# 文本特征提取

## 方法1.CountVectorizer

|  |
| --- |
|  |
|  |
|  |

### 实例

|  |
| --- |
| from sklearn.feature\_extraction.text import CountVectorizer  """ 文本特征提取 实例1  ["life is short,i like python","life is too long i dislike python"] """  data = ["life is short,i like like python", "life is too long i dislike python"] # 实例化转换器对象 transform = CountVectorizer() # 调用fit\_transform方法 ret\_data = transform.fit\_transform(data) print(ret\_data.toarray()) # 可以调用toarray方法把稀疏矩阵转化为一个ndarray |

### 效果

|  |
| --- |
|  |

### 注意:如果把data是数据改为中文句子,这个转换器会把一个句子中文特征,这样子非常不好,也就是说这个转换器只适合英文文本的处理

|  |
| --- |
|  |

### 当然如果把中文句子进行分词处理后,还是可以使用上面的转换器的

### 还有,这个转换器可以传递一个参数stop\_words是一个列表,列表里面的词会被忽略可以去网上搜索停用词表来查找这些停用词.

|  |
| --- |
| from sklearn.feature\_extraction.text import CountVectorizer  """ 文本特征提取 实例1  ["life is short,i like python","life is too long i dislike python"] """  data = ["life is short,i like like python", "life is too long i dislike python"] # data = ["人生苦短我喜欢她", "人生太长我不喜欢她"] # 用中文效果不好[[0 1] [1 0]] # 实例化转换器对象 transform = CountVectorizer(stop\_words=['is','too']) # 添加停用词列表把一些不太重要的词过滤掉 # 调用fit\_transform方法 ret\_data = transform.fit\_transform(data) print(ret\_data.toarray()) # 可以调用toarray方法把稀疏矩阵转化为一个ndarray print(transform.get\_feature\_names\_out()) # 可以调用toarray方法把稀疏矩阵转化为一个ndarray |

### 效果

|  |
| --- |
|  |

### 实例2.在上面的基础上添加结巴分词功能

|  |
| --- |
| # 结巴分词库 import jieba from sklearn.feature\_extraction.text import CountVectorizer   def cut\_text(text):  *"""  # 用jieba分词然后转化为列表再用" "连接成为一个字符串  :param text:  :return:  """* cutted = " ".join(list(jieba.cut(text)))  return cutted   """ 文本特征提取 实例2,利用jieba的cut方法来分词  ["一种还是一种今天很酷热，明天更残酷，后天很美好但绝对大部分是死在明天晚上，所以每个人不要放弃今天",  "我们看到的从很远星系来的光是在几百万年之前发出的这样当我们看到宇宙时,我们是在看它的过去",  "如果只用一种方式了解某种事物，你就不会真正了解它了解事物真正含义的秘密取决于如何将其与我们所了解的事物相联系"] """  data = ["一种还是一种今天很酷热，明天更残酷，后天很美好但绝对大部分是死在明天晚上，所以每个人不要放弃今天",  "我们看到的从很远星系来的光是在几百万年之前发出的这样当我们看到宇宙时,我们是在看它的过去",  "如果只用一种方式了解某种事物，你就不会真正了解它了解事物真正含义的秘密取决于如何将其与我们所了解的事物相联系"] # 对原始数据进行分词处理 new\_data = [] for line in data:  new\_data.append(cut\_text(line))  # 实例化转换器对象 transform = CountVectorizer() # 添加停用词列表把一些不太重要的词过滤掉 # 调用fit\_transform方法 ret\_data = transform.fit\_transform(new\_data) print(ret\_data.toarray()) # 可以调用toarray方法把稀疏矩阵转化为一个ndarray print(transform.get\_feature\_names\_out()) # 可以调用toarray方法把稀疏矩阵转化为一个ndarray |

#### 效果

|  |
| --- |
| [[2 0 1 0 0 0 2 0 0 0 0 0 1 0 1 0 0 0 0 1 1 0 2 0 1 0 1 1 0 0 0 1 1 0 0 1  0 1]  [0 0 0 1 0 0 0 1 1 1 0 0 0 0 0 0 0 1 3 0 0 0 0 1 0 0 0 0 2 0 0 0 0 0 1 0  1 0]  [1 1 0 0 4 2 0 0 0 0 1 1 0 1 0 1 1 0 1 0 0 1 0 0 0 1 0 0 0 2 1 0 0 1 0 0  0 0]]  ['一种' '不会' '不要' '之前' '了解' '事物' '今天' '光是在' '几百万年' '发出' '取决于' '只用' '后天' '含义'  '大部分' '如何' '如果' '宇宙' '我们' '所以' '放弃' '方式' '明天' '星系' '晚上' '某种事物' '残酷' '每个'  '看到' '真正' '秘密' '绝对' '美好' '联系' '过去' '还是' '这样' '酷热'] |

## 方法2.使用TfidfVectorizer,还是需要jieba库来分词,然后拼接

### 我们用来做特征词的词语应该由这么应该特点,就是在本篇文章中出现次数很多,但是在别的文章中出现的次数很少.

### 很显然CountVectorizer在这方面的效果不好

|  |
| --- |
|  |
| 上面第一篇文章可能是关于共享单车的,第二篇文章是财经类的文章,可见两个关键字正确,是非常方便处理数据的. |

### Tfidf工作原理

|  |
| --- |
|  |

### 计算方法说明

|  |
| --- |
|  |

### Api说明

|  |
| --- |
|  |

#### 注意:新版本的sklear的获取特征名称的函数是get\_feature\_names\_out()

### 实例3.

|  |
| --- |
| from sklearn.feature\_extraction.text import TfidfVectorizer import jieba  """ 文本特征提取 实例2,使用TfidfVectorizer  data = ["一种还是一种今天很酷热，明天更残酷，后天很美好但绝对大部分是死在明天晚上，所以每个人不要放弃今天",  "我们看到的从很远星系来的光是在几百万年之前发出的这样当我们看到宇宙时,我们是在看它的过去",  "如果只用一种方式了解某种事物，你就不会真正了解它了解事物真正含义的秘密取决于如何将其与我们所了解的事物相联系"] """   def cut\_text(text):  *"""  # 用jieba分词然后转化为列表再用" "连接成为一个字符串  :param text:  :return:  """* cutted = " ".join(list(jieba.cut(text)))  return cutted   data = ["一种还是一种今天很酷热，明天更残酷，后天很美好但绝对大部分是死在明天晚上，所以每个人不要放弃今天",  "我们看到的从很远星系来的光是在几百万年之前发出的这样当我们看到宇宙时,我们是在看它的过去",  "如果只用一种方式了解某种事物，你就不会真正了解它了解事物真正含义的秘密取决于如何将其与我们所了解的事物相联系"] proc\_data = [] for d in data:  proc\_data.append(cut\_text(d)) # 实例化转换器对象 transform = TfidfVectorizer() # 添加停用词列表把一些不太重要的词过滤掉 # 调用fit\_transform方法 ret\_data = transform.fit\_transform(proc\_data) print(ret\_data.toarray()) # 可以调用toarray方法把稀疏矩阵转化为一个ndarray print(transform.get\_feature\_names\_out()) # 可以调用toarray方法把稀疏矩阵转化为一个ndarray |

#### 效果

|  |
| --- |
| [[0.32200242 0. 0.21169724 0. 0. 0.  0.42339448 0. 0. 0. 0. 0.  0.21169724 0. 0.21169724 0. 0. 0.  0. 0.21169724 0.21169724 0. 0.42339448 0.  0.21169724 0. 0.21169724 0.21169724 0. 0.  0. 0.21169724 0.21169724 0. 0. 0.21169724  0. 0.21169724]  [0. 0. 0. 0.2410822 0. 0.  0. 0.2410822 0.2410822 0.2410822 0. 0.  0. 0. 0. 0. 0. 0.2410822  0.55004769 0. 0. 0. 0. 0.2410822  0. 0. 0. 0. 0.48216441 0.  0. 0. 0. 0. 0.2410822 0.  0.2410822 0. ]  [0.12826533 0.16865349 0. 0. 0.67461397 0.33730698  0. 0. 0. 0. 0.16865349 0.16865349  0. 0.16865349 0. 0.16865349 0.16865349 0.  0.12826533 0. 0. 0.16865349 0. 0.  0. 0.16865349 0. 0. 0. 0.33730698  0.16865349 0. 0. 0.16865349 0. 0.  0. 0. ]]  ['一种' '不会' '不要' '之前' '了解' '事物' '今天' '光是在' '几百万年' '发出' '取决于' '只用' '后天' '含义'  '大部分' '如何' '如果' '宇宙' '我们' '所以' '放弃' '方式' '明天' '星系' '晚上' '某种事物' '残酷' '每个'  '看到' '真正' '秘密' '绝对' '美好' '联系' '过去' '还是' '这样' '酷热'] |