クレジット:
UTokyo Online Educationデータマイニング入門 2018 森　純一郎

東京大学
THE UNIVERSITY OF TOKYO

CC BY NC ND

データマイニング入門 第5回

2018年度

テキスト分析の基礎

学習目標

- テキスト分析の基本的な処理の流れを理解する
- テキストの基本的な前処理を理解する
  - トークナイゼーション、ストップワード、ステミングなど
- 形態素とPOSタギングについて基本を理解する
- テキストのベクトル空間モデルについて基本を理解する
- 単語の重みづけについて基本を理解する
- テキスト間の類似度について基本を理解する
- Pythonで基本的なテキスト処理ができるようになる
  - tfidfとcos類似度の実装

データ行列

多次元データは行列（あるいはテンソル）とみなすことができる

- 多次元データ D
  - レコード数n
  - フィールド数 d
- Dはn ×dの行列で表現される
  - 行：レコード
  - 列：フィールド

文書-単語行列

- 行：文書
- 列：単語
  - 値：その文書におけるその単語の重要度
    - 単純には
      - 単語が出現するかしないかの0/1
      - 単語の出現回数

	データ	統計	分析
データを分析する	1	0	1
統計で分析する	0	1	1

文書集合を行列、各文書を単語（「データ」、「統計」、「分析」）を次元とするベクトルとして表現

テキスト分析の流れ

テキスト集合の取得
- コーパス構築

前処理
- トークナイゼーション
- 形態素解析

ベクトル化
- ベクトル空間モデル
- 単語の抽出と重み付け

ベクトル・行列演算
- 文書検索、文書分類、文書推薦など

自然言語処理の流れ

形態素解析

- テキストを単語に分割する

構文解析

- 単語間の構文的関係を決定する
  - 格解析など

意味や文脈解析

- 単語や文の意味を決定する
- 複数の文間の関係を決定する
  - 照応や省略、談話構造解析など

テキスト分析・言語処理の応用

- 自動分類
- 自動翻訳
- 自動要約
- 質問応答
- 情報検索
- 情報推薦
- 音声認識
- 感情分析
- ソーシャルメディア分析　など

コーパス

テキスト（文書）の集合

- 「目的」に応じて収集、構築される
  - 対象をよく表すようなテキストのサンプル集合
    - 新聞記事
    - 論文
    - 特許
    - カルテ
    - Wikipedia
    - ウェブ など
- 構造化、言語情報を付加することもある
  - 品詞
  - 構文構造、係り受け
  - 語義

など

日本語のコーパスの例

Wikipedia

- https://dumps.wikimedia.org/jawiki/

新聞記事コーパス（有料）

- http://www.nichigai.co.jp/sales/corpus.html

日本語コーパス

- 国立国語研究所
- https://pj.ninjal.ac.jp/corpus_center/
  - ウェブコーパス
    - https://pj.ninjal.ac.jp/corpus_center/nwjc/

特許コーパス

- https://alaginrc.nict.go.jp/resources/jpo-info/jpo-outline.html

APIによるテキストデータ収集

HTTPリクエストに基づくデータのリクエストと受け取り（REST API）

データ形式はJSONかXMLであることが多い

- JSONファイルの処理はPythonプログラミング入門参照
- TwitterのTweetデータ
  - https://developer.twitter.com/en/docs.html
- Googleの検索結果データ
  - https://developers.google.com/custom-search/v1/overview
- Wikipediaデータ
  - https://www.mediawiki.org/wiki/API:Main_page/ja
- 朝日新聞の記事データ
  - http://www.asahi.com/shimbun/medialab/webapi.html
- 論文データ
  - https://dev.elsevier.com/

スクレイピング

ウェブページからデータを自動収集

- ウェブクローラーと情報抽出

トークナイゼーション (TOKENIZATION)

テキスト分析の最初のステップ

テキストを「トークン」の集合に分割する
- トークンは単語でも記号、数字でもよい

英語の場合は、単純にスペースやコンマで区切ればよい
- It is very hot today.→ [it, is, very, hot, today]
- 頻出するトークンをストップワードとして除くことがある
  - a, the, など
- ステミング・レンマタイゼーション
  - 三単現、進行形、過去形を1つの語幹に統一する
    - laughing, laughs, and laughed → laugh-
  - ポーターアルゴリズムなど

日本語の場合はどのように区切ればよいだろうか？
- 今日はとても暑いです。→？

不要語（ストップワード）

頻出するトークンは処理上不要となることがある
- 日本語の助詞（は、が、など）や英語の冠詞（a, the, など）

一般に機能語（日本語の助詞・助動詞、英語の冠詞・前置詞など）は不要語となりやすい

形態素解析（後述）によりトークンの品詞を特定することで不要語と判断できる

機能語ではなくとも頻度があまりに多いトークンは不要語となりやすい

処理の上ではあらかじめ不要語リストを用意しておくことが多い
- 各言語の不用語リスト
  - https://www.ranks.nl/stopwords

形態素 (PART OF SPEECH)

形態素

- 意味を持つ表現要素の最小単位（言語学）
- より直感的には、類似した文法的役割を示す語のクラス
  - 名詞、動詞、形容詞、副詞, 前置詞など

形態素(POS)タグ

- 形態素を表すタグの集合
  - 名詞
    - 単数名詞：NN, 複数名詞：NNS, 固有名詞:NP など
  - 動詞
    - 現在形：VB, 過去形：VBD など
  - 形容詞
    - 形容詞：JJ, 比較級：JJR, 最上級：JJS など
  - 副詞
    - 副詞：RB など

形態素解析

文書を単語に分割し、各単語に品詞や語形などの情報を付与する

アノテーション
- 人手による形態素タグ付け…大変

POSタギンング
- コンピュータによる自動の形態素タグ付け
  - 系列ラベリングという問題を解く
    - 与えられた単語（トークン）の系列に対して尤もらしいPOSタグの系列を予測する
  - 具体的には隠れマルコフモデルやCRFなどで条件確率をモデル化

x	time	flies	like	an	arrow
y_1	NN	VBZ	IN	DT	NN
y_2	VB	NNS	IN	DT	NN
y_3	NN	NNS	VBP	DT	NN

$$argmax_y P(y|x)$$

本講義では、データ分析・データマイニングの基礎について学ぶとともに演習を通して実際にデータを分析するプロセスを学ぶ。
本　接頭詞,名詞接続,*,*,*,*,本,ホン,ホン
講義　名詞,サ変接続,*,*,*,*,講義,コウギ,コーギ
で　助詞,格助詞,一般,*,*,*,で,デ,デ
は　助詞,係助詞,*,*,*,*,は,ハ,ワ
、　記号,読点,*,*,*,*,、,、,、
データ　名詞,一般,*,*,*,*,データ,データ,データ
分析　名詞,サ変接続,*,*,*,*,分析,ブンセキ,ブンセキ
・　記号,一般,*,*,*,*,・,・,・
データ　名詞,一般,*,*,*,*,データ,データ,データ
マイニング　名詞,サ変接続,*,*,*,*,マイニング,マイニング,マイニング
の　助詞,連体化,*,*,*,*,の,ノ,ノ
基礎　名詞,一般,*,*,*,*,基礎,キソ,キソ
について　助詞,格助詞,連語,*,*,*,について,ニツイテ,ニツイテ
学ぶ　動詞,自立,*,*,五段・バ行,基本形,学ぶ,マナブ,マナブ
とともに　助詞,格助詞,連語,*,*,*,とともに,トトモニ,トトモニ
演習　名詞,サ変接続,*,*,*,*,演習,エンシュウ,エンシュー
を通して　助詞,格助詞,連語,*,*,*,を通して,ヲトオシテ,ヲトーシテ
実際　副詞,助詞類接続,*,*,*,*,実際,ジッサイ,ジッサイ
に　助詞,副詞化,*,*,*,*,に,ニ,ニ
データ　名詞,一般,*,*,*,*,データ,データ,データ
を　助詞,格助詞,一般,*,*,*,を,ヲ,ヲ
分析　名詞,サ変接続,*,*,*,*,分析,ブンセキ,ブンセキ
する　動詞,自立,*,*,サ変・スル,基本形,する,スル,スル
プロセス　名詞,一般,*,*,*,*,プロセス,プロセス,プロセス
を　助詞,格助詞,一般,*,*,*,を,ヲ,ヲ
学ぶ　動詞,自立,*,*,五段・バ行,基本形,学ぶ,マナブ,マナブ
。　記号,句点,*,*,*,*,。,。,。
EOS

形態素解析器

英語

- Stanford POS Tagger
  - http://nlp.stanford.edu/software/tagger.shtml

日本語

- MeCab
  - http://taku910.github.io/mecab/
- Chasen
  - http://chasen-legacy.osdn.jp/
- JUMAN
  - http://nlp.ist.i.kyoto-u.ac.jp/index.php?JUMAN

PYTHONでMECABを使う

MeCab, 辞書(mecab-ipadicなど），Pythonバイディング (mecab-python3)をインストール

```
import MeCab
t = MeCab.Tagger("-d /usr/local/lib/mecab/dic/mecab-ipadic-neologd")
text="本講義ではデータマイニングを学ぶ。"
t.parse('')
node = t.parseToNode(text)
while node:
    word = node.surface
    pos = node.feature
    print(word, pos)
    node = node.next
```

```
 BOS/EOS,*,*,*,*,*,*,*,*
本 接頭詞,名詞接続,*,*,*,*,本,ホン,ホン
講義 名詞,サ変接続,*,*,*,*,講義,コウギ,コーギ
で 助詞,格助詞,一般,*,*,*,で,デ,デ
は 助詞,係助詞,*,*,*,*,は,ハ,ワ
データマイニング 名詞,固有名詞,一般,*,*,*,データマイニング,データマイニング,データマイニング
を 助詞,格助詞,一般,*,*,*,を,ヲ,ヲ
学ぶ 動詞,自立,*,*,五段・バ行,基本形,学ぶ,マナブ,マナブ
。 記号,句点,*,*,*,*,。,。,。
 BOS/EOS,*,*,*,*,*,*,*,*
```

ベクトル空間モデル

文書に含まれる各単語の重みを要素とするベクトルで文書を表現（文書ベクトル）

- Bag-of-Words モデルとも呼ばれる

- $D_i=[w_{i1}, w_{i2}, \ldots, w_{in}]$
- w_{ij}は文書D_iにおける単語w_jの重み
- ベクトルの各次元は単語に対応

文書集合全体は以下のような文書・単語行列として表現できる

- $D=[D_1,D_2,\ldots, D_n]=[[w_{11}, w_{12}, \ldots, w_{1n}], [w_{21}, w_{22}, \ldots, w_{2n}], \ldots, [w_{n1}, w_{n2}, \ldots, w_{nn}]]$

	データ	統計	分析
データを分析する	1	0	1
統計で分析する	0	1	1

文書集合を行列、各文書を単語（「データ」、「統計」、「分析」）を次元とするベクトルとして表現

単語の重み付け

文書における単語の重要度を重み付け

- 局所的な重み
  - 文書内における単語の出現頻度に基づく重み
    - 頻繁に出現する単語は重みが大きくなる
    - 代表的なもの：TF (term frequency)
- 大域的な重み
  - 文書集合（コーパス）全体における単語の分布（偏り）に基づく重み
    - 特定の文書に偏って出現する単語は重みが大きくなる
    - 代表的なもの：IDF (inverse document frequency)

文書正規化

- 文書が長いほど、含まれる単語の数も増えるため、重みに文書の長さの影響を減らすための正規化
  - コサイン正規化
    - 文書に含まれるすべての単語の重みの2乗和で各単語の重みを割る

TFIDF

TFIDF (term frequency-inverse document frequency)
単語の重み付け方法

- TF (Term Frequency)
  - テキストにおける単語の出現頻度
    - 1+log(TF)とすることもある
- DF (Document Frequency)
  - 単語を含むテキストの数
- IDF (Inverse Document Frequency)
  - DFの逆数
  - log N/DF(Nはテキストの総数)
    - logをとるのは値の変化をゆるやかにするため
    - 0を避けるためlog(N/DF)+1とすることもある
- TFIDF
  - TF*IDF
    - 他のテキストにはあまり出現しないがそのテキストにはよく出現するような単語に重み付け

文書ベクトルの例

以下のテキストを「データ」と「分析」の2次元のベクトルで表現する

- 「本講義では、**データ分析**・**データ**マイニングの基礎について学ぶとともに演習を通して実際に**データ**を**分析**するプロセスを学ぶ。」

データは3回、分析は2回、テキスト中に出現するのでTF重みのベクトルは：

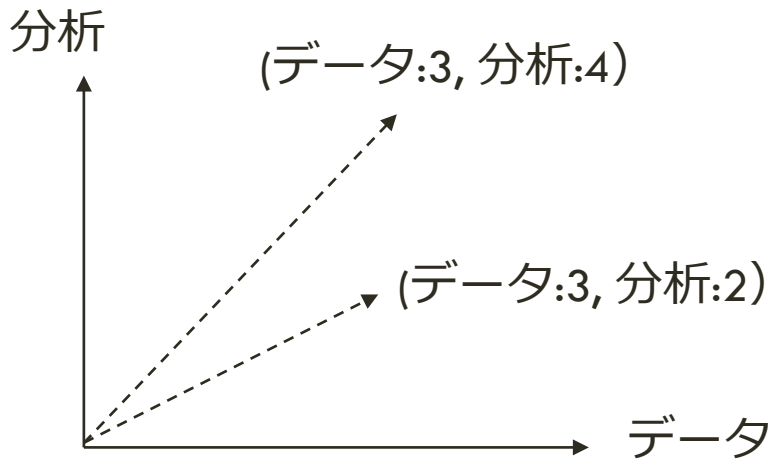
- (データ:3, 分析:2)

ここで、テキストの総数を100,「データ」を含むテキスト数が50,「分析」を含むテキスト数を25とすると「データ」と「分析」のTFIDF重みは

- データ：$3*\log_2(100/50)=3$, 分析：$2*\log_2(100/25)=4$
  - * 簡単のためlogの底を2としている

テキストのTFIDF重みベクトルは：

- (データ:3, 分析:4)
  - 「分析」はDFが相対的に小さく特定のテキストにしか出現しないため重みが大きくなった



ベクトル演算の復習

ベクトル

$$\vec{x} = (x_1, x_2, \cdots, x_n) \qquad \vec{y} = (y_1, y_2, \cdots, y_n)$$

ベクトルの内積

$$\vec{x} \cdot \vec{y} = x_1 y_1 + x_2 y_2, \cdots, x_n y_n$$

ベクトルの大きさ（L2ノルム）

$$\|x\|_2 = \sqrt{x_1^2 + x_2^2 +, \cdots, x_n^2}$$

cos類似度

ベクトルxとyのなす角をϑとすると以下が成り立つ

$$cos\theta = \frac{\vec{x}\cdot\vec{y}}{\|x\|_2\|y\|_2}$$

cos類似度はベクトル空間モデルにおける文書ベクトル間の類似度を与える

- 2つのベクトルの内積をそれぞれの長さで割ったもの
- 類似度の値は[-1:1]だがベクトルの重みは通常正なので類似度の値は[0:1]

類似度が高ければ、それらのテキストは関連している

- テキストの類似度は、テキスト分類、検索、推薦の基礎

COS類似度の例

テキストBとCのcos類似度

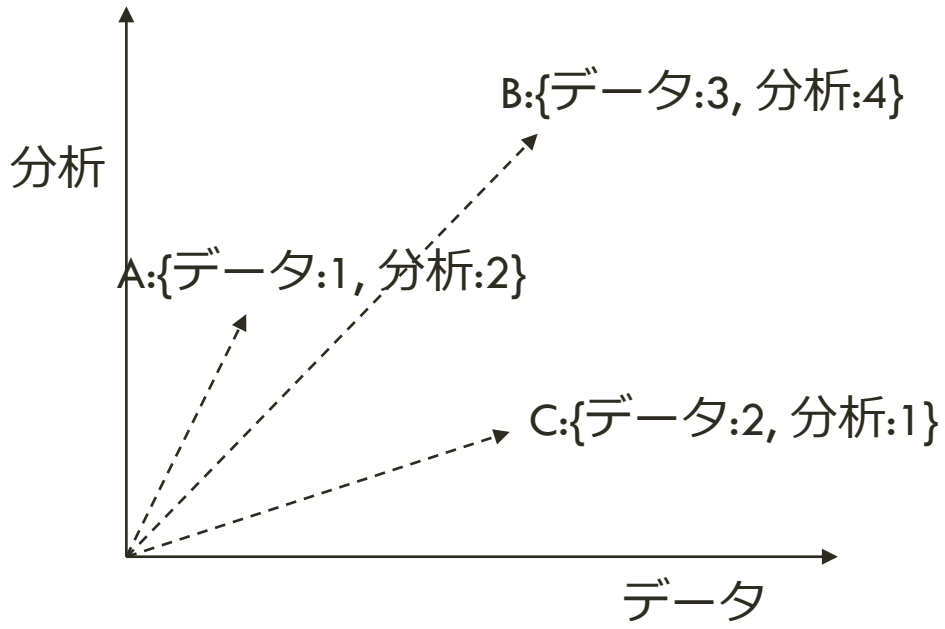
- 3*2+4*1/(√3*3+4*4)(√2*2+1*1)=10/(√25*√5)=2/ √5

テキストAとBのcos類似度

- 3*1+4*2/ /(√3*3+4*4)(√2*2+1*1)=11/(√25*√5)=2.2/ √5

Aの方がCよりBと関連している

- （この2次元のベクトル空間においては）



類似度と距離尺度

距離の公理

- 非負：距離は負にならない
- 距離が0ならばAとBは同じ点、 AとBは同じ点なら距離は0
- 対称：AからBの距離tとBからAの距離は等しい
- 三角不等式：AからCを経由したBへ距離は、AからBへの距離以上

ユークリッド距離

$$\Sigma_{i=1}^{n}(x_i - y_i)^2$$

マンハッタン距離

$$\Sigma_{i=1}^{n}|x_i - y_i|$$

マハラノビス距離

$$\sqrt{(\vec{x} - \vec{y})^T \Sigma^{-1} (\vec{x} - \vec{y})}$$

cos類似度とユークリッド距離

cos類似度は2つのベクトルの要素同士の相関を計算し、それを2つのベクトルのユークリッド空間における長さで割ることでそれぞれのベクトルを正規化した**正規化相関係数**とみなせる

正規化された（長さが1）のベクトル同士のcos類似度は：

$$cos\theta = \vec{x} \cdot \vec{y}$$

一方、これらのベクトル間のユークリッド距離の2乗は：

$$(|\vec{x} - \vec{y}|)^2 = \Sigma_{i=1}^{n} (x_i - y_i)^2 = 2(1 - \vec{x} \cdot \vec{y})$$

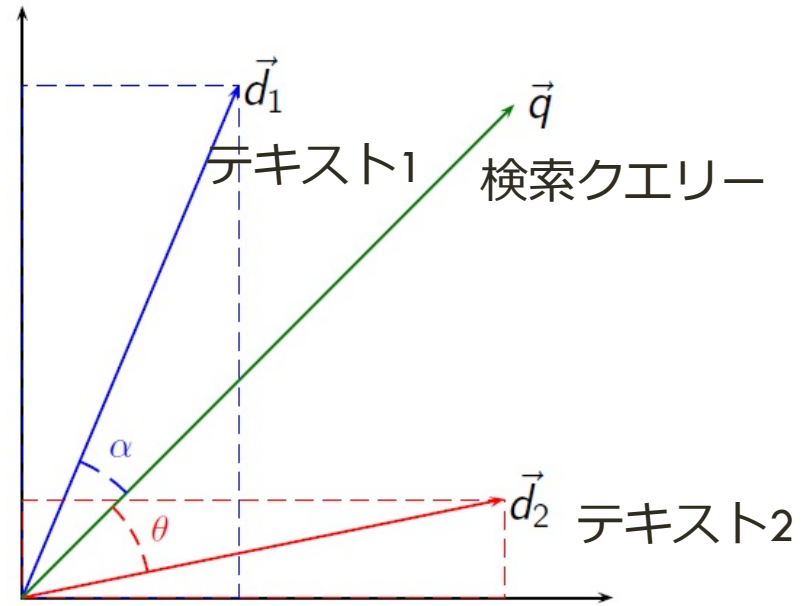
正規化されたベクトルにおいてはcos類似度による近さはユークリッド距離による近さと等しい

ベクトル空間モデルと情報検索

検索クエリー、検索対象テキストがそれぞれベクトル化されているとする

情報検索は、ベクトル空間におけるクエリーベクトルとテキストベクトルの類似度計算によって実現できる

- クエリーとcos類似度が近いテキストほど関連するテキスト



Vector space model.jpg
https://commons.wikimedia.org/wiki/File:Vector_space_model.jpg
CC BY 3.0

単語共起行列

文書単語行列D（1or0の要素）が与えられた時、D^TDで与えられる行列の要素c_{ij}は単語iと単語jがいくつの文書で共起したかを表す

- 共起の範囲は文書に限らず、段落や文としてもよい
- 単語同士の共起する度合いが大きければ、それらの単語は関連がある

文書単語行列（1or0の要素）の2つの単語に対応する列をそれぞれ集合X, Yと考えると単語の共起は以下の一致係数と考えられる

- 一致係数 $|X \cap Y|$

D	データ	統計	分析
データを分析する	1	0	1
統計で分析する	0	1	1

D^TD	データ	統計	分析
データ	1	0	1
統計	0	1	1
分析	1	1	2

単語共起行列

集合演算により他にも以下のような共起尺度が定義可能

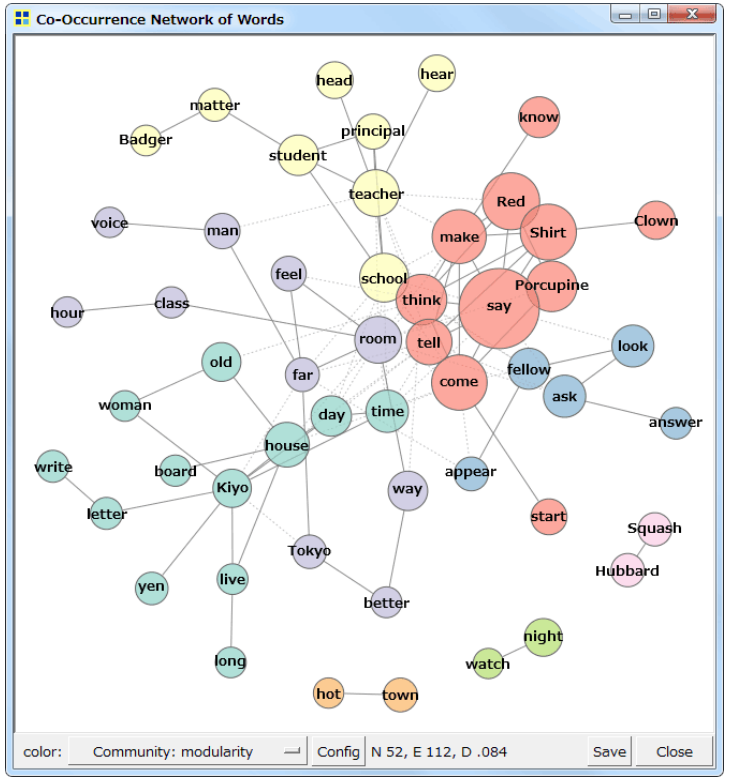
- Dice係数 $\frac{2|X \cap Y|}{|X|+|Y|}$
- Jaccard係数 $\frac{|X \cap Y|}{|X \cup Y|}$
- 重複係数 $\frac{|X \cap Y|}{min(|X|,|Y|)}$
- この他に相互情報量、対数尤度比、tスコア、カイ二乗値なども共起尺度として使われる

一般に単語共起行列の要素はこれらの単語間の共起尺度を値として持つ

共起ネットワーク

単語の共起ネットワーク

- 単語間の共起に基づき単語間の関係をネットワークとして表したもの
  - 共起の値がネットワークのリンクの重みとなる
- テキストの可視化や重要語の抽出などに用いられる
  - 次回のネットワーク分析の基礎



A co-occurrence network created with KH Coder
https://en.wikipedia.org/wiki/Co-occurrence_network

単語間の類似度

単語同士の意味が似ているとは？

- 単語の分布仮説：単語の意味はその単語と共に出現している単語群によって特徴付けられる
  - 2つの単語に共通して共起している単語が多ければそれらの単語は類似している

単語同士の類似度は単語共起行列の行（あるいは列）同士の類似度で計算できる

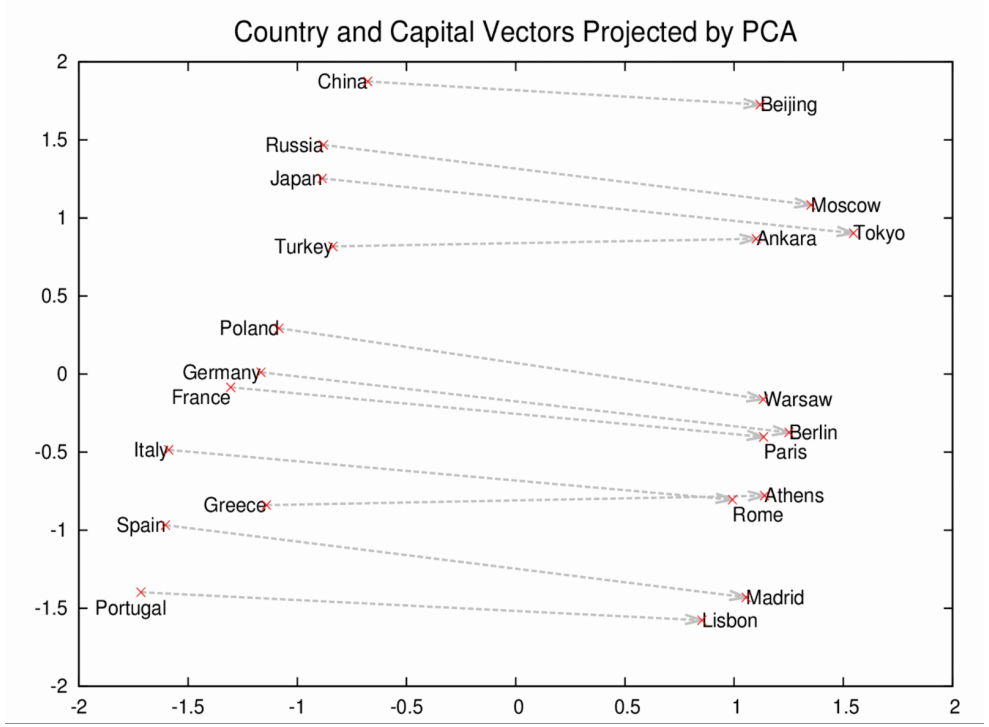
（各単語は他の単語との共起度合いで表されるベクトルとなっているため）

- cos類似度あるいはユークリッド距離
- DiceやJaccard係数のような集合演算（単語共起行列の要素が1/0であれば）
- 確率分布距離（KLダイバージェンス*非対称であることに注意、 JSダイバージェンス）
  - 単語共起行列の要素が頻度である場合、行の各要素をその行の要素の和で割ることにより確率の行列に変換できる

単語の分散表現（EMBEDDING）

word2vec（skip gramモデル）[Mikolov 13]のような単語の分散表現（Word Embedding）は単語共起行列を特異値分解により次元圧縮して得られたものと考えられる

- 具体的には共起行列の要素の値は単語同士の共起を相互情報情報量（PMI）で計算したものとする
  - より具体的にはShifted PMI [Levy 14]



単語ベクトル（単語の分散表現）を2次元に可視化したもの

国名のベクトルと首都のベクトルが対応が表れている

https://opensource.googleblog.com/2013/08/learning-meaning-behind-words.html
(ref. 20 Dec 2018)

テキスト分析と学習タスク

テキスト分類

- 文書-単語行列の各単語を特徴量（素性）として文書のラベルを予測する**教師あり学習**
  - 単語の重みが文書の特徴量値となる
  - フィルタリング、カテゴリ・トピック分類、書き手の推定、センチメント分析など

クラスタリング

- 文書-単語行列の各単語を特徴量（素性）として文書をクラスタリングする**教師なし学習**
  - 文書ベクトル間の類似度を元にクラスタリング
- 文書-単語行列の行列分解により文書あるいは単語のグループを抽出する次元削減の教師なし学習
  - LSAやLDA

	データ	統計	分析	ラベル
データを分析する	1	0	1	A
統計で分析する	0	1	1	B

参考書とツール

著作権の都合により
ここに挿入されていた画像を削除しました

書籍『統計的自然言語処理の基礎』表紙
Christopher D.Manning ・Hinrich Schutze 著・
加藤 恒昭・菊井 玄一郎・林 良彦・森 辰則訳
2017年11月
共立出版

https://www.kyoritsu-pub.co.jp/bookdetail/9784320124219

Pythonの自然言語処理モジュール

- NLTK
  - https://www.nltk.org/

Javaの自然言語処理ライブラリ

- Standford CoreNLP
  - https://stanfordnlp.github.io/CoreNLP/

