



特征工程

(Feature Engineering)

刘远超

哈尔滨工业大学

计算机科学与技术学院

什么是特征工程？

- 引用维基百科上的定义

(https://en.wikipedia.org/wiki/Feature_engineering)

- **Feature engineering** is the process of using domain knowledge of the data to create features that make machine learning algorithms work.

- **引自知乎：**“数据和特征决定了机器学习的上限，而模型和算法只是逼近这个上限而已。”

- 深度学习也要用到特征，需要对输入的特征进行组合变换等处理。

自动分词

- 何谓自动分词？自动分词就是将用自然语言书写的文章、句段经计算机处理后，以词为单位给以输出，为后续加工处理提供先决条件。
- 举例：
 - “我来到北京清华大学。”
→ “我/ 来到/ 北京/ 清华大学/ 。”
 - “I came to Tsinghua University in Beijing.”
→ “I/ came/ to/ Tsinghua/ University/ in/ Beijing/ ./”
- 思考一下：中文的自动分词和英文的自动分词有何不同？

词根提取与词形还原

- **词根提取 (stemming)** : 是抽取词的词干或词根形式 (不一定能够表达完整语义) 。

- 原文: 'And I also like eating apple'

- 词根提取后: ['and', 'I', 'also', 'like', 'to', 'eat', 'appl']])

- **词形还原 (lemmatization)** : 是把词汇还原为一般形式 (能表达完整语义) 。如将“drove”处理为“drive”。

- 原文: 'And I also like eating apple'

- 词形还原后: ['And', 'I', 'also', 'like', u'eat', 'apple']])

词性标注

- 词性标注 (part-of-speech tagging) ¹: 是指为分词结果中的每个单词标注一个正确的词性的程序, 也即确定每个词是名词、动词、形容词或者其他词性的过程。

- 举例: “I like eating apple.”的词性标注结果为

[('I', 'PRP'), ('like', 'VBP'), ('eating', 'VBG'), ('apple', 'NN'), ('.', '.')]]

PRP personal pronoun I, he, she 人称代词

VBP verb, sing. present, non-3d take 动词 现在

VBG verb, gerund/present participle taking 动词 动名词/现在分词

NN noun, singular 'desk' 名词单数形式

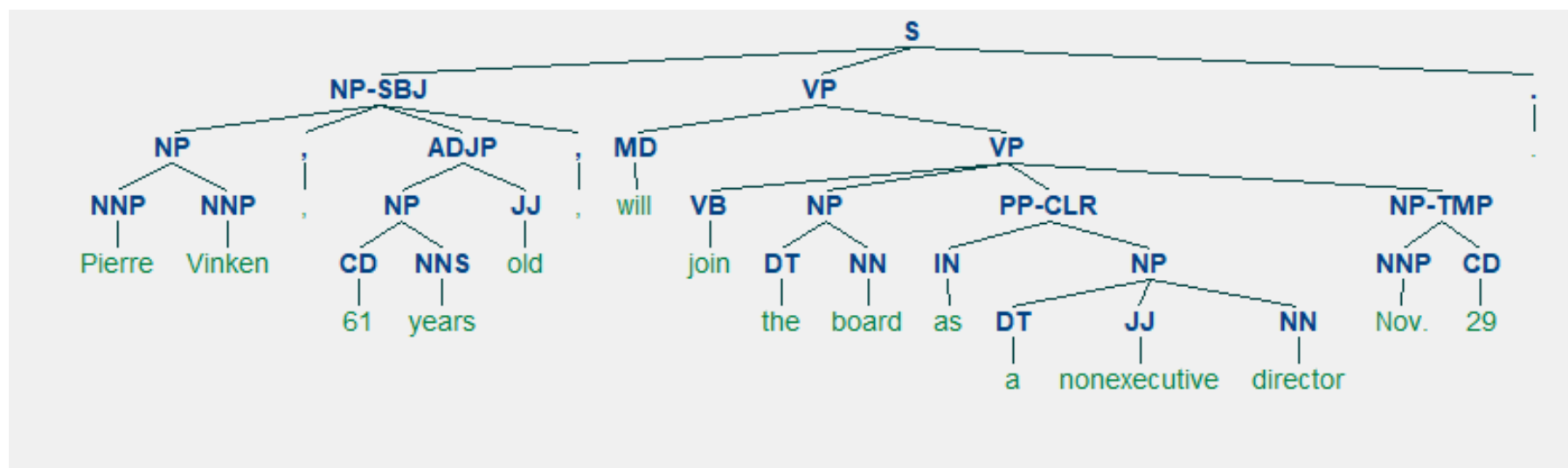
- 美国宾州树库词性标注规范:

http://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

1. 宗成庆《统计自然语言处理》清华大学出版社, 2013.8

句法分析

- 句法分析 (Syntactic analysis)：其基本任务是确定句子的句法结构或者句子中词汇之间的依存关系。



NLTK

NLTK 3.3 documentation

[NEXT](#) | [MODULES](#) | [INDEX](#)

Natural Language Toolkit

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to [over 50 corpora and lexical resources](#) such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active [discussion forum](#).

Thanks to a hands-on guide introducing programming fundamentals alongside topics in computational linguistics, plus comprehensive API documentation, NLTK is suitable for linguists, engineers, students, educators, researchers, and industry users alike. NLTK is available for Windows, Mac OS X, and Linux. Best of all, NLTK is a free, open source, community-driven project.

NLTK has been called “a wonderful tool for teaching, and working in, computational linguistics using Python,” and “an amazing library to play with natural language.”

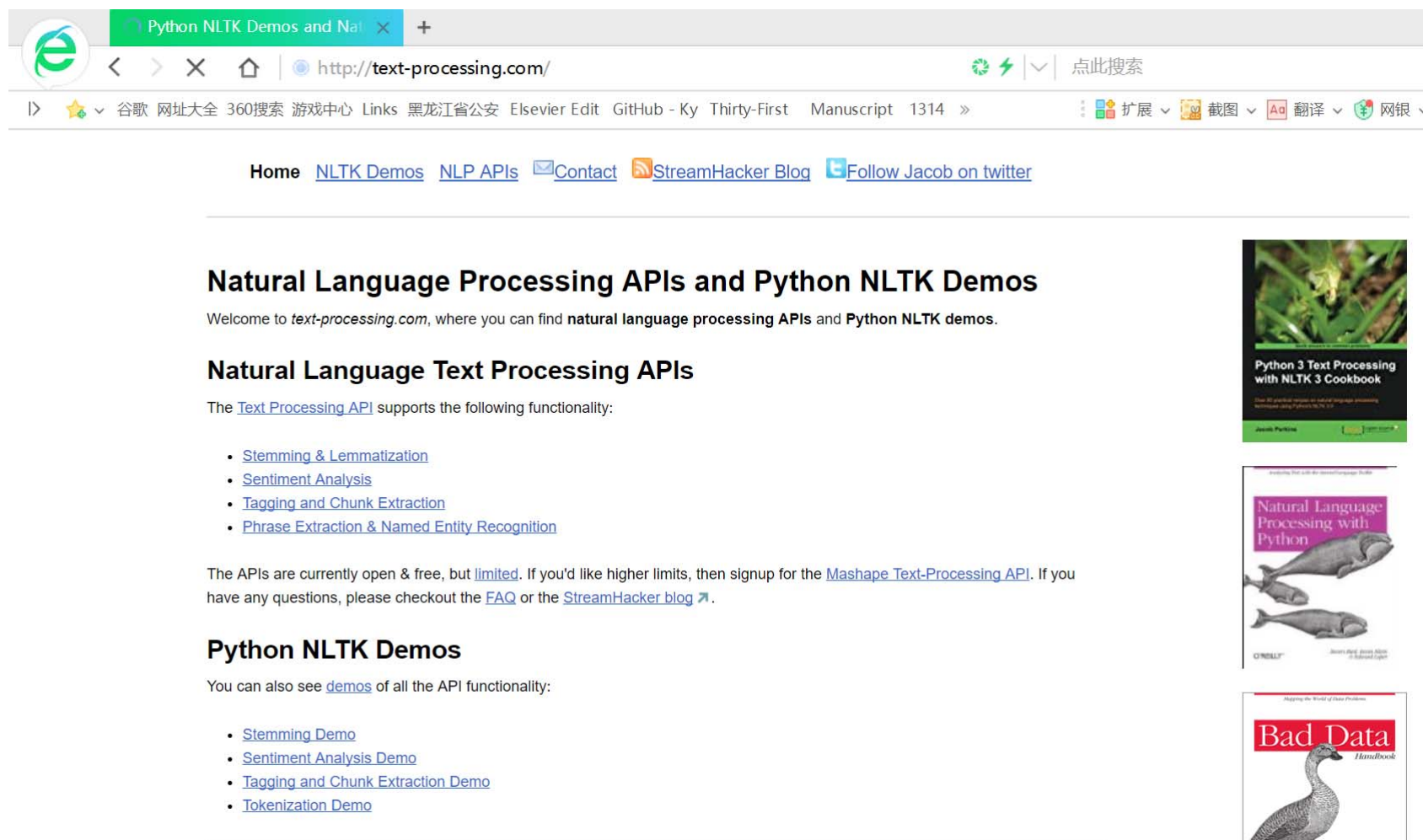
[Natural Language Processing with Python](#) provides a practical introduction to programming for language processing. Written by the creators of NLTK, it guides the reader through the fundamentals of writing Python programs, working with corpora, categorizing text, analyzing linguistic structure, and more. The online version of the book has been updated for Python 3 and NLTK 3. (The original Python 2 version is still available at

TABLE OF CONTENTS

- NLTK News
- Installing NLTK
- Installing NLTK Data
- Contribute to NLTK
- FAQ
- Wiki
- API
- HOWTO

SEARCH

Text Processing API



Python NLTK Demos and Nat... x +

http://text-processing.com/ 点此搜索

谷歌 网址大全 360搜索 游戏中心 Links 黑龙江省公安 Elsevier Edit GitHub - Ky Thirty-First Manuscript 1314 » 扩展 截图 翻译 网银

Home [NLTK Demos](#) [NLP APIs](#) [Contact](#) [StreamHacker Blog](#) [Follow Jacob on twitter](#)

Natural Language Processing APIs and Python NLTK Demos

Welcome to *text-processing.com*, where you can find **natural language processing APIs** and **Python NLTK demos**.

Natural Language Text Processing APIs

The [Text Processing API](#) supports the following functionality:


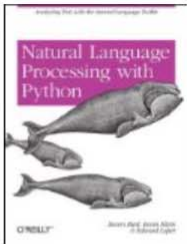
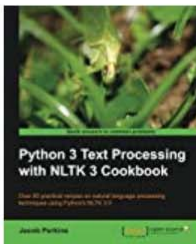
- [Stemming & Lemmatization](#)
- [Sentiment Analysis](#)
- [Tagging and Chunk Extraction](#)
- [Phrase Extraction & Named Entity Recognition](#)

The APIs are currently open & free, but [limited](#). If you'd like higher limits, then signup for the [Mashape Text-Processing API](#). If you have any questions, please checkout the [FAQ](#) or the [StreamHacker blog](#).

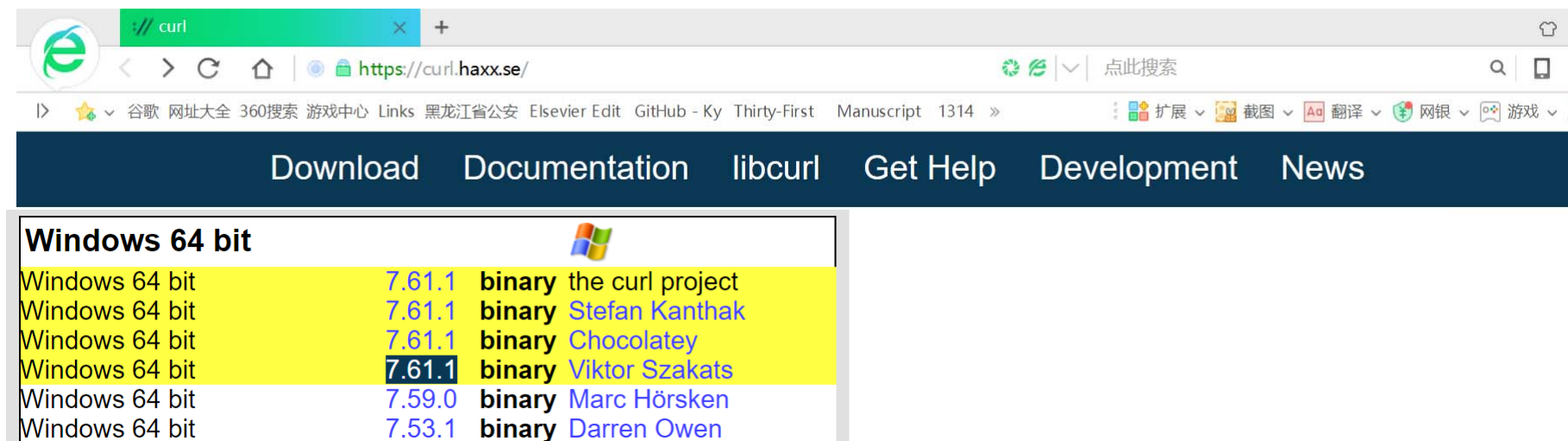
Python NLTK Demos

You can also see [demos](#) of all the API functionality:

- [Stemming Demo](#)
- [Sentiment Analysis Demo](#)
- [Tagging and Chunk Extraction Demo](#)
- [Tokenization Demo](#)



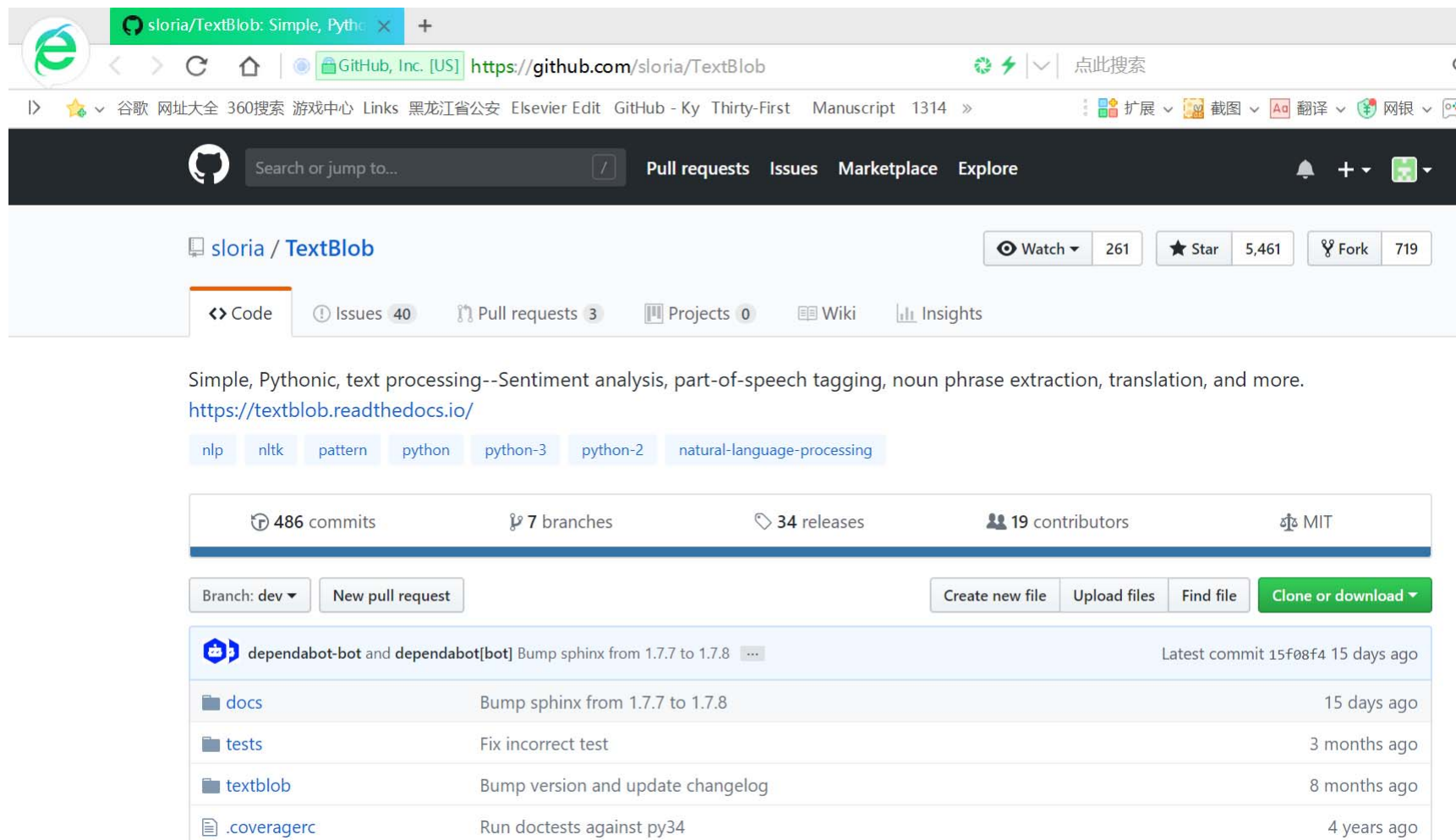
基于curl访问Text Processing API



```
$ curl -d "text=great" http://text-processing.com/api/sentiment/
```

```
{  
  "probability": {  
    "neg": 0.39680315784838732,  
    "neutral": 0.28207586364297021,  
    "pos": 0.60319684215161262  
  },  
  "label": "pos"  
}
```

TextBlob工具



sloria/TextBlob: Simple, Pythonic, text processing--Sentiment analysis, part-of-speech tagging, noun phrase extraction, translation, and more.

<https://textblob.readthedocs.io/>

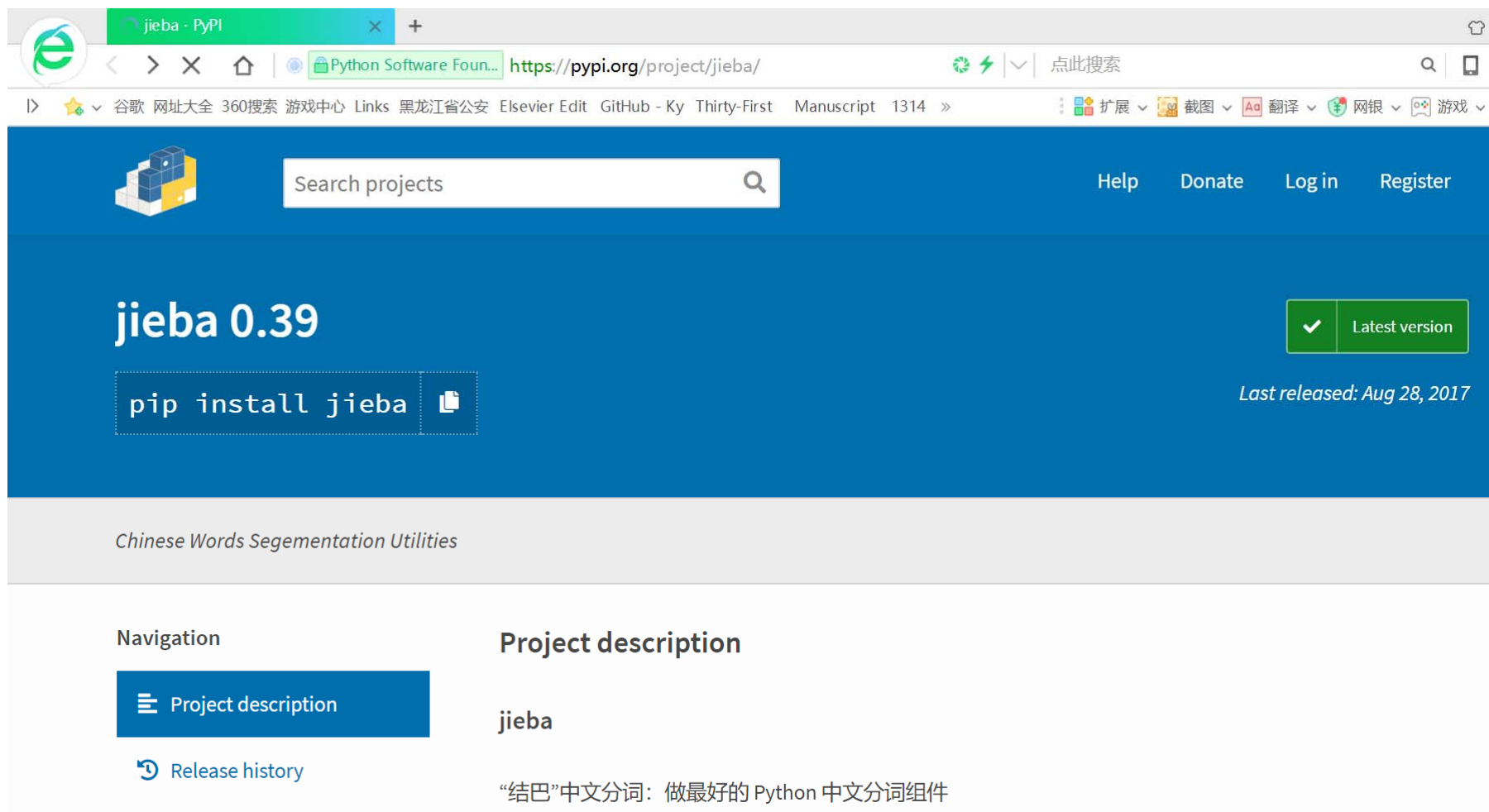
nlp nltk pattern python python-3 python-2 natural-language-processing

486 commits 7 branches 34 releases 19 contributors MIT

Branch: dev New pull request Create new file Upload files Find file Clone or download

dependabot-bot and dependabot[bot]	Bump sphinx from 1.7.7 to 1.7.8	Latest commit 15f08f4 15 days ago
docs	Bump sphinx from 1.7.7 to 1.7.8	15 days ago
tests	Fix incorrect test	3 months ago
textblob	Bump version and update changelog	8 months ago
.coveragerc	Run doctests against py34	4 years ago

中文处理工具jieba



The screenshot shows the PyPI project page for jieba. The browser's address bar displays the URL <https://pypi.org/project/jieba/>. The page header includes a search bar, navigation links (Help, Donate, Log in, Register), and a search bar. The main content area features the project name "jieba 0.39" and a green button labeled "Latest version". Below this, the command `pip install jieba` is shown in a dashed box. The page also indicates the last release date as "Aug 28, 2017". The footer section contains a navigation menu with "Project description" and "Release history", and a "Project description" section with the text "“结巴”中文分词：做最好的 Python 中文分词组件".

Search projects

Help Donate Log in Register

jieba 0.39

Latest version

`pip install jieba`

Last released: Aug 28, 2017

Chinese Words Segementation Utilities

Navigation

- Project description
- Release history

Project description

jieba

“结巴”中文分词：做最好的 Python 中文分词组件

Thanks!





向量空间模型及文本相似度计算

(Vector Space Model and Computation of Text Similarity)

刘远超

哈尔滨工业大学

计算机科学与技术学院

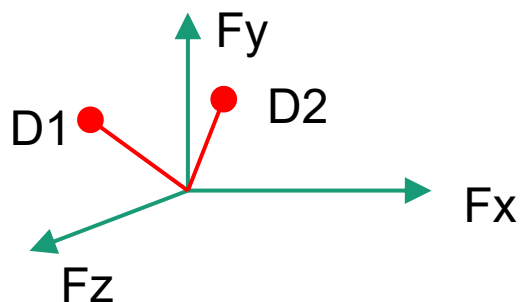
文档的向量化表示：BOW假设和VSM模型

为了便于计算文档之间的相似度，需把文档转成统一空间的向量。

- **BOW(bag-of-words model)**: 为了计算文档之间的相似度，假设可以忽略文档内的单词顺序和语法、句法等要素。将其仅仅看作是若干个词汇的集合。



- **VSM(Vector space model)**: 即向量空间模型。其是指在BOW词袋模型假设下，将每个文档表示成同一向量空间的向量。



BOW和VSM举例

假设有下面三个文档：

D1: 'Jobs was the chairman of Apple Inc, and he was very famous',

D2: 'I like to use apple computer',

D3: 'And I also like to eat apple'

- 类似这样一批文档的集合，通常也被称为**文集或者语料（corpus）**。

- 上述语料中，共有17个**不同的词**：

{0: 'also'; 1: 'and'; 2: 'apple'; 3: 'chairman'; 4: 'computer'; 5: 'eat'; 6: 'famous'; 7: 'he'; 8: 'inc'; 9: 'jobs'; 10: 'like'; 11: 'of'; 12: 'the'; 13: 'to'; 14: 'use'; 15: 'very'; 16: 'was'.}

- 因此可构造一个17维的向量空间：

Dim.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
D1	0	1	1	1	0	0	1	1	1	1	0	1	1	0	0	1	2
D2	0	0	1	0	1	0	0	0	0	0	1	0	0	1	1	0	0
D3	1	1	1	0	0	1	0	0	0	0	1	0	0	1	0	0	0

停用词

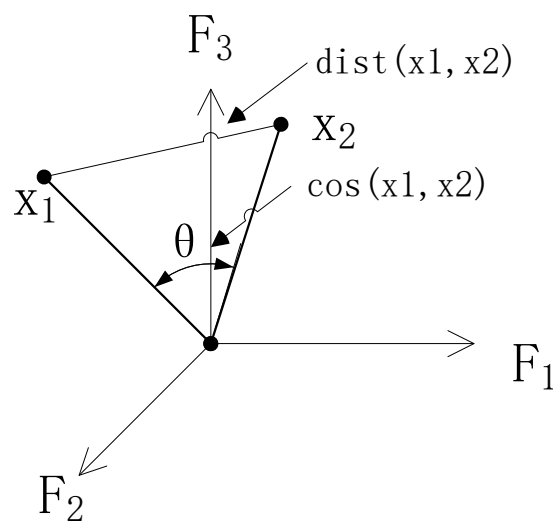
- 英文名称: Stop words
- 停用词通常**是非常常见且实际意义有限的词**，如英文中“the”，“a”，“of”，“an”等；中文中“的”、“是”、“而且”等。几乎可能出现在所有场合，因而对某些应用如信息检索、文本分类等区分度不大。
- 在信息检索等应用中，这些词在构建向量空间时通常会被过滤掉。因此这些词也被称为停用词。
- *note: 但在某些应用如短语搜索 phrase search 中，停用词可能是重要的构成部分，因此要避免进行停用词过滤。*

N-gram模型

- N-gram通常是指一段文本或语音中连续N个项目（item）的序列。
项目（item）可以是单词、字母、碱基对等。
- N=1时称为uni-gram，N=2称为bi-gram，N=3称为tri-gram，以此类推。
- 举例: 对于文本 ‘And I also like to eat apple’，则
 - **Uni-gram:** And, I, also, like, to, eat, apple
 - **Bi-gram:** And I, I also, also like, like to, to eat, eat apple.
 - **Tri-gram:** And I also, I also like, also like to, like to eat, to eat apple
- 20世纪80年代，N-gram被广泛地应用在拼写检查、输入法等应用中。
- 90年代以后，N-gram得到新的应用，如自动分类、信息检索等。即将连续的若干词作为VSM中的维度，用于表示文档。

文档之间的欧式距离

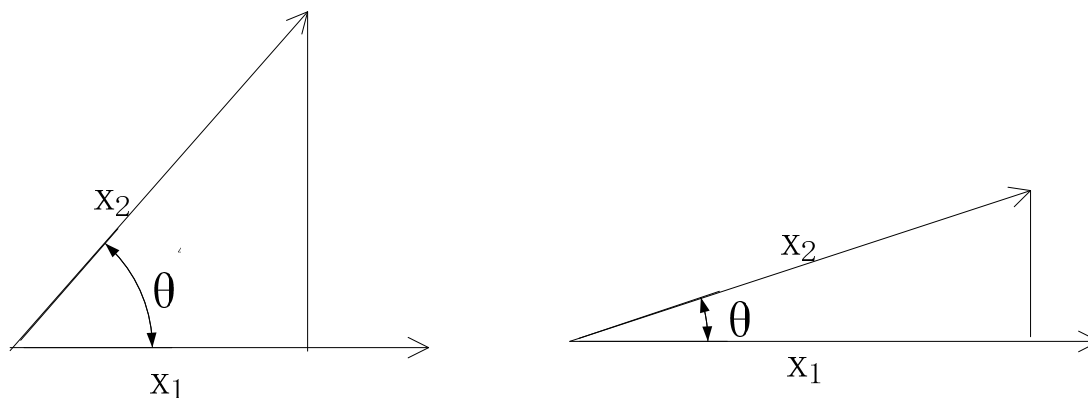
- 欧氏距离 (euclidean metric) 是一个通常采用的距离定义, 指在n 维空间中两个点之间的真实距离
- 公式: $d_{12} = \sqrt{\sum_{k=1}^n (x_{1k} - x_{2k})^2}$



文档之间的余弦相似度

- 余弦相似度，又称为余弦相似性，是通过计算两个向量的夹角余弦值来评估他们的相似度。
- 余弦值越接近1，就表明夹角越接近0度，也就是两个向量越相似，这就叫"余弦相似性"。

$$\bullet \cos(\vartheta) = \frac{\sum_{k=1}^n (x_{1k} \times x_{2k})}{\sqrt{\sum_{k=1}^n (x_{1k})^2} \times \sqrt{\sum_{k=1}^n (x_{2k})^2}} = \frac{x_1 \cdot x_2}{||x_1|| \times ||x_2||}$$



Tf-idf词条权重计算(1)

- 背景：特征向量里某些高频词在文集内其他文档里面也经常出现。它们往往太普遍，对区分文档起的作用不大。

- 例如：

- D1: 'Jobs was the chairman of Apple Inc.',

- D2: 'I like to use apple computer',

- 这两个文档都是关于苹果电脑的，则词条“apple”对分类意义不大。

- 因此有必要抑制那些在很多文档中都出现了的词条的权重。

- 在tf-idf 模式下，词条 t 在文档 d 中的权重计算为：

$$w(t) = tf(t, d) * idf(t)$$

其中， $tf(t, d)$ 表示为词条 t 在文档 d 中的出现频率， $idf(t)$ 表示与包含词条 t 的文档数目成反比（inverse document frequency）

Tf-idf词条权重计算(2)

- $idf(t)$ 怎么计算?

$$idf(t) = \left(\log \frac{n_d}{df(t)} + 1 \right)$$

- (optional) 数据平滑问题: 为了防止分母 $df(t)$ 为零

$$idf(t) = \left(\log \frac{1+n_d}{1+df(t)} + 1 \right)$$

Tf-idf 词条权重计算举例

counts = **[[3, 0, 1],**

[2, 0, 0],

[3, 0, 0],

[4, 0, 0],

[3, 2, 0],

[3, 0, 2]]

- 则第一个文档中的三个词条的 tf-idf 权重 可以如下计算:

$$1. \quad w(t) = tf(t, d) * \mathbf{idf(t)} = \mathbf{3} * \left(\log \frac{n_d}{df(t)} + 1 \right) = \mathbf{3} * \left(\log \frac{6}{6} + 1 \right) = \mathbf{3}$$

$$2. \quad w(t) = tf(t, d) * \mathbf{idf(t)} = \mathbf{0} * \left(\log \frac{n_d}{df(t)} + 1 \right) = \mathbf{0} * \left(\log \frac{6}{1} + 1 \right) = \mathbf{0}$$

$$3. \quad w(t) = tf(t, d) * \mathbf{idf(t)} = \mathbf{1} * \left(\log \frac{n_d}{df(t)} + 1 \right) = \mathbf{1} * \left(\log \frac{6}{2} + 1 \right) = \mathbf{2.0986}$$

- (可选) Then, applying the Euclidean (L2) norm, we obtain the

$$\text{following tf-idfs for document 1: } \frac{[3, 0, 2.0986]}{\sqrt{(3^2 + 0^2 + 2.0986^2)}} = [0.819, 0, 0.573]$$

Thanks!





特征处理 (特征缩放、选择及降维)

(Processing of Features -Feature Scaling, Feature Selection and Dimension Reduction)

刘远超

哈尔滨工业大学

计算机科学与技术学院

特征值的缩放

- **特征值缩放 (Feature Scaler) 也可以称为无量纲处理。**主要是对每个列，即同一特征维度的数值进行规范化处理。
- **应用背景：**
 - 不同特征（列）可能不属于同一量纲，即特征的规格不一样。例如，假设特征向量由两个解释变量构成，第一个变量值范围 $[0,1]$ ，第二个变量值范围 $[0,100]$ 。
 - 如果某一特征的方差数量级较大，可能会主导目标函数，导致其他特征的影响被忽略
- **常用方法**
 - 标准化法
 - 区间缩放法

特征值的缩放--标准化法

- 标准化的前提是特征值服从正态分布。
- 标准化需要计算特征的均值和标准差，公式表达为：

$$X_scale = \frac{(X(\text{axis}=0) - X.\text{mean}(\text{axis}=0))}{X.\text{std}(\text{axis}=0)}$$

- (相关知识) **标准差**：
 - 标准差 (Standard Deviation)，又常称均方差，用 σ 表示，是方差的算术平方根。
$$\sigma = \sqrt{\frac{1}{N} \sum_{i=0}^{N-1} (x_i - \mu)^2}$$
 - 标准差能反映一个数据集的离散程度。例如，两组数的集合 {0,5,9,14} 和 {5,6,8,9} 其平均值都是 7，但第二个集合具有较小的标准差。

特征值的缩放--区间缩放法

- 区间缩放法利用了边界值信息，将特征的取值区间缩放到某个特定范围。假设max和min为希望的调整后范围，则

$$X_scaled = \frac{(X(axis=0) - X.min(axis=0))}{(X.max(axis=0) - X.min(axis=0))} * (max - min) + min$$

- 由于希望的调整后范围一般为[0,1]。此时，公式变为

$$X_scaled = \frac{(X(axis=0) - X.min(axis=0))}{(X.max(axis=0) - X.min(axis=0))}$$

特征值的归一化

- 或称规范化 (Normalizer)

- 归一化是依照特征矩阵的行 (样本) 处理数据, 其目的在于样本向量在点乘运算或计算相似性时, 拥有统一的标准, 也就是说都转化为“单位向量”。即使每个样本的范式 (norm) 等于 1.

- 规则为L1 norm的归一化公式如下:

$$x' = \frac{x}{\sum_{j=0}^{n-1} |x_j|}$$

- 规则为L2 norm的归一化公式如下:

$$x' = \frac{x}{\sqrt{\sum_{j=0}^{n-1} x_j^2}}$$

定量特征的二值化

- 应用背景：对于某些定量特征，需要将定量信息转为区间划分。如将考试成绩，转为“及格”或“不及格”
- **方法：**设定一个阈值，大于或者等于阈值的赋值为1，小于阈值的赋值为0，公式表达如下：

$$x' = \begin{cases} 1, & x \geq threshold \\ 0, & x < threshold \end{cases}$$

缺失特征值的弥补计算

- 背景：数据获取时，由于某些原因，缺少某些数值，需要进行弥补。
- 常见的弥补策略：利用同一特征的均值进行弥补。

举例：

```
counts = [[1, 0, 1],  
           [2, 0, 0],  
           [3, 0, 0],  
           [NaN, 0, 0]]
```

则，NaN可以弥补为同列上其他数据的均值，即 $(1+2+3)/3=2$ 。

创建多项式特征

- 如果基于线性特征的模型不够理想，也可以尝试创建多项式特征。
 - 例如，两个特征 (X_1, X_2) ，它的平方展开式便转换成 $(1, X_1, X_2, X_1X_2, X_1^2, X_2^2)$ 。
 - 也可以自定义选择只保留特征相乘的多项式项。即将特征 (X_1, X_2) 转换成 $(1, X_1, X_2, X_1 * X_2)$ 。
- 得到多项式特征后，只是特征空间发生了变化。

特征选择

- **什么是特征选择?** 选择对于学习任务 (如分类问题) 有帮助的若干特征。
- **为什么要进行特征选择?** 1) 降维以提升模型的效率; 2) 降低学习任务的难度; 3) 增加模型的可解释性。
- **特征选择的角度:**
 - **特征是否发散:** 对于不发散的特征, 样本在其维度上差异性较小
 - **特征与目标的相关性:** 应当优先选择与目标相关性高的特征
- **几种常见的特征选择方法:**
 - 方差选择法
 - 皮尔逊相关系数法
 - 基于森林的特征选择法
 - 递归特征消除法

特征选择方法1--方差选择法

- **原理：** 方差非常小的特征维度对于样本的区分作用很小，可以剔除。
- 例如，假设数据集为布尔特征，想去掉那些超过80%情况下为1或者为零的特征。由于布尔特征是Bernoulli（伯努利）随机变量，其方差可以计算为 $Var[x] = p * (1 - p)$ ，因此阈值为 $.8 * (1 - .8) = 0.16$ ：

$X = \begin{bmatrix} 0, 0, 1, \\ 0, 1, 0, \\ 1, 0, 0, \\ 0, 1, 1, \\ 0, 1, 0, \\ 0, 1, 1 \end{bmatrix}$

第一列的方差为 $\frac{5}{6} \cdot \frac{1}{6} = 0.14$ ，小于0.16。因此可以被过滤掉。

特征选择方法2--皮尔森相关系数法

- 皮尔森相关系数(Pearson correlation coefficient)显示两个随机变量之间线性关系的强度和方向。计算公式为

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y}$$

其中, $cov(X,Y)$ 表示 X 和 Y 之间的协方差 (Covariance)

σ_X 是 X 的均方差, μ_X 是 X 的均值, E 表示数学期望

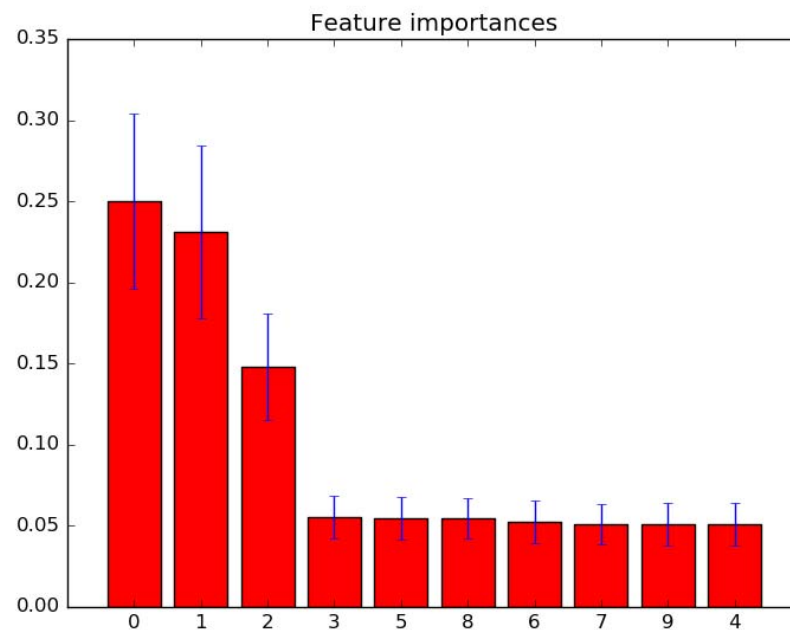
- 计算完毕后, 可以将与目标值相关性较小的特征过滤掉。
- *Note: Pearson 相关系数对线性关系比较敏感。如果关系是非线性的, 即便两个变量具有一一对应的关系, Pearson 相关性也可能会接近0。*

特征选择方法3--基于森林的特征选择

- 其原理是某些分类器，自身提供了特征的重要性分值。因此可以直接调用这些分类器，得到特征重要性分值，并排序。
- 本例中3个特征比较重要（informative），其他的分值较低。

Feature ranking:

1. feature 0 (0.250402)
2. feature 1 (0.231094)
3. feature 2 (0.148057)
4. feature 3 (0.055632)
5. feature 5 (0.054583)
6. feature 8 (0.054573)
7. feature 6 (0.052606)
8. feature 7 (0.051109)
9. feature 9 (0.051010)
10. feature 4 (0.050934)



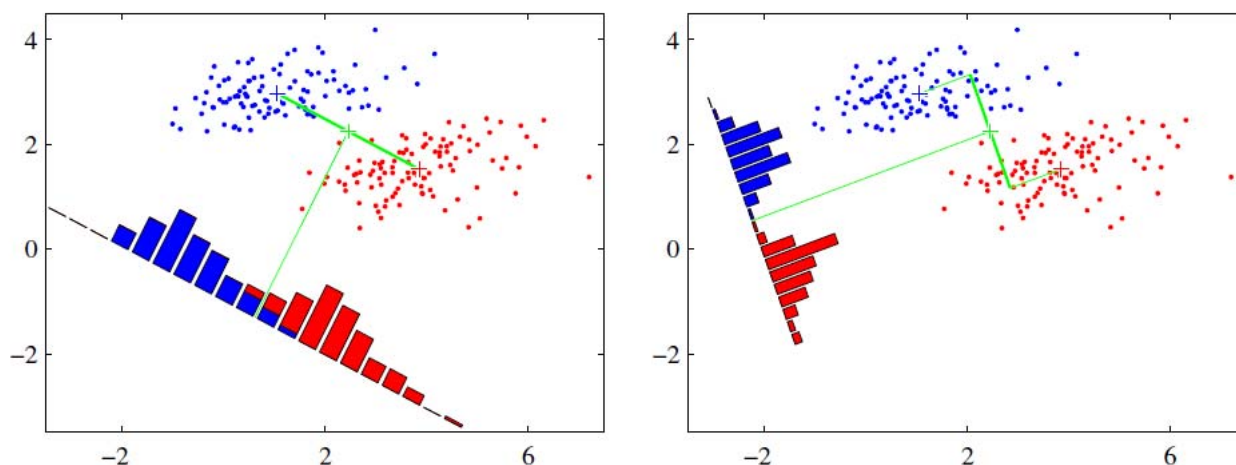
特征选择方法4--递归特征消除法

- （以 sklearn 中的函数为例）递归特征消除（recursive feature elimination, 即RFE）的基本步骤：
 1. 首先在初始特征或者权重特征集合上训练。通过学习器返回的 `coef_` 属性 或者 `feature_importances_` 属性来获得每个特征的重要程度。
 2. 然后最小权重的特征被移除。
 3. 这个过程递归进行，直到希望的特征数目满足为止。

特征降维--线性判别分析法 (LDA)

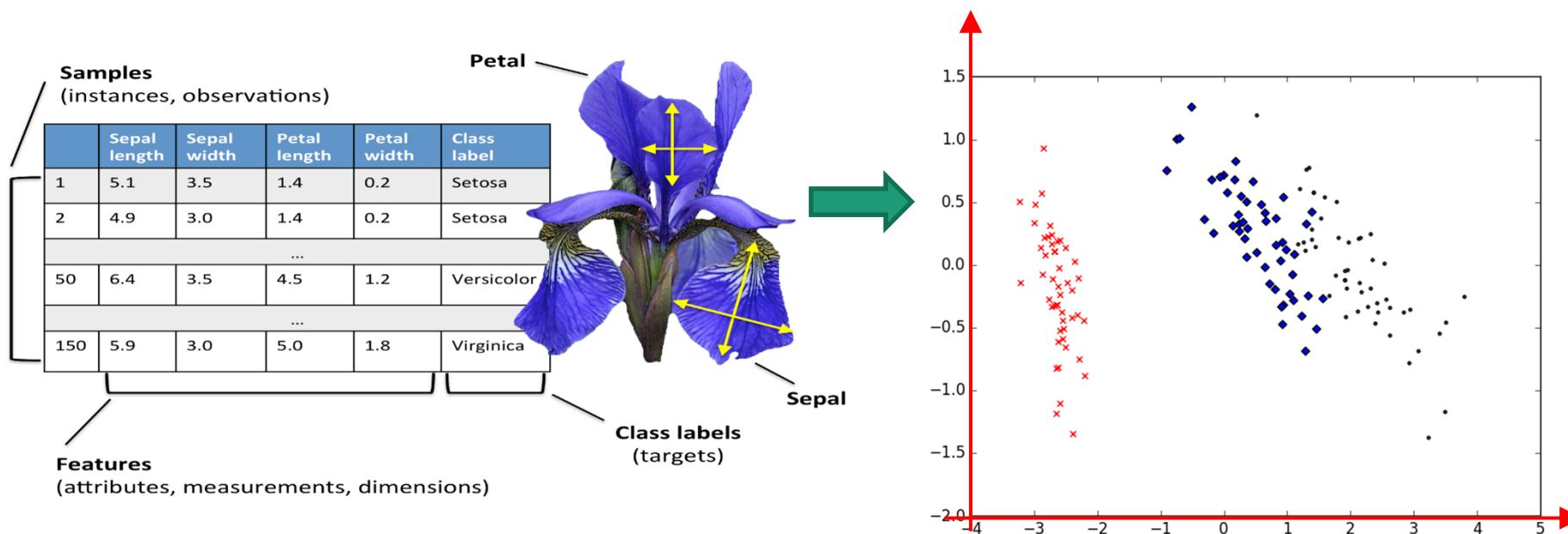
降维本质上是从一个维度空间映射到另一个维度空间。

- 线性判别分析 (Linear Discriminant Analysis, 简称LDA) 是一种监督学习的降维技术, 即数据集的每个样本有类别输出。
- LDA的基本思想: “投影后类内方差最小, 类间方差最大”。即将数据在低维度上进行投影, 投影后希望同类数据的投影点尽可能接近, 而不同类数据的类别中心之间的距离尽可能的大。



特征降维--主成分分析法 (PCA)

- 主成分分析 (principal component analysis) 是一种无监督的降维方法。
- PCA的基本思想是采用数学变换, 把给定的一组相关特征维度通过线性变换转成另一组不相关的维度 (即principal components), 这些新的维度按照方差依次递减的顺序排列: 形成第一主成分、第二主成分等等。



Thanks!

