Statistical Classifier

Classifying Dictionary Sense Definitions

Jason S. Chang
National Tsing Hua University

2019 0423

Introduction to statistical models

- Statistical models learn from data to build a model for prediction
  - Relate input and output in terms of formula and parameters
  - Estimate these parameters
- Two kinds of machine learning algorithms
  - Supervised learning
    - Training data = inputs and outputs (labels)
    - Need experts to annotate labels (expensive)
    - E.g., Maximum Entropy Model, Support Vector Machine, Decision Tree, Decision List, Transformation Rules
  - Unsupervised learning
    - Training data = inputs (no labels)
    - E.g., Expectation Maximization

分類器實例

- 前後分類的關係
  - 有關係（序列標示）：全句詞性標示、全句專有名詞標示
  - 無關係：搭配詞的詞性、 介詞偵錯改錯、電影評論之褒貶
- 分類別的個數
  - 二元分類器
    - 垃圾郵件分類（正常、垃圾）
    - 標點分類（句子結束，縮寫代號）
    - 介詞偵錯（句中的介詞是否正確？）
    - 電影評論之褒貶
    - 姓名之性別
  - 多元分類器
    - 介詞改錯（句中錯誤介詞，可以改為哪個介詞？）
  - 開放型分類器
    - 動詞改錯（句中錯誤動詞，可以改為哪個動詞？）

統計分類器範例—姓名之性別

- 原始資料

```
import nltk, random

names = ([(name, 'male') for name in nltk.corpus.names.words('male.txt')] +\
          [(name, 'female') for nltk.corpus.name in names.words('female.txt')])

random.shuffle(names)
```

- 設定特徵值

```
def gender_features(word):

   return {'last_letter': word[-1], 'first_4': word[:4]}


>>> print gender_features('Shrek')

{'last_letter': 'k', 'first_4': 'Shre'}
```

- 準備訓練、測試資料

```
featuresets = [(gender_features(n), g) for (n,g) in names]

train_set, test_set = featuresets[500:], featuresets[:500]
```

- 訓練分類器

```
classifier = nltk.NaiveBayesClassifier.train(train_set)
```

- 測試單筆資料

```
>>> classifier.classify(gender_features('Neo'))
'male'
```

- 測試整個測試集

```
>>> print nltk.classify.accuracy(classifier, test_set)
0.838
```

完整示範程式

```
import nltk, random
names = ([(name, 'male') for name in nltk.corpus.names.words('male.txt')] +\
           [(name, 'female') for name in nltk.corpus.names.words('female.txt')])
random.shuffle(names)
def gender_features(word):
    return {'last_letter': word[-1], 'first_4': word[:4]}
print gender_features('Shrek')
featuresets = [(gender_features(n), g) for (n,g) in names]
train_set, test_set = featuresets[500:], featuresets[:500]
classifier = nltk.NaiveBayesClassifier.train(train_set)
print classifier.classify(gender_features('Neo'))
print nltk.classify.accuracy(classifier, test_set)
```

Example: Pos/Neg Movie Reviews

- 原始資料

```
>>> import nltk, random
>>> nltk.corpus.movie_reviews.categories()
['neg', 'pos']
>>> documents = [ list(nltk.corpus.movie_reviews.words(fileid), category) \
                          for category in nltk.corpus.movie_reviews.categories() \
                          for fileid in nltk.corpus.movie_reviews.fileids(category)[:50] ]
>>> random.shuffle(documents)
```

- 設定特徵值

```
>>> all_words = nltk.FreqDist(w.lower() for w in nltk.corpus.movie_reviews.words())
>>> word_features = all_words.keys()[:2000]
>>> def document_features(document):
       features = dict([('has(%s)' %word, word in document) for word in word_features])
       return features
```

- 測試特徵值函數

```
>>> print document_features(movie_reviews('pos/cv957_8737.txt'))
{ 'has(may)': False, 'has(company)': False, 'has(lucas)': False, 'has(aliens)': False, ...}
```

- 準備訓練、測試資料

```
>>> featuresets = [(document_features(document), pos_neg) \
                        for (document, pos_neg) in documents]
>>> train_set, test_set = featuresets[:50], featuresets[50:]
```

- 訓練分類器

```
>>> classifier = nltk.NaiveBayesClassifier.train(train_set)
```

- 測試整個測試集

```
>>> print nltk.classify.accuracy(classifier, test_set)
0.72
```

最具資訊值之特徵值

```
>>> classifier.show_most_informative_features(10)
Most Informative Features
        has(outstanding) = True              pos : neg    =    10.6 : 1.0
              has(mulan) = True              pos : neg    =     8.3 : 1.0
             has(seagal) = True              neg : pos    =     8.2 : 1.0
        has(wonderfully) = True              pos : neg    =     7.3 : 1.0
              has(damon) = True              pos : neg    =     6.1 : 1.0
              has(flynt) = True              pos : neg    =     5.6 : 1.0
             has(wasted) = True              neg : pos    =     5.6 : 1.0
             has(poorly) = True              neg : pos    =     5.0 : 1.0
         has(ridiculous) = True              neg : pos    =     5.0 : 1.0
               has(lame) = True              neg : pos    =     5.0 : 1.0
```

```
import nltk, random
names = ([(name, 'male') for name in nltk.corpus.names.words('male.txt')] +\
        [(name, 'female') for name in nltk.corpus.names.words('female.txt')])
random.shuffle(names)
def gender_features(word):
  return {'last_letter': word[-1], 'first_4': word[:4]}
print gender_features('Shrek')
featuresets = [(gender_features(n), g) for (n,g) in names]
train_set, test_set = featuresets[500:], featuresets[:500]
classifier = nltk.NaiveBayesClassifier.train(train_set)
print classifier.classify(gender_features('Neo'))
print nltk.classify.accuracy(classifier, test_set)
```

完整示範程式

```
import nltk, random
print nltk.corpus.movie_reviews.categories()
documents = [ (nltk.corpus.movie_reviews.words(fileid), category) \
                              for category in nltk.corpus.movie_reviews.categories() \
                              for fileid in nltk.corpus.movie_reviews.fileids(category) ]
all_words = nltk.FreqDist(w.lower() for w in nltk.corpus.movie_reviews.words())
word_features = all_words.keys()[:2000]
def document_features(document):
    features = dict([('has(%s)' %word, word in document) for word in word_features])
    return features

print document_features(nltk.corpus.movie_reviews('pos/cv957_8737.txt'))

featuresets = [(document_features(document), category) for (document, category) in documents]

train_set, test_set = featuresets[100:], featuresets[:100]

classifier = nltk.NaiveBayesClassifier.train(train_set)

print nltk.classify.accuracy(classifier, test_set)
```

Reuters Corpus

- Contains 10,788 news documents (1.3 million words)
- The documents have been classified into 90 topics, and grouped into two sets, called "training" and "test“
- Categories overlap with each other, because a news story often covers multiple topics.

```
>>> from nltk.corpus import reuters

>>> reuters.fileids()

['test/14826', 'test/14828', 'test/14829', 'test/14832', ...]

>>> reuters.categories()

['acq', 'alum', 'barley', 'bop', 'carcass', 'castor-oil', 'cocoa', 'coconut', 'coconut-oil', 'coffee',
'copper', 'copra-cake', 'corn', 'cotton', 'cotton-oil', 'cpi', 'cpu', 'crude', 'dfl', 'dlr', ...]
```

Lab Work

- 目的：實作詞典語意定義的分類器（分類：劍橋詞典的 guidword 見上週講義）
- 輸入；輸出：劍橋詞典語意定義句；guideword
- 簡化：為考慮部分 guidword 分類過系，只篩選 Top-50 guidwords
- 特徵：(1) bag of words (2) word embeddings (所有定義詞的加總再求頻均)
- 改進：(1) 改變特徵，如高頻字、在定義中 guidewords、第一個名詞（動詞）
  (2) 類神經網路分類器
- 參考：
  - https://scikit-learn.org/stable/modules/generated sklearn.feature_extraction.text.CountVectorizer.html
  - https://nlp.stanford.edu/projects/glove/
  - https://scikit-learn.org/stable/modules/generated/ sklearn.svm.LinearSVC.html