectors to feature values

Features are defined by feature detectors functions

The label "sports"

Example feature: whether a document contains the word "ball" and has

relevant to deciding how likely a label is for that text (classification)

Each feature selection is a subset of a labeled text that provides do-

A way of encoding information used to build classifiers. This provides do-

Feature-Based Classification

Training

Classifiers

Training corpus (usually hand-labeled, assumed mostly correct)

Features

Boolean (binary) features

Others (not implemented in NLTK)

Integer features

Return new classifier

Input: list of labeled tokens

Single method: train

See code examples
See code examples

- the list contains one feature detector for each (val, l) pair
- inputs a list of relevant words and a list of labels
- whose detector functions have a specific form
- begin by implementing a feature detector using containing boolean features
  $\text{PassedFast}$. Each check which words are present in text

Implementations, continued

---

See code examples

- Efficient implementation for related feature detectors
  - unique within a feature detector list but globally
  - $N$-wise for a feature detector list with $N$ features
  - $\ldots$

Data structures that represent the feature detector functions for a set of

Feature Detector List Interface

---

Feature Detector List Interface for Implementing Feature Detector Lists

```python

```

```

---

Four methods

- add list operator
  - a detector method
  - a length operator
  - an indexing operator

---

Implementations of Feature Detector Lists

---

```python

```
See code examples

assessments returns (feature-id, feature-value) pairs for the test

default method

default method

4 output (old)

4 output (old)

Four methods

Feature Selection

Feature

Feature Value Lists

See reference documentation for other available implementations

one feature detector for each (val, l) pair

a list of recurrent words, and a list of labels

Begin Doublelist

one feature detector for each (val, l) pair

function defined over texts, a list of function values, and a list of labels

TextFeatureDoublelist

Comparison

Detector List

deleted from the feature value list corresponding with those in the feature
created with detector method of a Feature Detector List

Compare with Feature Detector Lists: data structures that represent the

Data structures that represent the feature values for a set of features
Instead we will return to decision trees.

For our purposes we will view NLTK Classifier Model Implementations (c).

Let's revisit how we started using classifiers.