Text Mining with Machine Learning

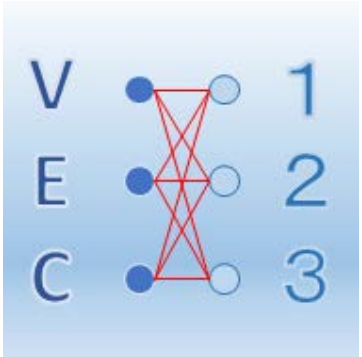
『Vector to』

powered by fastText

for Linux, macOS

リファレンスマニュアル ver. 1.6

竹岡　志朗



目次

はじめに

Vector to は Facebook 社の公開している機械学習、より具体的には分散表現(distributed representation)を用いたテキストマイニング・ソフトウェア fastText を基本に、fastText[1]の計算結果を応用することで、これまでとは異なるテキストマイニングを可能にすることを目的に構築されたアプリケーションソフトウェアです。対応する OS は Linux(Ubuntu 64bit)と macOS(High Sierra)です。Windows に関しては、Vector to 自体は対応できているのですが、分散表現の獲得に使用する fastText が対応しておらず、また開発者が独自にビルドした fastText やインターネット上で公開されている fastText には不具合[2]があり、今回は公開を見送りました。

現在はまだ開発の途上にあり、現在公開されている ver. 0.2.0 はβ版になります。今後、様々な機能を実装する予定ですが、発見できていないバグ等も多くあることが考えられます。ご使用に際しては、細心の注意を払って、また何らかの不具合が生じた際にはターミナル上に残ったエラーログ等を報告していただく等、正式版の開発に向けてご協力いただけると幸いです。

このリファレンスマニュアルでは Vector to の使い方について説明しますが、加えて後半で分散表現テキストマイニングの簡単な説明も行います。この中で KH Coder といったソフトウェアを用いた計量テキストマイニングと Vector to を用いた分散表現テキストマイニングの違いについても説明しています。分散表現テキストマイニングの分析事例に関する部分は竹岡(2018a)[3]、竹岡(2018b)、高木・竹岡(2018)[4]、竹岡(2018c)、竹岡(2018d)、竹岡(2019)を大部分そのまま使用しています。論文等で参考にされる場合にはこれをあげていただけると幸いです。

日進月歩の先端技術が基本となっているため開発者も十分に理解できていないところが

---

[1] fastText と同様に分散表現を用いた機械学習ソフトウェアとしては Word2vec などもありますが、fastText は他のソフトウェアと比較して正確性や速度の点で優れているとされています。また、プログラミング言語 python などを使用すれば、複数のモジュールがすでに用意されており、容易に使用することができます。また、ターミナル上から直接 fastText で分析することも可能です。

[2] 不具合としては、処理が重たい場合に強制終了する点、ターミナルからの「echo "word" | fasttext print-word-vectors model.bin」コマンドで bin ファイルから取得した単語ベクトルと vec ファイルに記載されている単語ベクトルが異なっている点があり、特に後者は重大な欠陥であることから、今回は公開を見送ることとしました。Windows 版は複雑なインストール作業がいらず、非常に簡単に使用を開始できることから、今後は、不具合が解消され次第公開したいと思います。

[3] https://stars.repo.nii.ac.jp/?action=pages_view_main&active_action=repository_view_main_item_detail&item_id=8915&item_no=1&page_id=13&block_id=67 でダウンロードいただけます。

[4] https://toyama.repo.nii.ac.jp/?action=repository_uri&item_id=16734&file_id=18&file_no=1 でダウンロードいただけます。

あります。もし間違っている点があればご指摘いただければ幸いです。

セットアップについて

Vector to は python で開発されており、Linux(Ubuntu[5])、macOS でご使用いただけますが、それぞれの OS によってセットアップの方法が異なりますので、ご使用の環境に合わせてセットアップをお願いします。

Windows しか手元にない方も Oracle 社の VirtualBox 等を使用すれば Windows 上に無料で Linux の仮想環境を簡単に構築することができます。インターネット上には画像付きでの解説も多く見つけられます。今後も継続して機械学習に関わる諸技術を利用していきたいと考えておられる場合には Linux でのご使用をお勧めします[6]。

Linux と macOS では python[7]関連ライブラリや MySQL をご自身でインストールしていただく必要がありますが、それについても下記で説明しています。

下記ではターミナル上で入力が必要な個所に関しては黄色のマーカーを入れています。これらを参考にしてください。なお、下記の方法でのインストールで動作を保証しているわけではありませんので、ご了承ください[8]。

Linux 版、macOS 版ともにソースコードからそのままご使用いただく形になっています。初めての方にはわかりにくい点もあるかとは思いますが、ご自身でソースコードを書き換えることで簡単に様々な変更を行えるという利点もあります。

---

[5] Ubuntu では動作確認済みですが、他の Linux ディストリビューションでは確認しておりません。

[6] 機械学習を行う環境として python が最も注目されていますが、Windows は python を使用する環境としてはあまりおすすめできません。最も不便だと感じるのが文字コードの違いと改行コードです。python や Ubuntu などの Linux を含む unix 環境では utf-8 という文字コードが使用されています。他方 Windows は cp932 という shift-jis が使用されています。どちらも広く普及していますが、Windows 上で収集した情報を python で使用する場合、初めから utf-8 で保存する(デフォルトは cp932)か、あるいは文字コードを変更する必要があります。他のプログラムとの連携を行う場合など、エラーが生じる可能性があるため、今後も機械学習などを使用したいと考えている場合には、Linux の環境を構築することをお勧めします。なお、Vector to では、shift-jis で作成したテキストファイルも直接読み込めるようにはしていますが、不具合がある場合には、ご自身で utf-8 に変換したものをご使用ください。

[7] python はインストールされています。ターミナルで python3 --version というコマンドを実行することで、現在インストールされている python3 のバージョンを確認することができます。

[8] Ubuntu はバージョンなどによって全く異なる動作をします。開発者がリファレンスマニュアルを作成した時点でも、動作しないことがありました。バージョンを変更する等で対応してみてください。

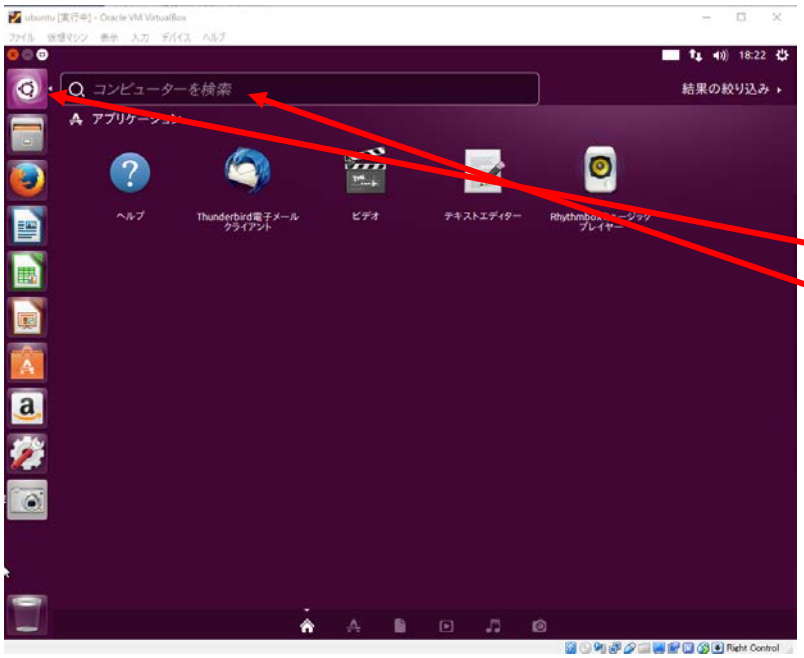
Linux(Ubuntu16.04 64bit)

Linux(Ubuntu)では OS をインストールするだけで python3[9]がインストールされています。現在 Ubuntu の最新バージョンは 18 です。ですが、バージョン 18 は安定していないところもあり、今回は比較的安定しているバージョン 16 での操作方法です。しかし、環境によっては Ubuntu16 で Vector to を使用すると IME が起動せず、日本語入力ができない場合があります(コピー&ペーストでの入力は可能です)。その場合は 17、18 をインストールしてみてください。以下では、17、18 をインストールする際の注意点も記入しています。

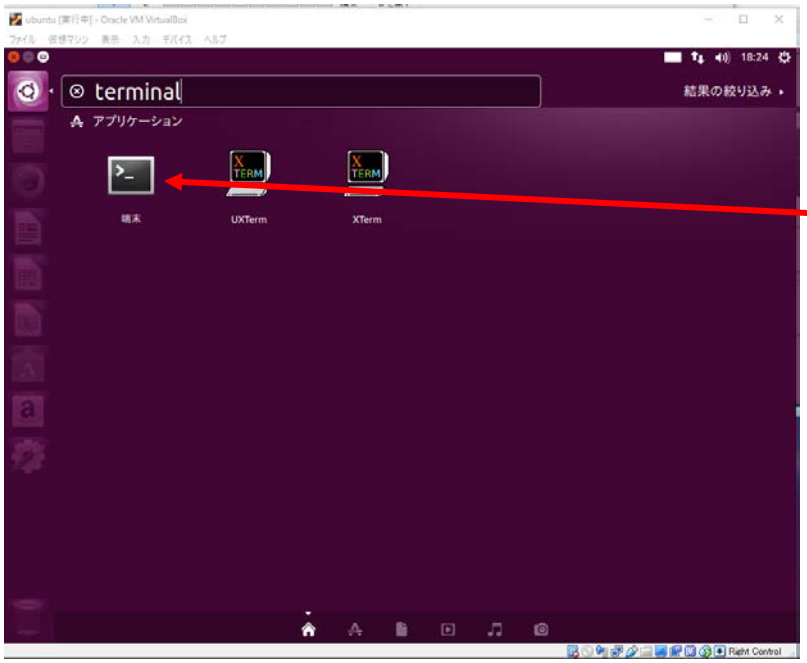
Vector to のご使用に際しては MySQL、Mecab と辞書、python のモジュールのインストールも必要です。

インストールはターミナルにコマンドを入力することで実行します。

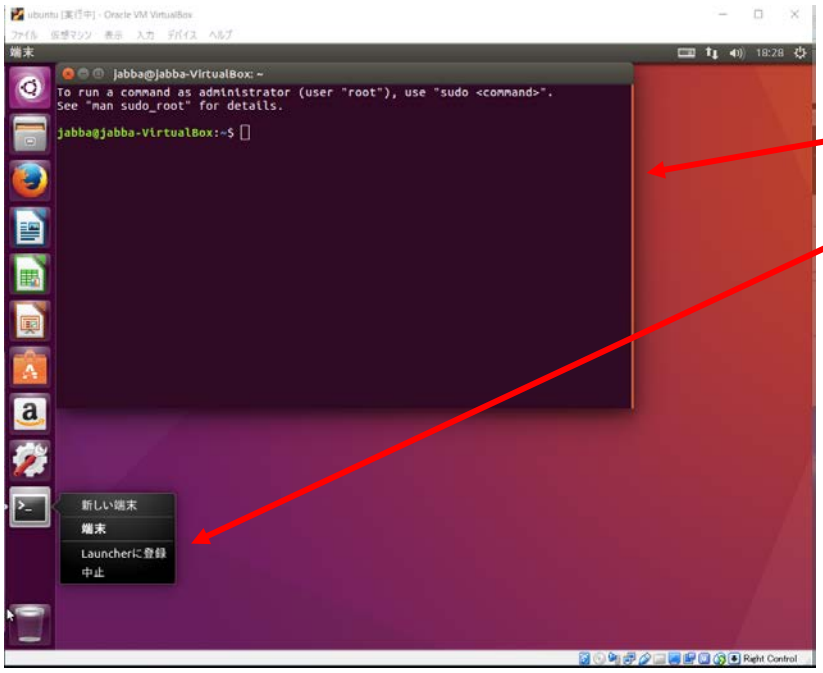
ターミナルの起動の方法



[9] Vector to は python2 ではご使用いただけません。python3 をご使用ください。



3. 端末をクリック



4. ターミナルが起動
5. 右クリックし Launcher に登録しておくと今後はここをクリックするだけで端末を起動できます

各ソフトウェアとモジュールのインストール[10]

ダウンロードした Vector to を任意のフォルダーに解凍してください。

インストールを進めると頻繁に Yes/No 型の質問やパスワードの入力が求められますの

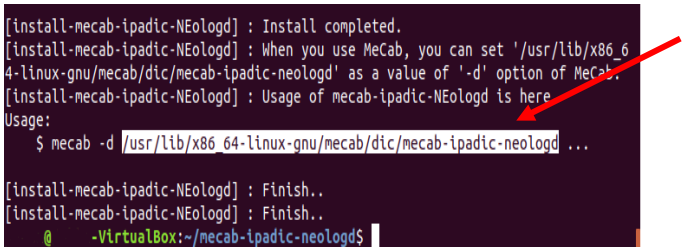
[10] 初めて Ubuntu をインストールされた方は sudo apt update と sudo apt upgrade を先に行ってください。sudo は unix 系のコマンドで、ルート権限で実行する場合に使用します。

で、随時入力してください。

ターミナルを起動

sudo apt-get install mysql-server　　#MySQL のダウンロードとインストール
インストール中にパスワードの設定を求められますので、各自で設定してください。
それが root のパスワードになります。

sudo apt install git　　#git のインストール
sudo apt-get install build-essential g++　　#c++コンパイラのインストール（既にインストール済みの場合もありますが、上書きしても問題はありません）
sudo apt-get install curl　　#web 言語のインストール
sudo apt-get install mecab libmecab-dev mecab-ipadic mecab-ipadic-utf8　　#Mecab のダウンロードとインストール
git clone --depth 1 https://github.com/neologd/mecab-ipadic-neologd.git　　#Mecab 辞書のダウンロード
cd mecab-ipadic-neologd　　#Mecab の辞書フォルダーに移動
./bin/install-mecab-ipadic-neologd -n -a　　#辞書ファイル [11]のインストール。インストール中に Yes or No を尋ねられますので Yes と入力してください。

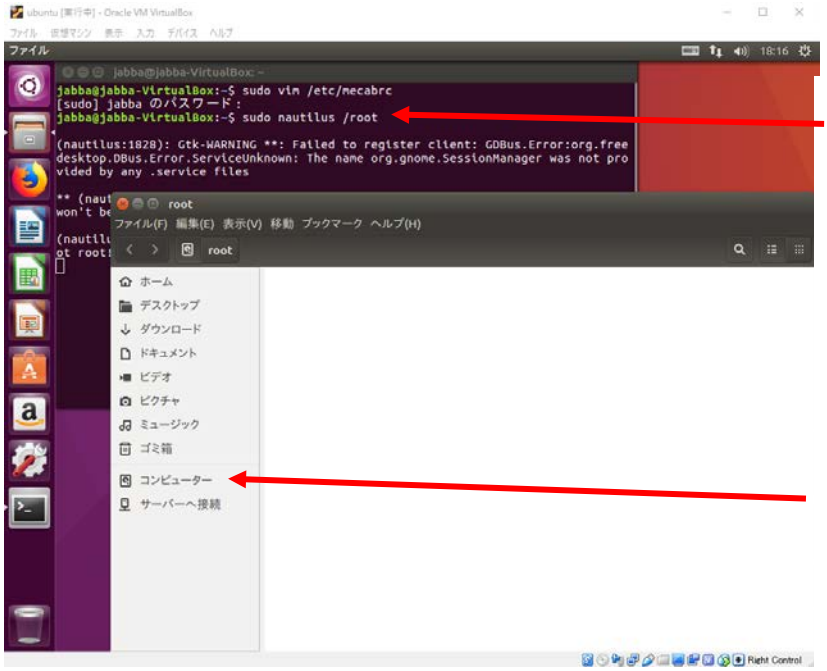


インストール終了後にターミナル上に辞書ファイルのインストール先が表示されます。次のデフォルト辞書ファイルの変更では、ここで表示されているアドレスを入力してください。Ubuntu のバージョンによって異なるようです。

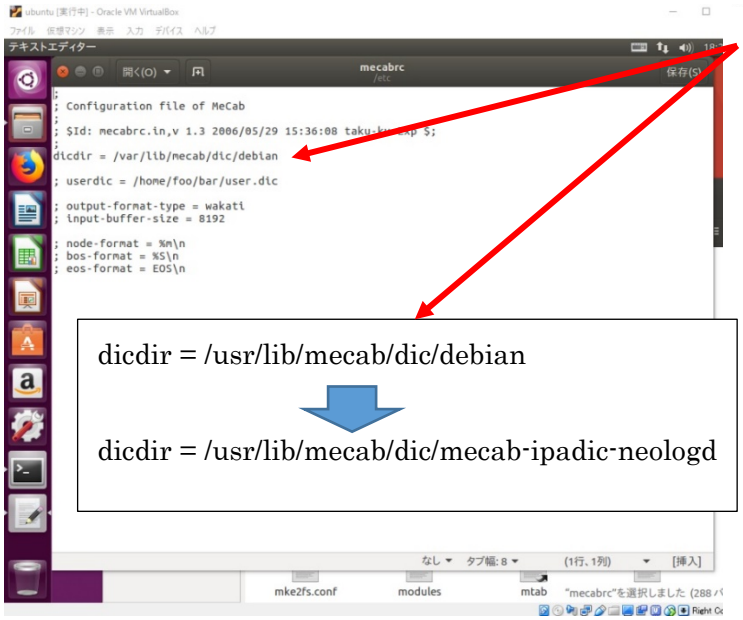
次に Mecab のデフォルトの辞書ファイルを変更します。手順が複雑ですので画像付きで

11 辞書ファイルは頻繁に更新されていますので、研究などで使用される場合は、辞書のバージョンを記載することをお勧めします。辞書のバージョンは更新日で代用するのが一般的なようです。

確認します[12]。



1. ターミナルに sudo nautilus /root と入力。
   #root 権限でフォルダーやファイルを編集するモードに入る
2. コンピューター→ etc フォルダー→ mecabrc ファイルを開く



3. アドレスを修正。このアドレスは Ubuntu16.04 環境下でのものです。
   Ubuntu17.10 環境下では異なるアドレスにインストールされていましたので、先ほど確認した辞書ファイルのインストール先を記入してください。
4. テキストエディターを閉じる
5. フォルダーを閉じる
6. ターミナルに戻り ctrl+z で root 権限での編集モードを終了
7. ターミナルを閉じる

[12] Ubuntu17.10 では、一度ログアウトし、ログイン時の設定を Ubuntu on Xorg に変更の上再ログインしてください。そのうえで上記の方法で修正してください。
Ubuntu on Xorg の設定に関する詳細は
https://www.youtube.com/watch?v=_YKCW8lAkd8 をご覧ください。

python ライブラリのインストール

次に python が使用するライブラリをインストールします。複数あるので順番に入れていきます。下記コマンドの実行に際して「You are using pip version 8.1.1, however version 9.0.3 is available. You should consider upgrading via the 'pip install --upgrade pip' command.」のようなエラーメッセージがターミナル上に出る場合がありますが、インストールには問題ありませんので、続けてください。

新しくターミナルを開く

```
sudo apt install python3-pip                #ライブラリを簡単にインストールするためのソフトウェアをインストールする
pip3 install PyQt5                          #GUI ライブラリ
sudo apt-get install libmysqlclient-dev     #MySQL ライブラリ
pip3 install mysqlclient                    #上記に同じ
pip3 install numpy                          #ベクトル行列の計算
pip3 install scipy                          #上記に同じ
pip3 install matplotlib                     #グラフ描画ライブラリ
sudo apt-get install python3-tk             #GUI ライブラリ
pip3 install pandas                         #データ解析
pip3 install scikit-learn                   #機械学習ライブラリ
pip3 install cython                         #python 拡張ライブラリ
pip3 install pyfasttext                     #学習済みファイルを開くライブラリ
```

pyfasttext のインストールに失敗する場合は「pip3 install --upgrade setuptools」を先に実行してみてください。

fastText 本体(v0.1.0)のコンパイル

```
wget https://github.com/facebookresearch/fastText/archive/v0.1.0.zip
                                            #fastText のソースコードのダウンロード
unzip v0.1.0.zip                            #ZIP ファイルの解凍
cd fastText-0.1.0                           #カレントディレクトリの移動
make                                        #コンパイル
```

作成された「fastText-0.1.0(home フォルダーにあると思います)」フォルダーの名前を「fastText(2 つ目の「T」は大文字です。)」に変更し、Vector_to フォルダー内に移動させてください。

MySQL でのユーザーの作成と権限の変更

MySQL は、ID に「vector」、パスワードに「to」でユーザーを作成し、すべての権限を付与してください。

ターミナルを起動

```
mysql -u root -p                                    #MySQL にルート権限でログイン
```

MySQLl にログイン後(下記コマンドはセミコロンまで入力してください)

```
create user vector@localhost identified by "to";   #ユーザーの作成
grant all privileges on *.* to vector@localhost;    #権限の変更

exit                                                #MySQL からのログアウト
```

以上で Vector to の使用の準備は整いました。

macOS(High Sierra)

以下では macOS(High Sierra)のインストール手順を記しています。macOS は Linux と同じ UNIX 系の OS のため、基本的には Linux 版と同じ手順になります。まず、ダウンロードした Vector to を任意のフォルダーに解凍してください。

インストールを進めると頻繁に Yes/No 型の質問やパスワードの入力が求められますので、随時入力してください。

ターミナルの起動の方法

#Launchpad を起動

#「その他」をクリック

#「ターミナルをクリック」

各ソフトウェアとモジュールのインストール

ソフトウェアのインストールは Homebrew を使用して行いますので、インストールされていない方は、こちらを先にインストールしてください。ブラウザーで https://brew.sh/index_ja にアクセスしてください。

1. この部分をコピーし、ターミナルに張り付けて実行してください。

続いて、python をインストール行いますが、その前に python のインストールの有無と、そのバージョンを確認します。

ターミナルを起動
python3 -V

このコマンドで「python 3.×.×」が出た場合にはインストールが済んでいますので、モジュールのインストールに進んでください。

python のインストール

python3 がインストールされていない場合は、下記を実行します。

ターミナルを起動
brew install python3　　#python のインストール

python ライブラリのインストール

次に python が使用するライブラリをインストールします。複数あるので順番に入れていきます。ターミナルで下記コマンドを実行してください。

pip3 install PyQt5　　#GUI ライブラリ
pip3 install numpy　　#ベクトル行列の計算
pip3 install scipy　　#上記に同じ
pip3 install matplotlib　　#グラフ描画ライブラリ

```
pip3 install pandas                    #データ解析
pip3 install scikit-learn              #機械学習ライブラリ
pip3 install chardet                   #文字コード判定ライブラリ
pip3 install pillow                    #画像ファイル表示
pip3 install cython                    #python 拡張ライブラリ
pip3 install pyfasttext                #学習済みファイルを開くライブラリ

pip3 install mysqlclient               #MySQL 関連ライブラリ
brew install tcl-tk                    #GUI ライブラリ
```

MySQL のインストール

```
brew install mysql                     #MySQL のインストール
mysql.server start                     #MySQL の起動
mysql_secure_installation              #MySQL の設定の開始
mysql.server stop                      #MySQL の終了
```

MySQL の設定の変更

my.cnf という MySQL の設定ファイルの一部を書き換えます。

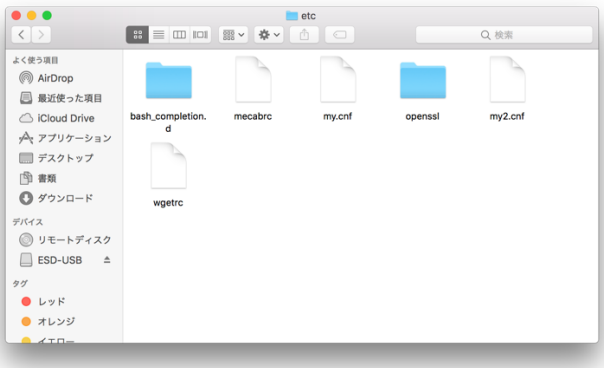
Finder でコンピュータに移動

command + shift + .(ドット)を同時押し　　#隠しファイルの表示

/usr/local/etc/に移動　　#フォルダーを移動

my.cnf ファイルをコピーして名前を変えて残しておく

#下記スクリーンショットでは「my2.cnf」というファイル名に変更して元になるファイルを残しています。



my.cnf ファイルを開く　　#「テキストエディット」等で開い

てください。

「bind-addres = 127.0.0.1」の下に

character-set-server=utf8mb4

validate-password=OFF

socket=/tmp/mysql.sock

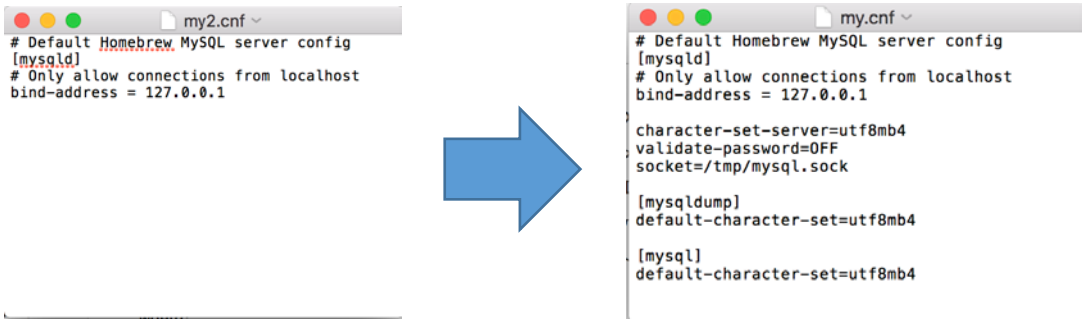
[mysqldump]

default-character-set=utf8mb4

[mysql]

default-character-set=utf8mb4

を加えてください。



入力を終えましたら、保存して終了してください。

Mecab のインストール

ターミナルを開く

brew install mecab　　#Mecab のインストール

cd ~/Downloads　　#downloard フォルダーに移動

git clone --depth 1 https://github.com/neologd/mecab-ipadic-neologd.git

#辞書のダウンロード

cd mecab-ipadic-neologd　　#フォルダーの移動

./bin/install-mecab-ipadic-neologd -n　　#辞書のインストール

ターミナルを閉じる

Mecab の設定の変更

command + shift + .(ドット)　　#隠しファイルの表示

/usr/local/etc/に移動

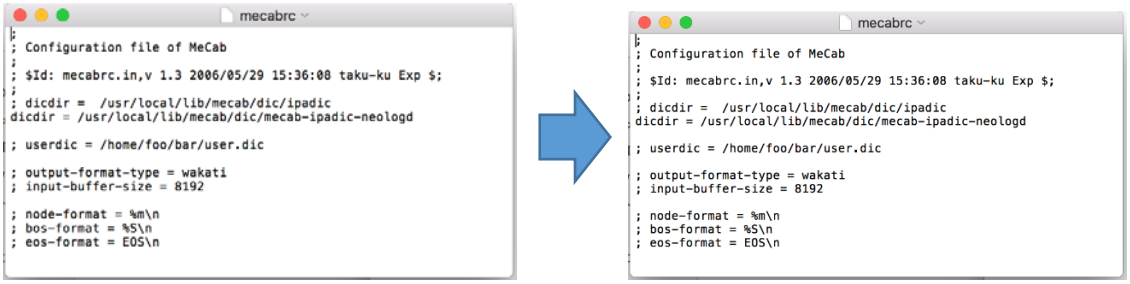
Mecabrc ファイルを開く　　#「テキストエディット」等で開いてください。

ファイル内の

dicdir = /usr/local/lib/mecab/dic/ipadic

をコメントアウト（先頭に「；」を打つ）し、下の行に

dicdir = /usr/local/lib/mecab/dic/mecab-ipadic-neologd



を入力してください。

入力を終えましたら、保存して終了してください。

これで Mecab で使用する辞書ファイルが変更されます。ターミナルを閉じて終了してください。

fastText のインストール

cd vector_to　　#カレントディレクトリの変更（実際にはもう少し長いアドレスになります。分からない場合は、Vector to のフォルダーを 2 本指クリックし、コンテキストメニューが出たところで「option」キーを押し、「Vector to のパス名をコピー」を選んでください。ターミナル上で「cd （cd の後にスペースを忘れずに)」と入力したのちに、貼り付けを行い、実行するとカレントディレクトリを変更できます。)

brew install wget　　#ダウンローダーのインストール

wget https://github.com/facebookresearch/fastText/archive/v0.1.0.zip

#fastText のダウンロード

unzip v0.1.0.zip　　#fastText の解凍

cd fastText-0.1.0　　#解凍したフォルダーに移動

make　　#fastText のビルド

Vector to フォルダー内に「fastText-0.1.0」フォルダーができていると思います。最後にフォルダー名を「fastText-0.1.0」から「「fastText(2 つ目の「T」は大文字です。)」」

に変更してください。

MySQL でのユーザーの作成

MySQL は ID に「vector」、パスワードに「to」でユーザーを作成し、すべての権限を付与してください。

ターミナルを起動

```
mysql.server start                    #MySQL の起動
mysql -u root -p                      #MySQL にルート権限でログイン
```

MySQLl にログイン後(下記コマンドはセミコロンまで入力してください)

```
create user vector@localhost identified by "to";   #ユーザーの作成
grant all privileges on *.* to vector@localhost;   #権限の変更

exit                                  #MySQL からのログアウト
mysql.server stop                     #MySQL の停止
```

以上 Vector to 使用の準備が終わりました。

Vector to を使った分析のチュートリアル

「習うより慣れよ」で、実際に分析を行い、Vector to の基本的な使い方を説明します。

練習に使用するのは宮沢賢治の「風の又三郎」、夏目漱石の「こころ」、太宰治の「走れメロス」と「列車」です[13]。この中で「風の又三郎」「こころ」「走れメロス」を教師データとし、「列車」の各段落を「風の又三郎」「こころ」「走れメロス」に分類したいと思います。小説などに対してテキストマイニングを用いる研究の歴史は古く、シェイクスピアがフランシスコ・ベーコンではないかという説を否定するために用いられたりもしています(西内, 2013)。この際のテキストマイニングは計量テキスト分析であり、筆者の予測が目的となっていましたが、Vector to で行う分散表現テキストマイニングは文章の内容により分類を行うものであり、これまでの分析とは異なります。

先に結論から書きますと、この分析には何も意味はありません。本来であればクチコミデータなどを分析することで理解を深めていただきたいのですが、著作権の問題があり 1 次データを配布することができず、意味のあるデータを用いての練習はできません。その点をご了承ください。また、このチュートリアルで説明できなかった事柄に関しましてはチュートリアルの後に説明します。

分析に入る前に、分析対象となるファイルの作成について触れておきます。Vector to での分析の基本は段落単位になります。文章の分類を行う場合には、教師データは段落単位でラベルを張る必要があり、また分類を行う対象も段落単位になります。分析に使用するファイルを作成する場合にはこの点に注意してください。例えばクチコミを分析する場合、1 件のクチコミが複数段落にわたる場合がありますが、分析の目的によっては前もって複数段落にわたる 1 件のクチコミを 1 段落に変換する処理を行うほうが良いでしょう。

分析ファイル(教師データ)の登録

vector to フォルダー内の tutrial フォルダーに kazeno.txt、kokoro.txt、merosu.txt、ressya.txt というテキストファイルがあります。このなかで kazeno.txt、kokoro.txt、merosu.txt を教師データとして登録します。まず Vector to を起動してください。

Vector to の起動

起動の方法は OS によって異なります。

Linux

ターミナルを起動し vector_to フォルダーに移動後、python3 vector_to.py と入力してください。

---

[13] これらの小説はすべて「青空文庫（http://www.aozora.gr.jp/）」にアップロードされているものを分析可能な形に変換の上使用しています。

例：personal フォルダーに Vector to を解凍した場合

```
cd /home/personal/vector_to                    #Vector_to フォルダーに移動
python3 vector_to.py                           #Vector to の起動
```

macOS

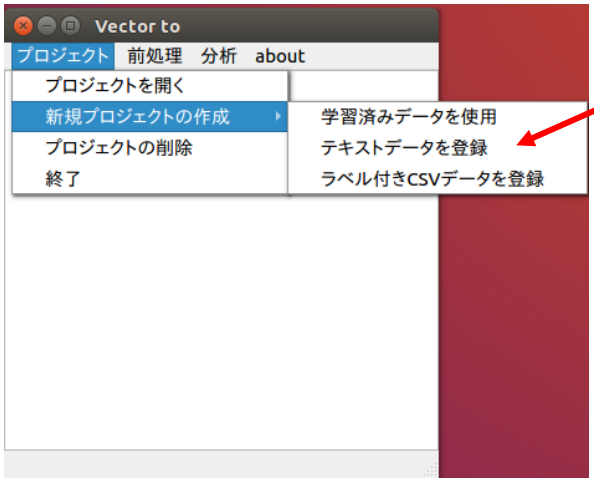
ターミナルを起動し、Vector to フォルダーに移動後、python3 vector_to.py と入力してください。

例：Documents フォルダーに Vector to を解凍した場合

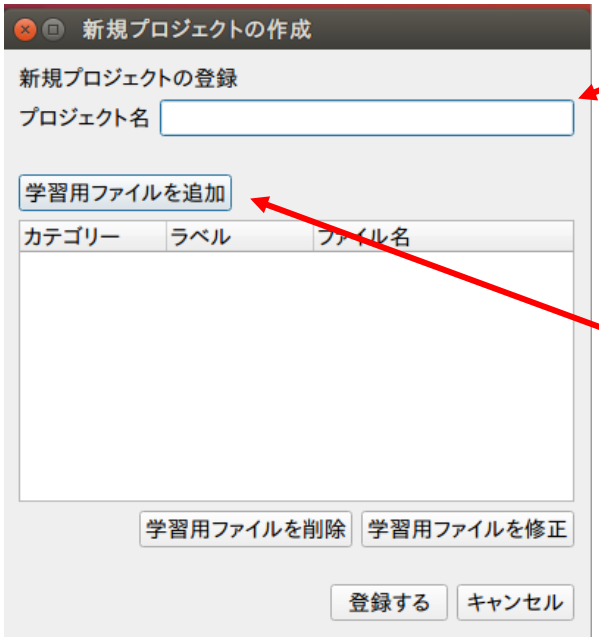
```
cd /Users/〇〇〇〇 /Documents/vector_to    # Vector_to フォルダーに移動
python3 vector_to.py     # Vector to の起動
```

プロジェクトの登録

Vector to では教師データなどの分散表現の取得に用いるファイルをプロジェクトとして登録する必要があります。今回は複数のテキストファイルを登録しますので、メニューバーの「プロジェクト」から「新規プロジェクトの作成」、「テキストデータを登録」と進んでください。

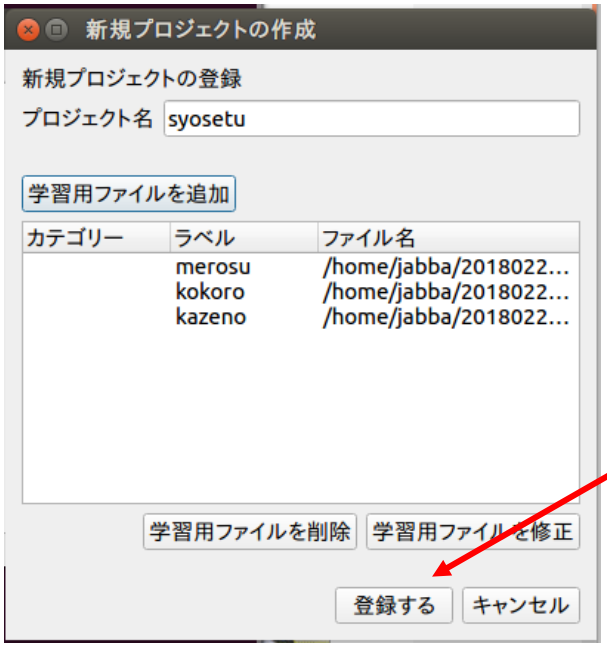


1. 「テキストデータを登録」を選択



2. プロジェクト名を入力してください。プロジェクト名は半角英数字で登録してください。日本語を用いますとバックグラウンドでエラーが発生します。
   ここでは「syosetu」と入力します。
3. 「学習用ファイルを追加」をクリック。

4. 「ファイルの参照」をクリックし、テキストファイルを登録します。テキストファイルは 1 つずつ登録してください。まず、kazeno.txt を選択します。
5. ファイル名をラベル名として使用にチェックを入れてください。ラベル名をご自身で登録したい場合はご自身で入力してください。ラベル名には学習時の分類名を入力します。半角英数字をご使用ください

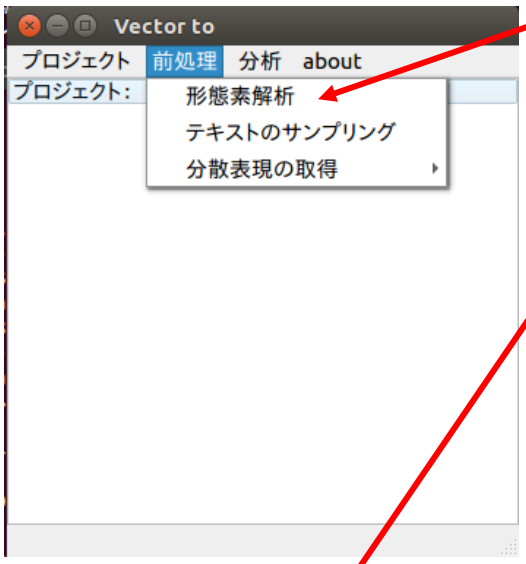


6. 「プロジェクトに追加」をクリック
7. 同じように kokoro.txt、merosu.txt も登録してください。必ずラベル名 kokoro、merosu と入力しておいてください。
8. 「登録する」をクリックします

以上でプロジェクトと、分散表現の取得に使用するファイルの登録は終了です。続いて、前処理に入ります。ここでは形態素解析、テキストのサンプリング、分散表現の取得を行います。

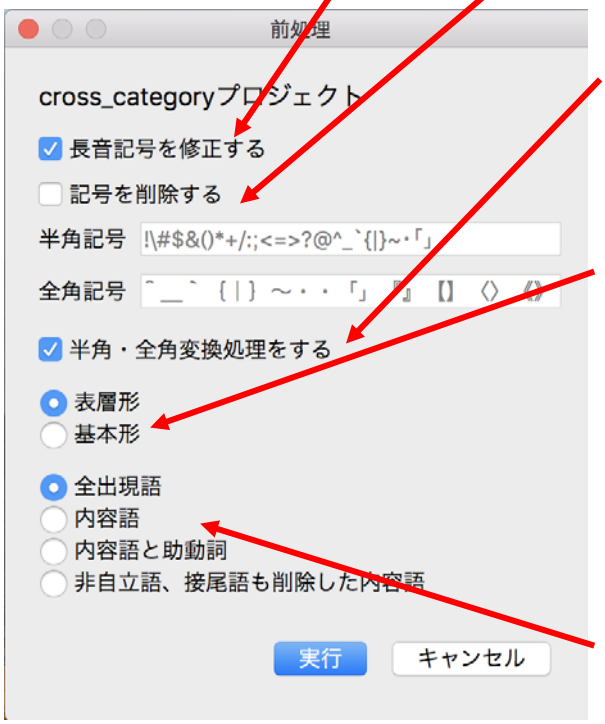
形態素解析と分散表現の取得

続いて、形態素解析と分散表現の取得を行います。これらはテキストマイニングのための前処理に当たります。

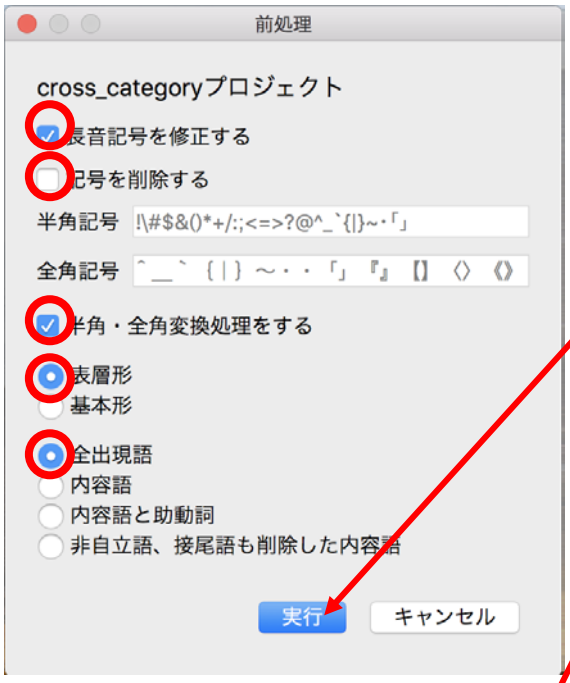


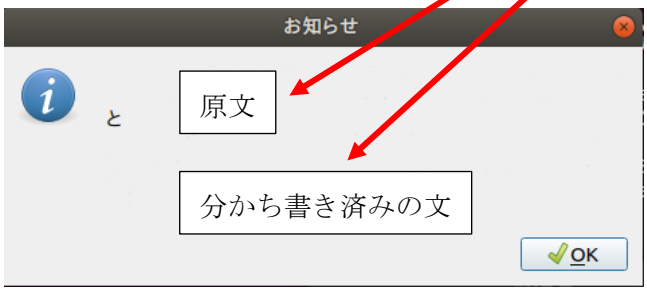
1. メニューバーの「前処理」から「形態素解析」を選択してください

形態素解析を行う前に、分析対象となる文章に対して前処理を行うことができます。

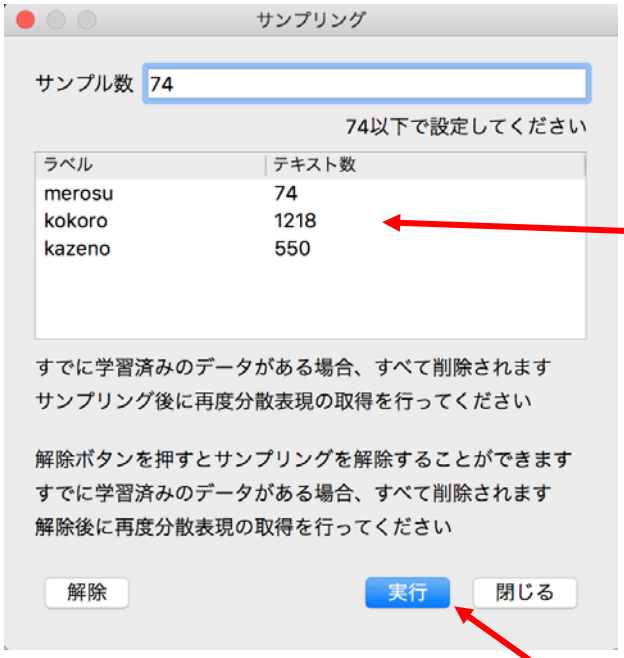


- 長音記号を修正する
  - インターネット上のクチコミなどでは長音記号に「ー」「—」「‐」など様々な記号が使われています。この欄にチェックを入れるとまとめて修正できます。
- 記号を削除する
  - 文章中では様々な記号が使用されていますが、これらを削除することができます。
- 半角・全角変換処理をする
  - カタカナ文字を全角に、アルファベットとアラビア数字を半角に変換します
- 表層形・基本形
  - たとえば、文章中には「見」、「見る」、「見れ」という単語が登場します。これらを基本形の「見る」に変換して処理するのか、それとも「見」、「見る」、「見れ」として処理するのかを選択します。表層形は文中の表現を使用します
- どういった単語を学習に利用するかを選択できます。内容語は名詞、動詞、形容詞といったそれだけで意味を持つ単語です。助動詞は否定表現などです。内容語の存在しない段落が存在する場合、エラーが発生しますので、その場合は、全出現語にするか、そういった段落を削除してください。





2. 今回は長音記号の修正、記号の削除、半角・全角処理にチェックを入れ、表層形、全出現語で形態素解析を行いますので、左図のようにチェックを入れてください。
3. 「実行」をクリックすると形態素解析が始まります。この処理は文章が長いほど時間がかかります。終了すると、左図のウィンドウが自動的に閉じられメイン画面に戻りますので、メイン画面に戻るまでお待ちください。
4. Ubuntu17 では「"vector_to の応答がありません"」というメッセージが出ることがありますが、「応答を待つ」をクリックしてください。以下同じです。
5. 形態素解析が終わるとダイアログボックスがあらわれます。この中で表示される原文と分かち書き済みの文が同じものであることを確認してください。これが異なる場合には登録したテキストファイルに問題があります。テキストファイルを修正して、再度、形態素解析を行ってください。また、終了時にターミナル上にエラーが表示される場合があります。この場合にもテキストファイルを修正して、再度形態素解析を行ってください。記号だけの段落や内容語の存在しない段落がある場合にエラーが発生します。

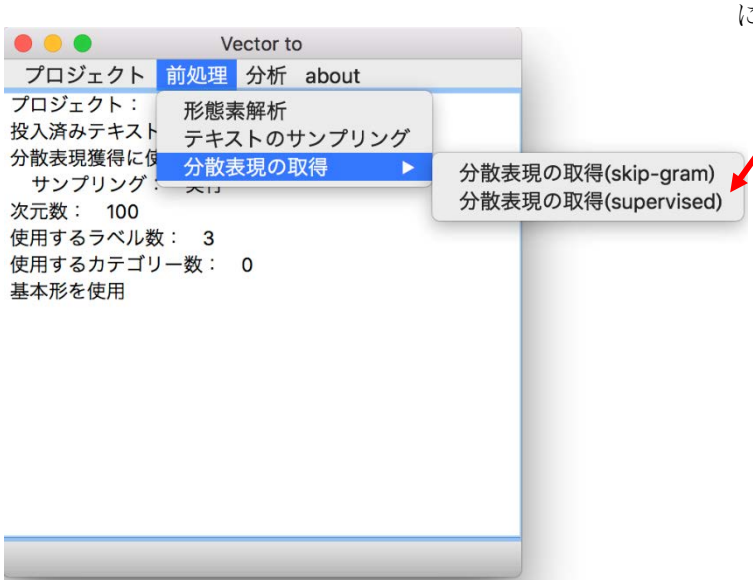


6. 続いてテキストのサンプリングを行います。メニューバーの「前処理」から「テキストのサンプリング」を選択してください

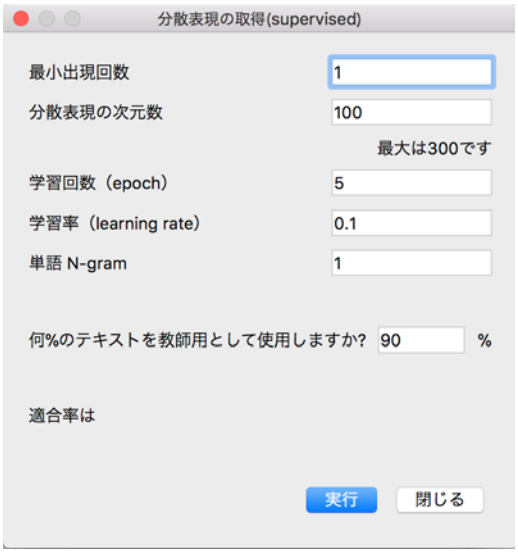
7. サンプリング画面が開くと merosu ラベルのファイルにはテキストが 74、kokoro には 1218、kazeno には 553 段落あることが表示されています。サンプリング機能はすべてのラベルが同数になるように調整を行います。今回の場合は、段落数の最も少ない merosu に合わせて 74 以下になるように設定できます。

8. 今回はサンプル数を 74 にして、「実行」をクリックしてください。

9. 「実行」をクリックするとサンプリングがバックグラウンドで行われ、終了するとメイン画面に戻ります。



10. 続いて分散表現を取得します。今回は「分散表現の取得（supervised）」を選択してください。ラベル名が登録されている場合には、こちらでパラメーターを調整することで学習の精度を高めることができます。



パラメーターとハイパーパラメーター

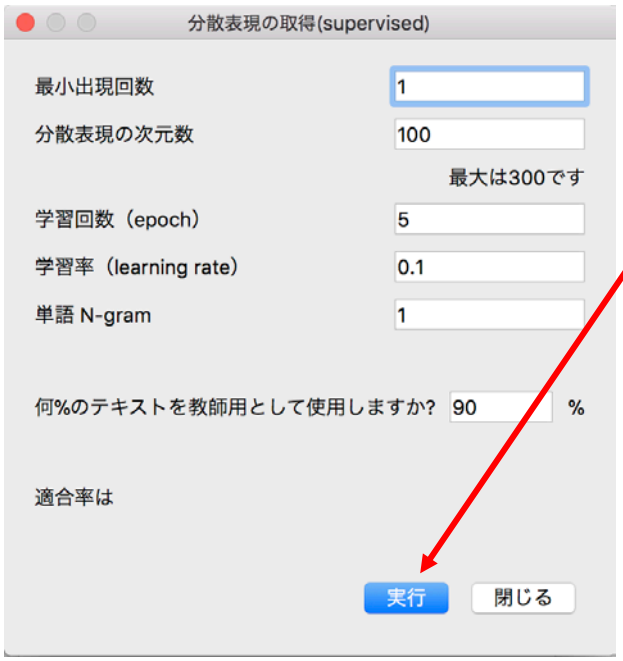
パラメーターはニューラルネットワークを用いた分析では重みやバイアスといった自動的に調整されるものを指すことが多く、利用者が設定可能なものはハイパーパラメーターと呼ばれます。

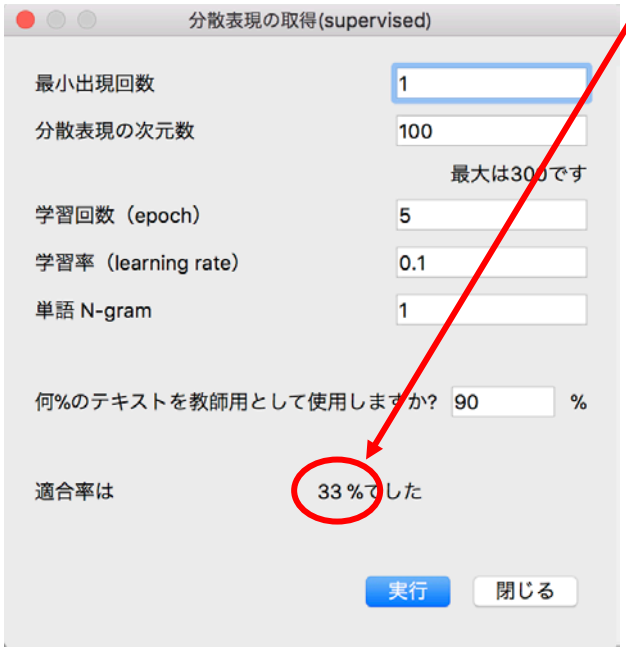
単語 N-gram について

例えば「新しい 本 を 注文 した」という文章は通常であれば「新しい」「本」「を」「注文」「した」というように単語ごとに分散表現を算出しますが、単語 N-gram を 2 に設定すると「新しい 本」「本 を」「を 注文」「注文 した」というように 2 語を 1 トークンとして分散表現を算出します。

11. ここでハイパーパラメーターを調整し学習の精度を高めます。初期値は fastText のデフォルト値になっていますが、このままでは不十分な成果しか得られません。下記ハイパーパラメーターを調整することで適合率を上げてください。

- 最小出現回数
  - 文章全体で何回以上出現した単語を分散表現の計算の対象にするかを設定します。
- 分散表現の次元数
  - 何次元のベクトル表現にするかを設定します。
- 学習回数
  - 全文章を何回学習するかを設定します。[推奨値：5-50]
- 学習率
  - 学習ごとにモデルをどの程度更新するのかの設定。大きいほど最適解に早く近づくことができるが、最適解を飛ばしてしまう可能性もある。[推奨値：0.1-1]
- 単語 N-gram
  - 分散表現を取得するにあたって、左右何単語を考慮に入れるかを設定します。語順が重要な役割を果たすような分類作業で活用します。[推奨値：1-5]
- 教師データと検証用データの割合
  - 教師データとして何%のデータを使用するかを設定します。
- 適合率
  - 全データから教師データを除いたデータを用いて、学習成果の適合率を判定します。





12. まず、デフォルト設定で文章分類を行い、その適合率を確認します。
13. 「実行」をクリックしてください。この処理は時間がかかります。終了するとダイアログボックスが出現し、検証画面に適合率が表示されますので、お待ちください。

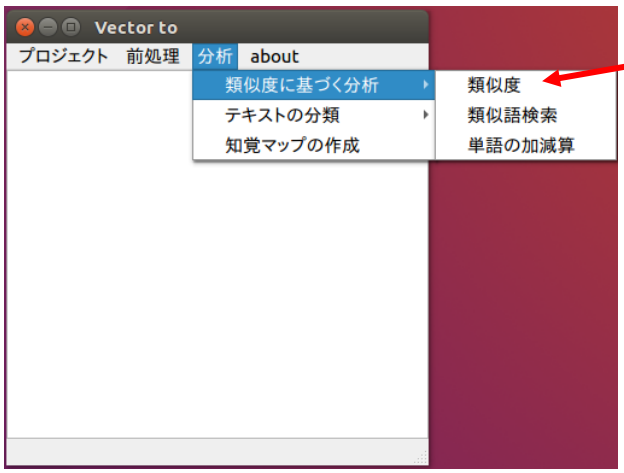
14. 開発者の環境ではデフォルト設定での適合率は 33%でした。fastText では学習の初期化に際してランダム処理を行っています。そのため、学習の試行ごとに値は変わります。
15. 設定を変更することで精度を高めます。
    - 学習回数→50
    - 学習率→1

    に変更し、実行をクリックします。開発者の環境では 58%まで上がりました。
    適合率はハイパーパラメーターを調整することで上げることができます。しかし、過学習の可能性も考慮して調整してください。単純に 100%まで上がればよいというわけではありません。
16. これ以上の向上は望めそうにありませんので、これを学習成果として「閉じる」をクリックして学習を終了します。

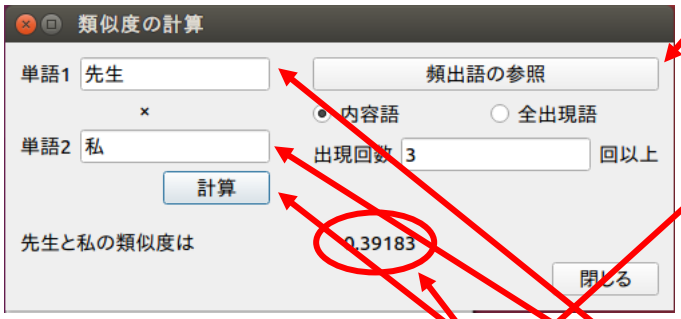
以上で前処理と学習は終了です。

データの分析

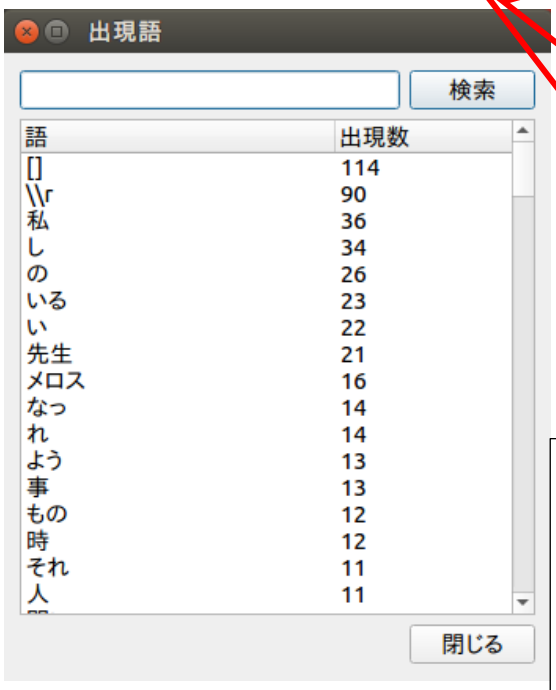
類似度



1. 単語の類似度を計算します。メニューバーの分析から「類似度」を選択します。



2. 「頻出後の参照」をクリックすると内容語(全出現語も可)の中で出現回数の多い語を確認することができます(出現回数で絞ることもできます。学習済み bin ファイルを用いている場合は頻出語の参照機能は使用できません)。

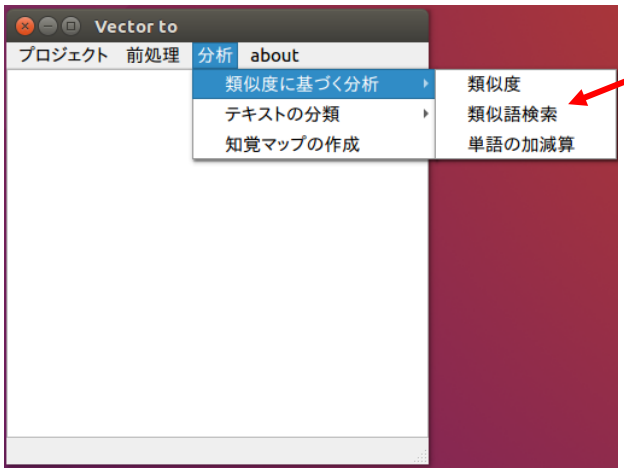


3. 「先生」と「私」の類似度を調べてみます。単語 1 に「先生」を、単語 2 に「私」を入力し「計算」をクリックします。

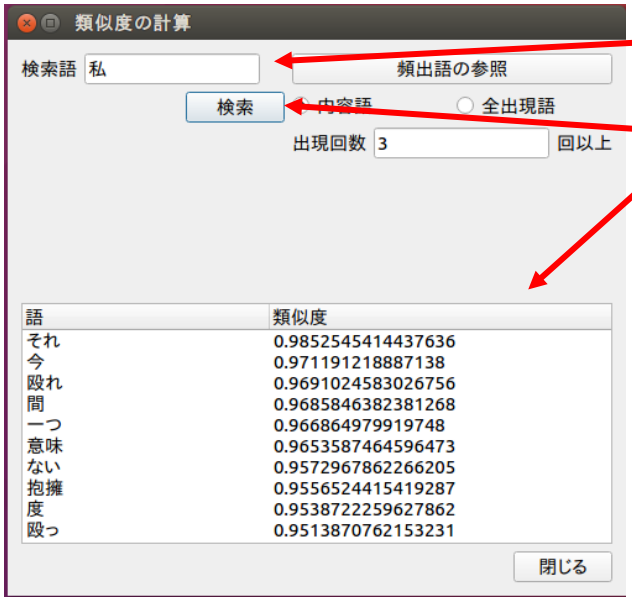
4. 類似度の計算結果が表示されます。「先生」と「私」の類似度は 0.39183 でした。

チュートリアルはデータ数が少ないため、分析結果にかなりばらつきが生じます。チュートリアルにある単語や数値情報とご自身の分析結果に違いがあっても、ターミナルにエラー表示などがない場合、ソフトウェアのエラーではなく、正常な分析を行えています。

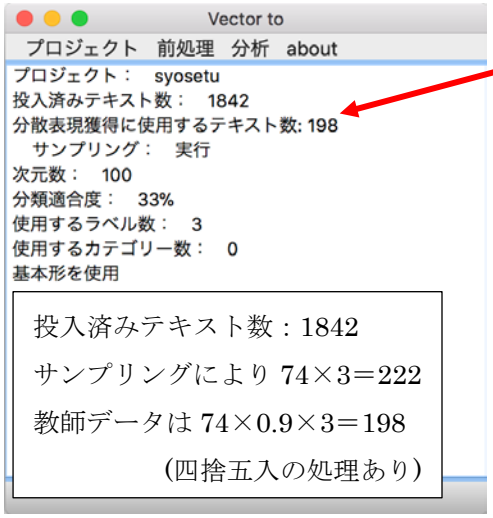
類似語検索



1. 単語を指定して、それと類似する単語を検索します。メニューバーの分析から「類似語検索」を選択します。

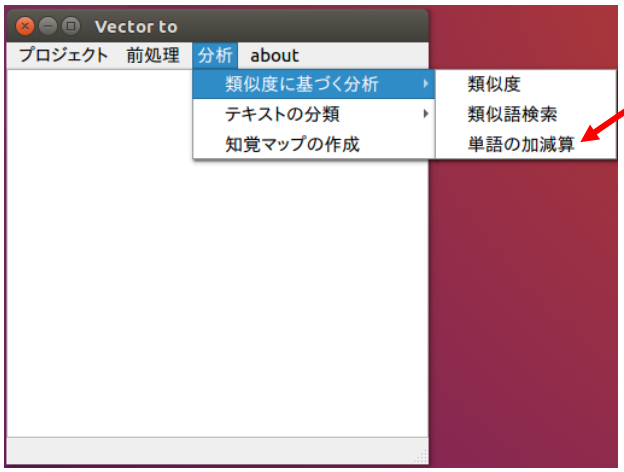


2. 検索語に「私」と入力します。
3. 「検索」をクリックすると下部に「私」と類似度の高い語の一覧が表示されます。今回の場合、関係のないと思われる語が出ていますが、これは学習に利用した文章数が少ない（サンプリングと精度分析で検証データとして一部データを利用しており、実際に学習に利用しているデータが 198 件と少ない）ためです。1 万件以上のデータがあれば分析精度は高まります。

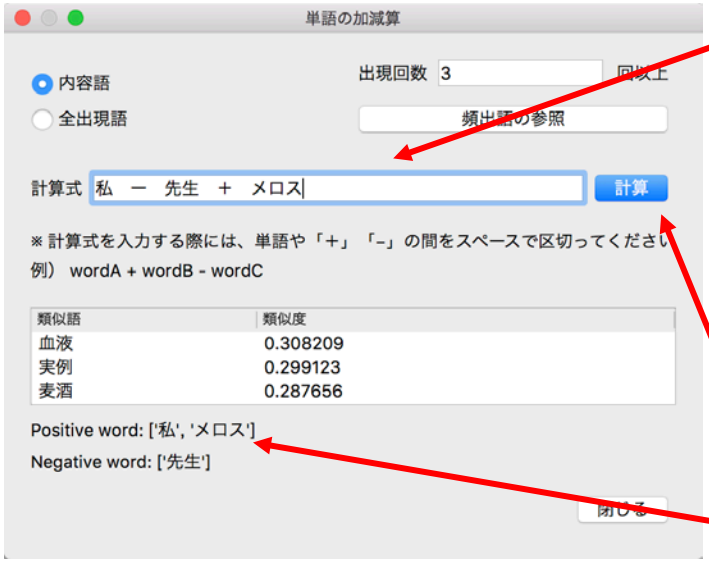


投入済みテキスト数：1842
サンプリングにより 74×3＝222
教師データは 74×0.9×3＝198
（四捨五入の処理あり）

単語の加減算



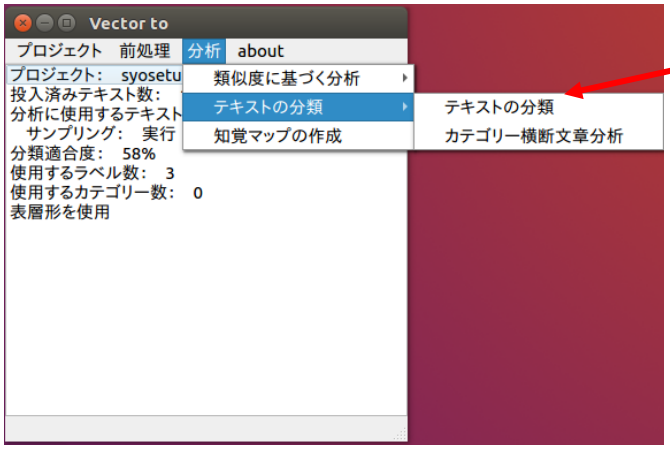
1. 加法構成性を利用して、単語の加減算を行います。メニューバーの分析から「単語の加減算」を選択します。



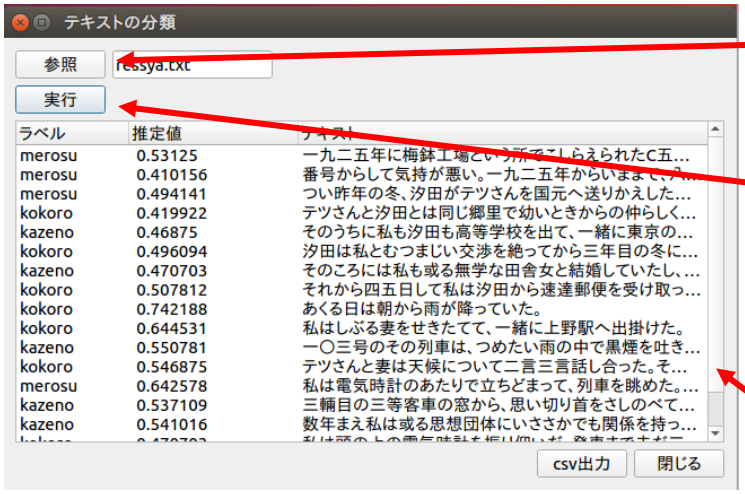
2. 計算式欄に「私 - 先生＋メロス」と入力します。計算式は単語をプラス記号やマイナス記号でつないで記入してください。単語とプラス記号やマイナス記号の間はスペースで区切ってください。
3. 「計算」をクリックしてください。
4. Positive は計算式の中で「＋」記号のもの、Negative は「－」記号のものです。
5. 下部に類似語と類似度が表示されます。今回は教師データが少ないためこのような結果になりましたが、十分なデータ数があれば、「王様－男＋女＝女王」のような計算も可能です。

テキストの分類

次に太宰治の「列車」の各段落を「風の又三郎」「こころ」「走れメロス」に分類します。



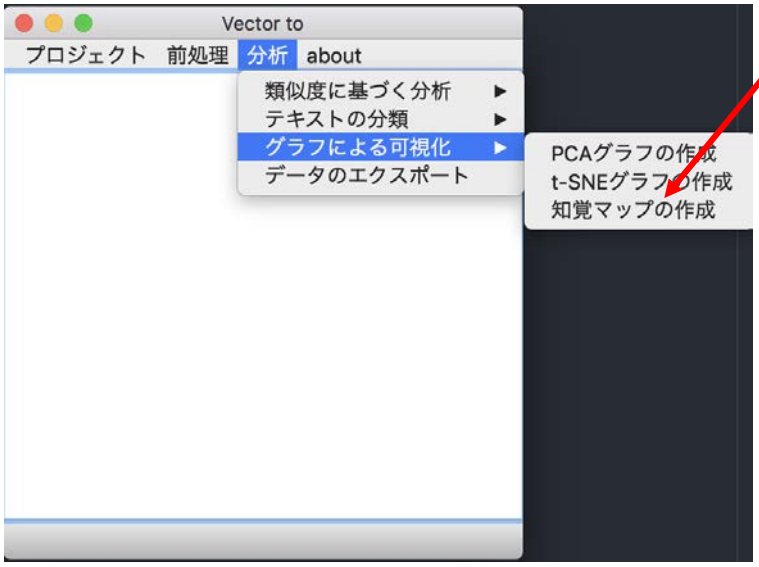
1. メニューバーの分析から「テキストの分類」、「テキストの分類」を選択します。



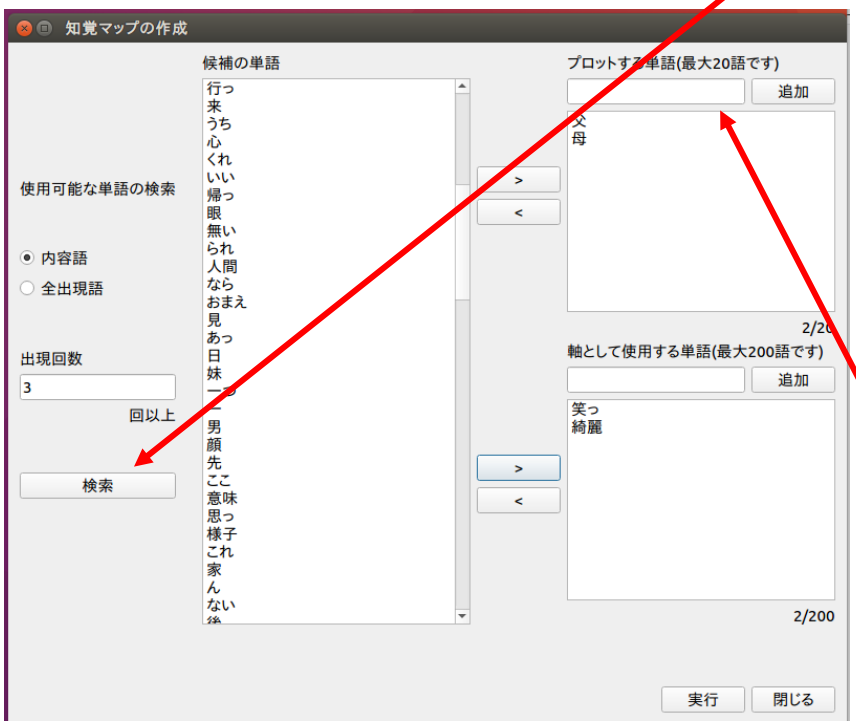
2. 「参照」をクリックし、ressya.txt を選択してください。
3. 「実行」をクリックすると分類が始まります。終了するまでしばらくお待ちください。
4. 分類が終わると結果が下部に表示されます。「列車」の第一段落は 0.53125 で「走れメロス」と推定されています。

知覚マップの作成

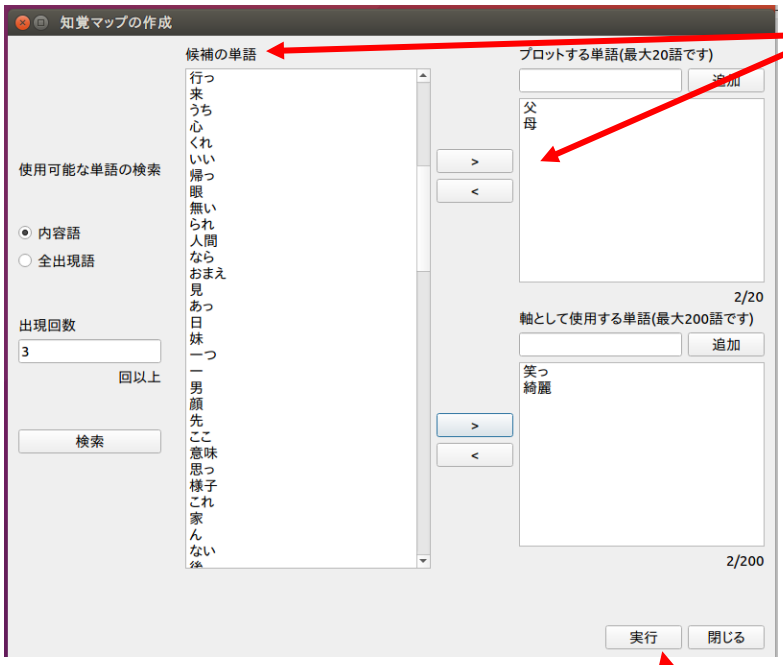
Vector to では学習済みのデータを用いて簡単に知覚マップ(ポジショニングマップ)を作成することができます(分析事例も参照)。



2. メニューバーの「分析」から「グラフによる可視化」→「知覚マップの作成」を選択してください。

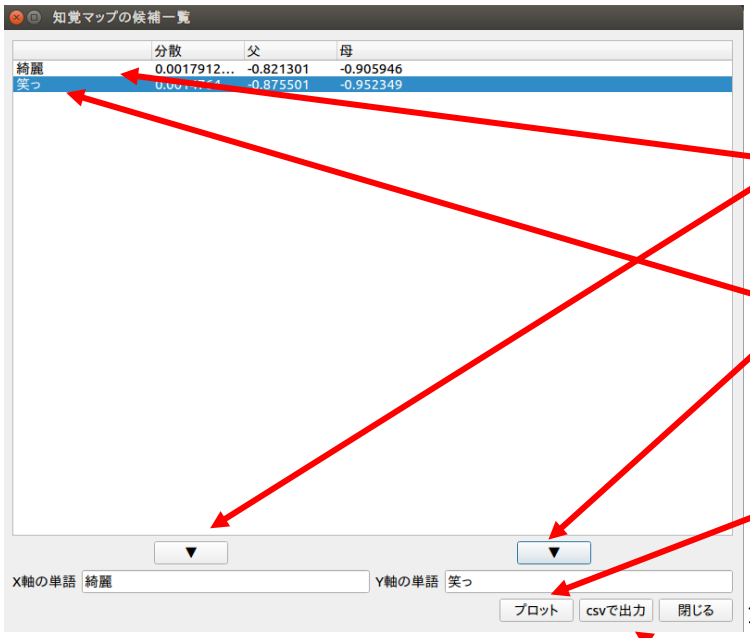


1. 「検索」をクリックすると学習データで使用されている単語の一覧が候補の単語として表示されます。(学習済み bin ファイルを使用しているプロジェクトでは検索機能はご利用いただけません。直接入力し、追加を行ってください)
今回は 3 回以上登場している内容語を対象にしたいと思います。

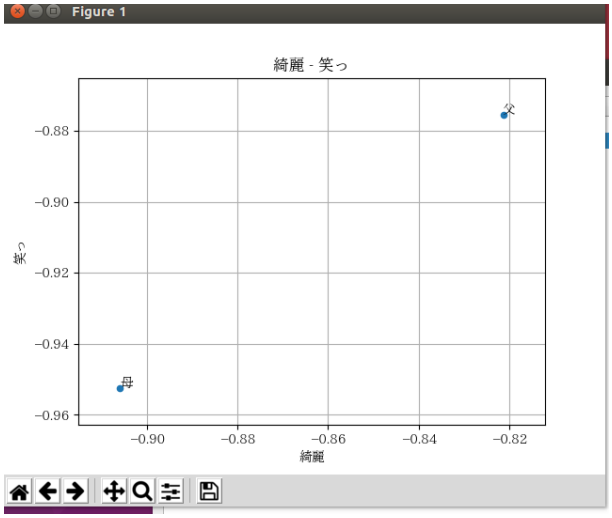


3. 候補の単語欄からプロットしたい単語を選び「＞」ボタンをクリックします。
今回はプロットする単語として「父」と「母」を、軸として使用する単語として「笑っ」と「綺麗」を選択します。
プロットする単語と軸として使用する単語は直接入力することもできます。その際、複数の語を追加する場合は、１語ごとに１回追加を入力するか、スペースで区切って入力し(「父　母」のように)「追加」ボタンをクリックしてください。
不要な単語は「＜」ボタンで候補一覧に戻せます。

4. 「実行」をクリックしてください。少し時間がかかりますので、お待ちください。



5. 知覚マップの候補一覧表が作成されます。
6. 「綺麗」を選択し左側の「▼」ボタンをクリックします。
7. 続いて「笑っ」を選択し、右側の「▼」ボタンをクリックします。
8. 「プロット」をクリックするとグラフが作成されます。
このグラフは簡易的なもので、綺麗なグラフを作成したい場合には「csv で出力」し、あらためてグラフを作成してください。



14 表の見方

表には軸として使用する候補の単語、プロットする単語と軸として使用する単語の類似度、プロットする各単語の類似度の分散が表示されています。　今回、グラフの中にプロットされる単語は「父」と「母」で、知覚マップの軸として利用可能な単語が「綺麗」と「笑っ」になります。　この表を見ると「父」と「綺麗」の類似度が-0.821301、「母」と「綺麗」の類似度が-0.905946、「父×綺麗」と「母×綺麗」の類似度の分散が 0.0017912 であることが示されています。この結果から、「父」「母」ともに「綺麗」と逆のベクトルの要素を単語の構成要因として強く持っており、また、綺麗を構成要因として考えた場合の「父」と「母」には差がないことがわかります。登録されていない単語の場合 0 が表示されます。

Vector to の使用方法

以下ではチュートリアルでは触れることのできなかった操作について簡単に説明と解説を行います。

Vector to の起動

起動の方法は OS によって異なります。

Linux

ターミナルを起動しカレントディレクトリを Vector to にし、python3 Vector_to.py と入力

例：personal フォルダーに Vector to を解凍(unzip)した場合

```
cd /home/personal/vector_to          #Vector_to フォルダーに移動
python3 vector_to.py                 #Vector to の起動
```

macOS

ターミナルを起動し、Vector to フォルダーに移動後、python3 vector_to.py と入力

例：Documents フォルダーに Vector to を解凍(unzip)した場合

```
cd /Users/○○○○  /Documents/vector_to   # Vector_to フォルダーに移動
python3 vector_to.py                   # Vector to の起動
```

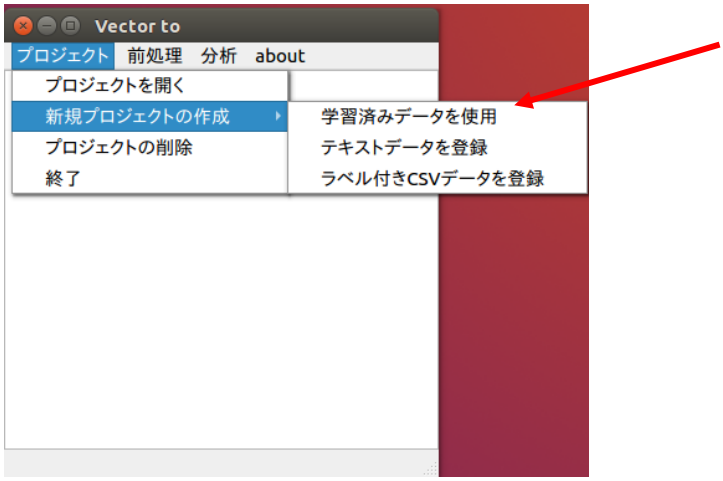
プロジェクトの作成

まず、プロジェクトの作成と分析に使用するテキストファイルや csv ファイルを登録します。Vector to では第 3 者が作成した学習済みファイル(bin ファイル [15])を使用して類似度の計算などを行うこともできます。

Vector to では fastText にはないカテゴリーという枠組みが存在します。これは文章分類に基づくカテゴリー横断分析で使用するものです。

bin 形式の学習済みファイルでプロジェクトを作成

第 3 者の作成した学習済みファイル [16]を使用して類似度の計算を行う場合、この方法でプロジェクトの登録を行ってください。登録する bin ファイルは必ず vector to フォルダ内 fastText フォルダーに置いてください。この方法で作成したプロジェクトではメニューバーの分析にある「類似度」「類似語検索」「単語の加減算」「知覚マップの作成」が可能です。

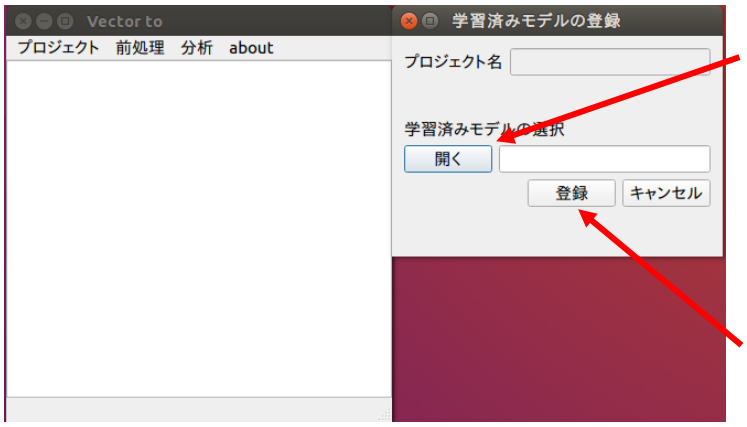


1. メニューバーから「新規プロジェクトの作成」→「学習済みデータを使用」と選択してください。

---

[15] gensim で利用可能な vec 形式の学習済みファイルも配布されていますが、Vector to ではご使用いただけません。bin 形式のファイルをご使用ください。

[16] 例えば Facebook research が次の URL で wikipedia を利用した学習済みファイルを公開しています。この中で Japanese の bin+text をダウンロードして Vector to に登録すれば使用できます。https://github.com/facebookresearch/fastText/blob/master/pretrained-vectors.md
また、pixiv inside では pixiv 小説の本文を学習データとして用いて作成したファイルを公開しています。https://devpixiv.hatenablog.com/entry/2016/09/13/161454
学習対象の違いによる意味の違いなども確認できると思います。

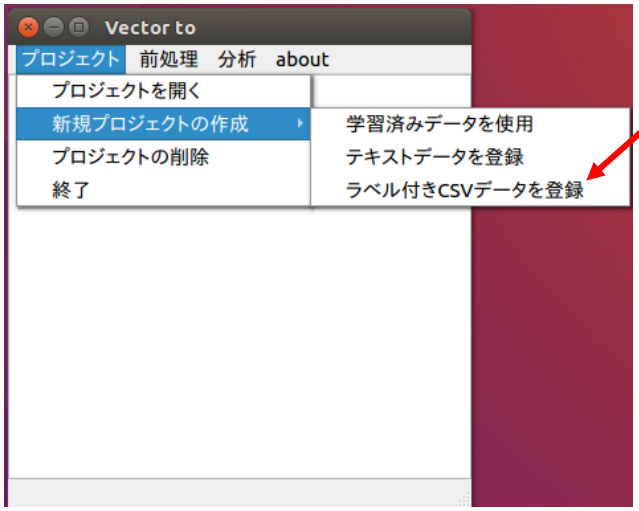


2. 「開く」をクリックし、学習済みbinファイルを選択してください。
bin ファイルのファイル名がプロジェクト名になります。ファイル名に不要な「.(ドット)」がある(「file.name.bin」のようなファイル名) 場合は削除してください。
3. 「登録」をクリックしてください

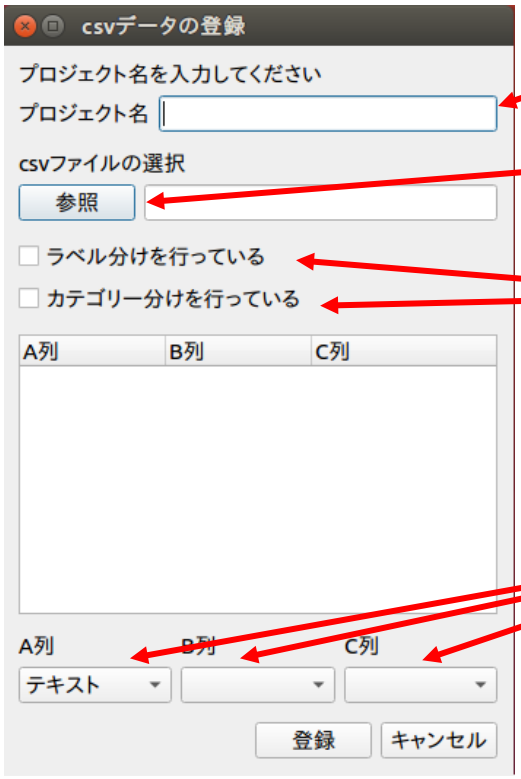
以上でプロジェクトの登録は終了です。学習済みファイルを使用する場合、前処理は必要ありませんので、続いて分析に進んでください。

csv ファイルを分析ファイルとして登録し、プロジェクトを作成

分析対象ファイルに Excel などでラベリングを行った csv ファイルを登録することができます。csv ファイルを用いて分析ファイルを登録する際には、不要なカンマが本文中に存在していないかを十分にチェックしてください。不要なカンマが存在する場合は Vector to でそれが存在する行をエラーメッセージの形で出力しますので、修正のうえ、再度登録してください。

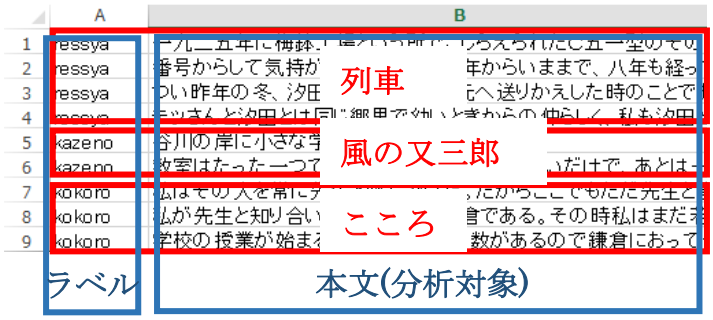


1. 「ラベル付き csv データを登録」を選択



2. プロジェクト名を入力してください（半角英数字）。
3. 「参照」をクリックし、登録する csv ファイルを選択してください。
4. csv ファイル上でラベル名やカテゴリー名を付けている場合はチェックを入れてください。
5. csv ファイルの「A 列」「B 列」「C 列」が「テキスト（本文）」、「ラベル」、「カテゴリー」のどれに当たるかを選択してください。
6. 「登録」をクリックしてください

CSV ファイルの作成例

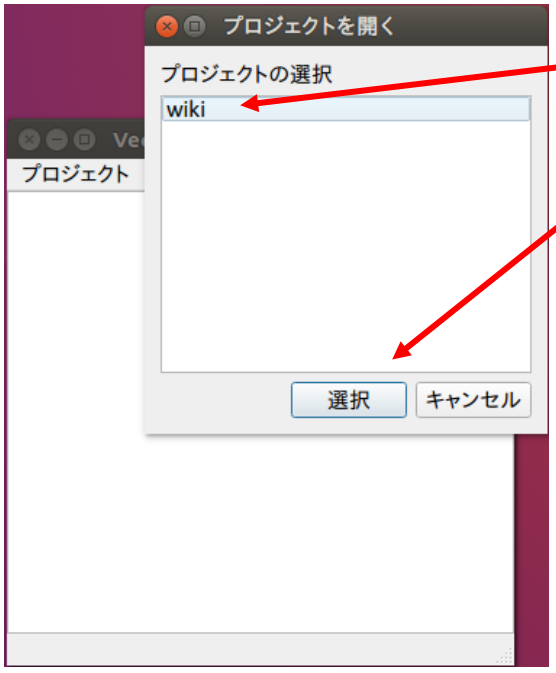


- CSV ファイルでプロジェクトを作成する場合は、ひとつのファイルにまとめてください。
- 見出し行は作成しないでください。
- ラベル名、カテゴリー名は半角英数字をご使用ください。
- 1 行が分析の単位になりますので、ラベル名やカテゴリー名はすべての行に入力する必要があります。

登録済みのプロジェクトを開く

以前に登録したプロジェクトなど、すでに Vector to に登録しているプロジェクトを開く場合はメニューバーの「プロジェクト」から「プロジェクトを開く」を選択してください。

1. 「プロジェクトを開く」を選択



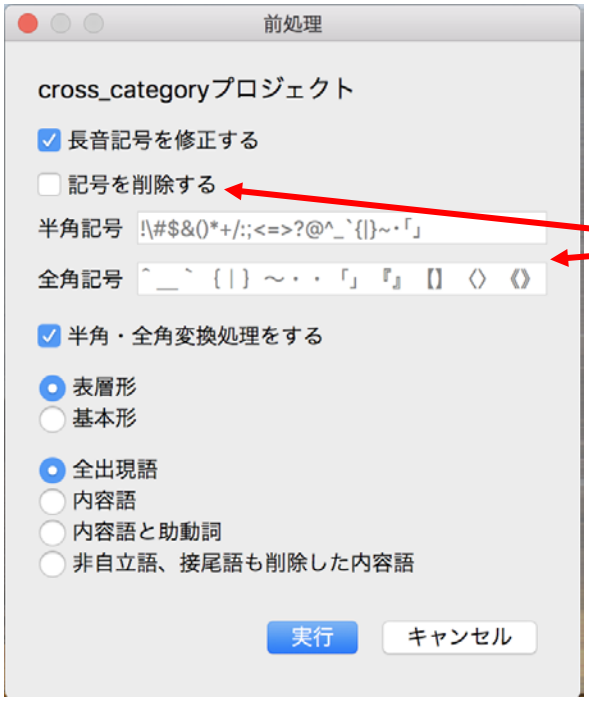
2. プロジェクトを選択
3. 「選択」をクリック

テキストデータの前処理

前処理ではテキストのクリーニング、サンプリング、単語および文章の分散表現化を行います。

テキストのクリーニング

Vector to では①長音記号の統一、②不要な記号の削除といったテキストのクリーニングを行えます。これによってかなりの程度、以後の結果を正確なものにすることができます。しかし、表記ゆれの修正には対応しておりませんので、これへの対応は後述の分散表現化の後に分散表現化の過程で作成される vec ファイルを確認のうえ、ご自身で修正をお願いします。vec ファイルは fasttext フォルダー内に作成されており、ベクトル表現化されたデータも確認することができます。

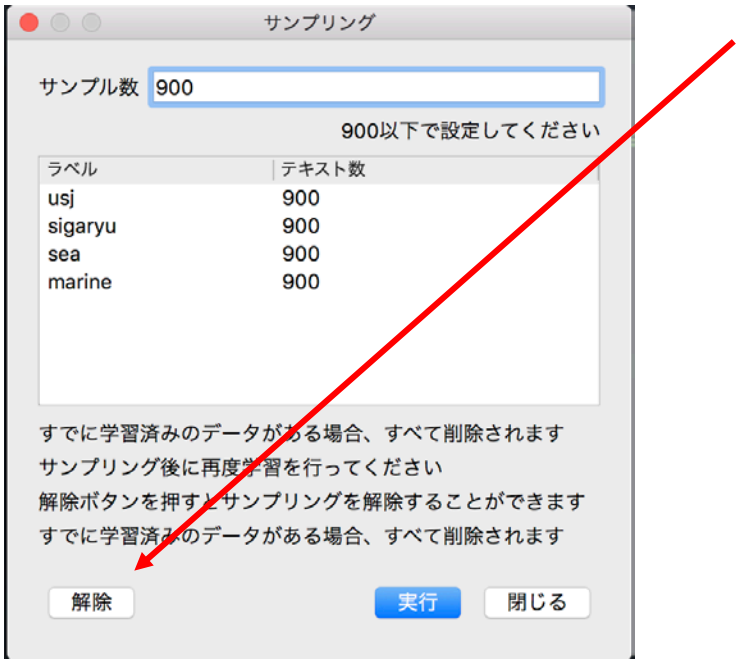


チュートリアルの中でも説明していますので、ここでは補足説明だけにとどめます。

削除する記号ですが、「記号を削除する」チェックボックスにチェックを入れると、削除する記号をご自身で変更することができます。一部記号(「,」)は削除できませんのでご注意ください。(半角と全角の区別は便宜上のものです。)

サンプリング

文章分類を行うためには、教師データとして用いる各文章にラベルを張り、それを学習させる必要があります。教師データに用いる文章数に偏りがある、たとえば A というラベルを張った文章数が 2000、B というラベルを張った文章数が 1000 ある場合、文章数の多寡によるバイアスが学習結果に影響をおよぼす可能性があります。Vector to ではサンプリング機能を用いて各ラベルで学習に使用する文章数をそろえることが可能です。



解除ボタンをクリックすると、以前に行ったサンプリングを解除することができます。しかし、学習済みのデータ等はすべて失われますので注意してください。

解除後に、再度分散表現の取得を行ってください。

fastText による学習

fastText には 2 種類の機能があります。ひとつは単語を分散表現（数百次元のベクトル）に変換し類似度の計算を可能にする機能、もうひとつは教師データに基づいたテキストの分類学習の機能です。fastText には学習方法(コマンド)として supervised、Skip-gram、CBOW があり、Vector to では、supervised と Skip-gram に対応しています[17]。すべての学習方法で単語の分散表現化は行われますが、本来 supervised による学習で得られた単語の分散表現は単語間の類似度の計算やそれにもとづく様々な機能を用いることには適していません。分散表現の取得には後述の分布仮説という考えが背景にあります。Skip-gram や CBOW はこの分布仮説に基づいて分散表現を計算しています。これらの考えに基づいた分散表現の取得は、指定された周辺語を（から）予測するという方法で分散表現を算出しています。言い換えれば、ある意味、周辺語を教師データとして用いることで分散表現を算出しているのです。しかし、supervised では、文章に出現する全単語のベクトル（分散表現）の合計が与えられた教師データ、つまりラベルに応じて 1 or 0 になるように単語の分散表現が算出されます。ゆえに、付与されたラベルに応じてバイアスがかかり、かつ文脈語が文章に登場する全単語ということになるため、単語の分散表現の算出に際する誤差の修正も、Skip-gram や cbow と比べると精度の低いものとなっています。

しかし、Vector to では supervised で得られた単語の分散表現を用いた類似度の計算も可

[17] fastText のチュートリアルによれば、subword 情報を用いた場合、Skip-gram の方がより効果があるとされているため、簡略化もあって CBOW には対応せず Skip-gram だけに対応する形にしています。インターネット上では日本語の subword 情報には対応していないという意見もありますが、pyfasttext モジュールの「model.get_subwords」コマンドで確認したところ、対応していました。

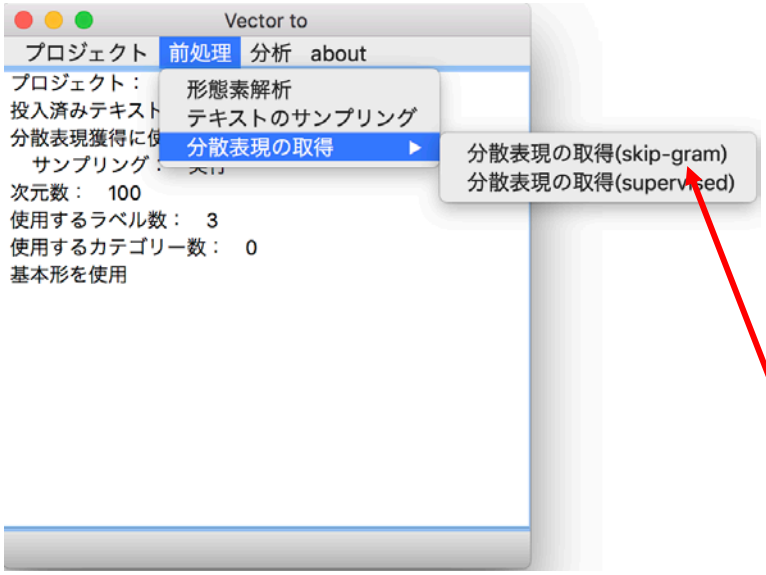
能になっています。これは、文章分類を必要とする分析・研究においても単語間の類似度を測定したいという要求があると考えるからです。また、製作者のあくまでも個人的な印象ですが、supervised によって得られた分散表現も Skip-gram と比較してそれほど劣るものではない、という認識があります。以上の理由から、Vector to では supervised で得られた単語の分散表現でも、類似度の計算や、それらに基づく様々な分析が可能になっています。しかし、先述の通り、supervised で行われる学習では分布仮説に基づいたものではないため、fastText 製作者の意図していない使い方ということになります。分析・研究においては、このあたりは注意する必要があります。

また、supervised で学習された場合には subword 情報を用いた類似度の計算はできません。subword とは単語を文字の羅列と考え、この羅列の中から取り出した一部の文字の羅列のことです。これを利用することで未知の単語にも対応できるようになります。これによって、学習済みの単語と学習されていな単語(未知語)との類似度も計算することができます。

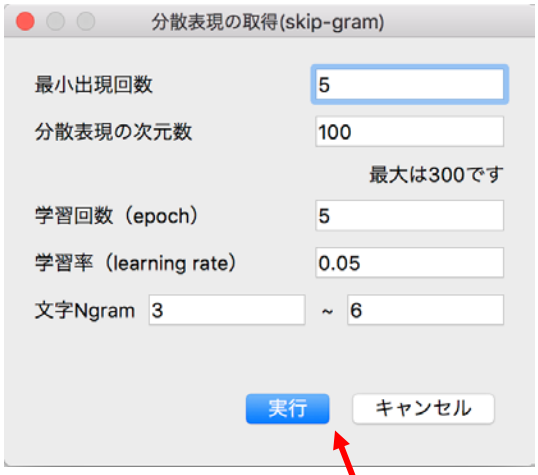
テキスト分類も必要だが、subword 情報も活用したいという要求もあるかとは思いますが、supervised での文章ベクトルの算出方法は、文章を構成する単語のベクトルと EOS(end of sentence)「</s>」のベクトルを合計した後に文章を構成する単語数+1(この+1 は EOS)で除したものを用いています。これら単語ベクトルと文章ベクトルの関係に一貫性を持たせる必要がありますので、文章分類を行いたい場合には supervised で学習を行い、subword 情報を用いた類似度の計算は、再度プロジェクトの登録を行い、実行してください。もし、supervised で学習を行い分析を行った後に、再度同じプロジェクトで skip-gram で学習を行うと、以前の学習内容はすべて失われます。同じデータであっても、異なる学習を行う場合には十分に注意してください。また、先述の通り、fastText は初期化の処理の中でランダム処理が入っているので、同じ学習結果を得ることはほぼ不可能だと考えてよさそうです。再学習を行う際には、十分に注意してください。新しいプロジェクトを登録することをお勧めします。

supervised による学習も skip-gram による学習もともに単語の分散表現化は可能であり、類似度の計算などは可能ですが、skip-gram による分散表現化が優れている点が先述の未知語への対応です。たとえば次章では開発者が以前に行った分析の事例が掲載されていますが、その分析の中では様々なテーマパーク等のクチコミを分析しました。この中では東京ディズニーランド等の東京に関係する施設名への言及があり、東京ディズニーランドと東京ディズニーシーの類似度なども計算することができます(類似度は 0.73436 です)。しかし、この分散表現化に当たっては supervised を用いたため、たとえばクチコミとして登場していない東京バナナとの類似度は計算することができませんでした。skip-gram を用いると subword 情報を活用して(意味があるかどうかや、カテゴリーの異なるものを比較して精度が確保されるのかといった問題はおいておくとして)東京ディズニーランドと東京バナナの類似度を計算することもできます。

skip-gram による学習



1. 必ずラベル登録のされていないプロジェクトで行ってください。ラベル登録されている場合には、自動的に supervised で計算されます。
2. 「分散表現の取得(skip-gram)」を選択します。



3. ハイパーパラメーターを調整することで分析の精度を上げることができます。
4. ここで学習率を上げ過ぎるとメモリ不足のエラーが発生します。ターミナル上で学習の進み具合が表示されていますが、一番左の列 Progress が 100%まで進まない場合は、fastText が強制終了しており、Vector_to 上で「処理が終了しました」と表示されても学習はされていません。学習率を小さくして、再度実行してください。

- 文字 Ngram
  - 何文字単位で subword 情報として活用するのかを決定します。日本語の場合 3 から 4 あたりに設定するのが妥当でしょうか。最小を 2 にすると未知語に強いモデルを作成できますが、日本語の単語は 2 文字で意味を構成するものが多いため、類似度を計算する際にノイズが多く入るようです。

5. 「実行」をクリックします。

分析

カテゴリー横断文章分析

チュートリアルでは、使用可能な文章の限界から、この分析手法の説明は省きました。この分析手法を用いてできることに関しては分析事例を見ていただくこととし、ここでは簡単な説明を行います[18]。

この分析を行うにあたっては、各文章に対して、ラベルに加えてカテゴリーも登録します。下記の表は、分析事例の中でのラベルとカテゴリーの関係です。東京ディズニーランドの各クチコミには「東京ディズニーランドのラベル(実際の分析では半角英数字を使用します)」が、東京ディズニーシーの各クチコミには「東京ディズニーシー」のラベルが貼られ、それに加えて両ラベルのクチコミに対して「テーマパーク・レジャーランド」というカテゴリーが貼られます。

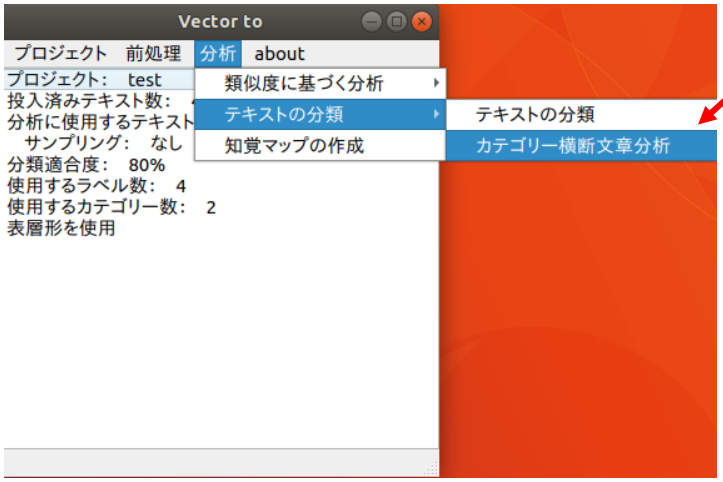
カテゴリー	ラベル
テーマパーク･レジャーランド	東京ディズニーランド
テーマパーク･レジャーランド	東京ディズニーシー
テーマパーク･レジャーランド	ユニバーサル･スタジオ･ジャパン
テーマパーク･レジャーランド	横浜･八景島シーパラダイス
テーマパーク･レジャーランド	ナガシマスパーランド
水族館	沖縄美ら海水族館
水族館	鳥羽水族館
水族館	鴨川シーワールド
水族館	海遊館
水族館	名古屋港水族館
動物園･植物園	アドベンチャーワールド
動物園･植物園	旭川市旭日山動物園
動物園･植物園	神戸市立王子動物園
動物園･植物園	東山動植物園
動物園･植物園	上野動物園
アウトレットモール	神戸三田プレミアム･アウトレット
アウトレットモール	三井アウトレットパークジャズドリーム長島
アウトレットモール	三井アウトレットパーク木更津
アウトレットモール	三井アウトレットパーク滋賀竜王
アウトレットモール	三井アウトレットパークマリンピア神戸

fastText の通常の学習と分類機能では、ラベル付きテキストを教師として、テキストのラベルごとの特徴を計算し、その後の分類ではラベルなしテキストが投入され、それが学習結果に基づいて分類されます。これはラベルに「属するもの」という観点、つまり排他的な特徴という観点からの分析になります。Vector to でのカテゴリー横断分析は、これとは反対にどのラベルにも共通するため「分類できないもの」、あるいは「どのラベルにも属さない

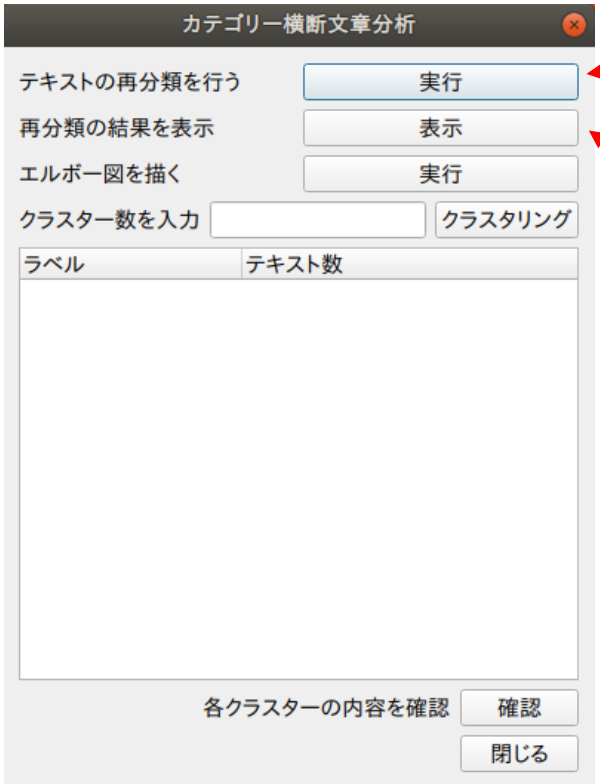
18 この使用方法の説明の中でのデータと分析事例で使用しているデータは異なります。

もの」という観点からの分析ということができます。

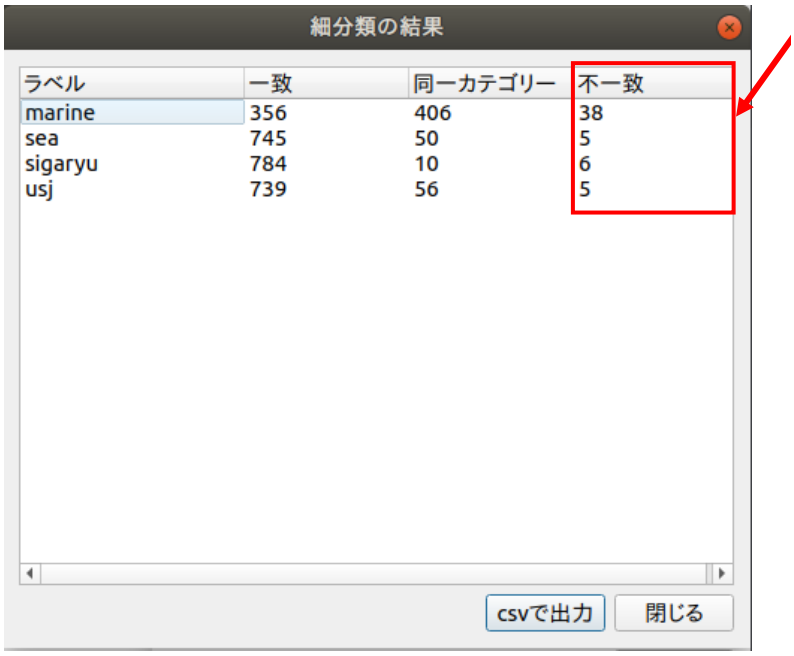
具体的な分析としては、学習に利用したテキストデータを、再度 fastText によって分類させ、それぞれの文章がどのラベルになるのかを判定します。この判定では、「ラベルが一致した」、「ラベルは一致しなかったが同一のカテゴリーのラベルが張られた」、「カテゴリーも異なるラベルが張られた」、の 3 通りで判定します。そして、判定の結果をもとにカテゴリーが異なるラベルを張られたものを分析の対象として、それらに共通する要因を探り出します。



8. メニューバーの「分析」から「テキストの分類」→「カテゴリー横断文章分析」を選択してください。



9. 学習に利用したファイルの再分類(ラベリング)を行います。「実行」をクリックします。
10. 再分類の結果を確認します。「表示」をクリックしてください。



11. ラベルが一致したテキスト数、同一カテゴリーとして判定されたテキスト数、カテゴリーの異なるラベルとして判定されたテキスト数が表形式で出力されます。続く分析では、ここで不一致と分類されたデータを分析の対象とします。

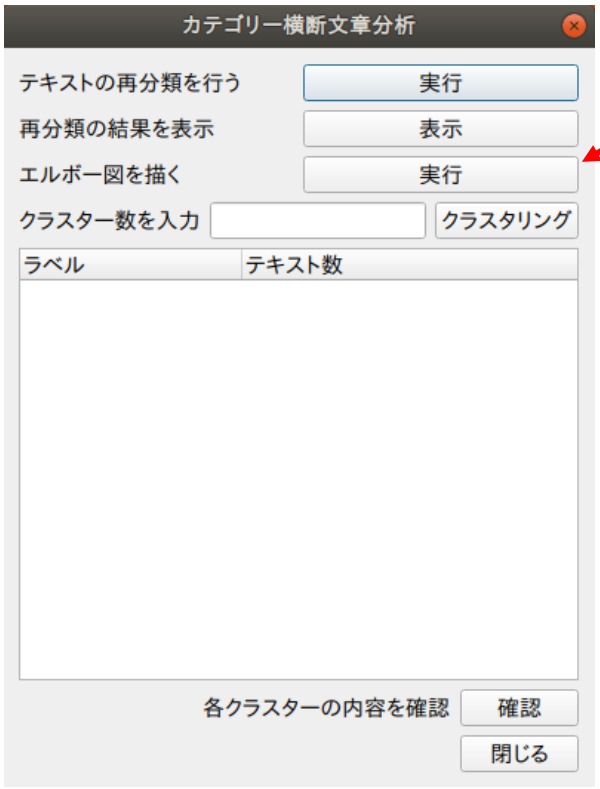
fastText は各テキストを指定された次元のベクトルに変換し、それを教師データとして学習し、分類に使用しています。そのため、分析者が、どのようなテキストが不一致カテゴリーに分類されたのかを確認しようとしても、ベクトル形式でしか見ることができません。たとえば、下記は「走れメロス」の一節と、それを 100 次元のベクトル表現に変換したものです。このベクトル表現から何らかの示唆を得ることは難しいと思います。

例：

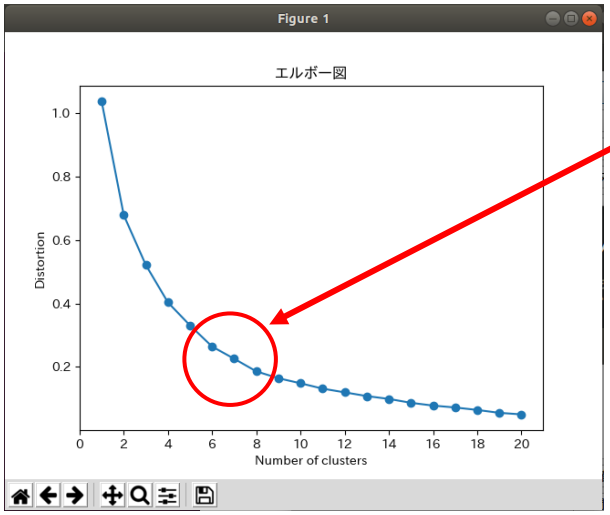
メロスは、単純な男であった。買い物を、背負ったままで、のそのそ王城にはいって行った。たちまち彼は、巡邏の警吏に捕縛された。調べられて、メロスの懐中からは短剣が出て来たので、騒ぎが大きくなってしまった。メロスは、王の前に引き出された。　　「走れメロス」の一節

[0.00046942 0.0032485 0.00014361 -0.0027153 0.0035908 0.0044724 -0.0011215 -0.00088636 0.0037339 0.0038991 -0.0032646 0.0007755 -0.00076629 0.0030221 -0.00051451 -0.0011642 0.0004482 -0.00053487 -0.0011027 1.2658e-05 0.0015936 0.0049606 0.00097712 -0.00045528 -0.0023102 0.00019531 0.0014424 -0.00063197 -0.0035805 0.0026032 -0.0011044 -0.00025962 -0.001738 -0.0024207 -0.0053958 0.0021536 -0.0013145 -0.00095922 3.1509e-05 0.0019281 -0.0018763 0.0038084 0.0017963 0.0024844 0.0050655 -0.000566 0.0050953 -0.0010162 -0.0012072 0.0027825 -0.0039259 0.0021719 -0.0016006 0.001074 -0.001168 0.00020393 -0.0018611 0.0014714 0.0027242 0.00065239 -0.0017154 0.0030112 0.0034748 0.0015612 0.00017073 0.00016752 -0.0045199 -0.0010992 -0.001105 -0.00078624 -0.0024343 0.0015289 0.0013716 -0.00079297 -0.0013762 -0.00056616 0.0020952 0.00091085 0.0029211 -0.0026901 -0.00029499 0.0014438 0.0022512 0.00057329 0.0028533 -0.00088012 0.00059071 0.002693 -0.0032757 -0.0003471 0.0028676 0.00054188 0.003604 -0.0011732 0.0016139 -0.0023582 -0.0012136 0.0016602 -0.0036149 -0.0033901]

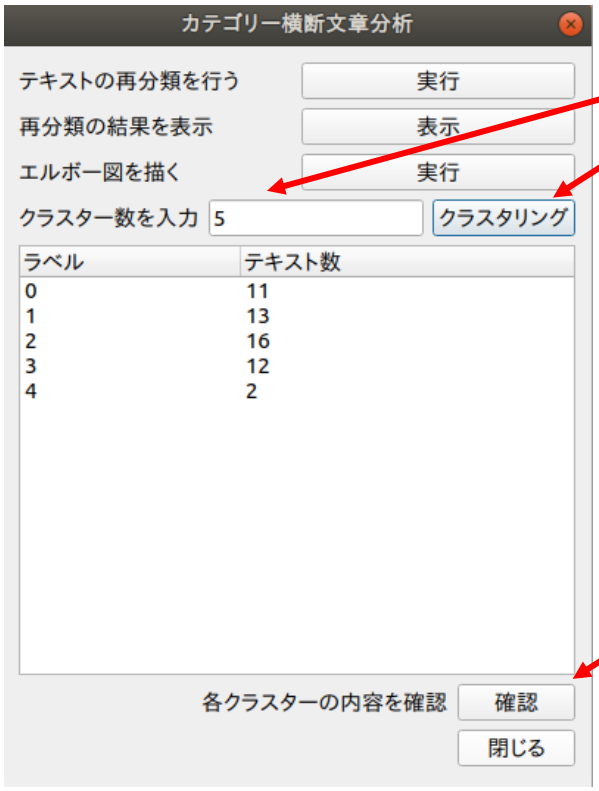
しかし、ラベルだけではなくカテゴリーも不一致となったテキストの集合を分類することができれば、どのラベルやカテゴリーにも分類できないテキストの特徴を抽出することが可能です。そこで Vector to では、このようなテキストを k-means 法で分類し、その分類結果を計量テキスト分析の手法で可視化できるようにしています。



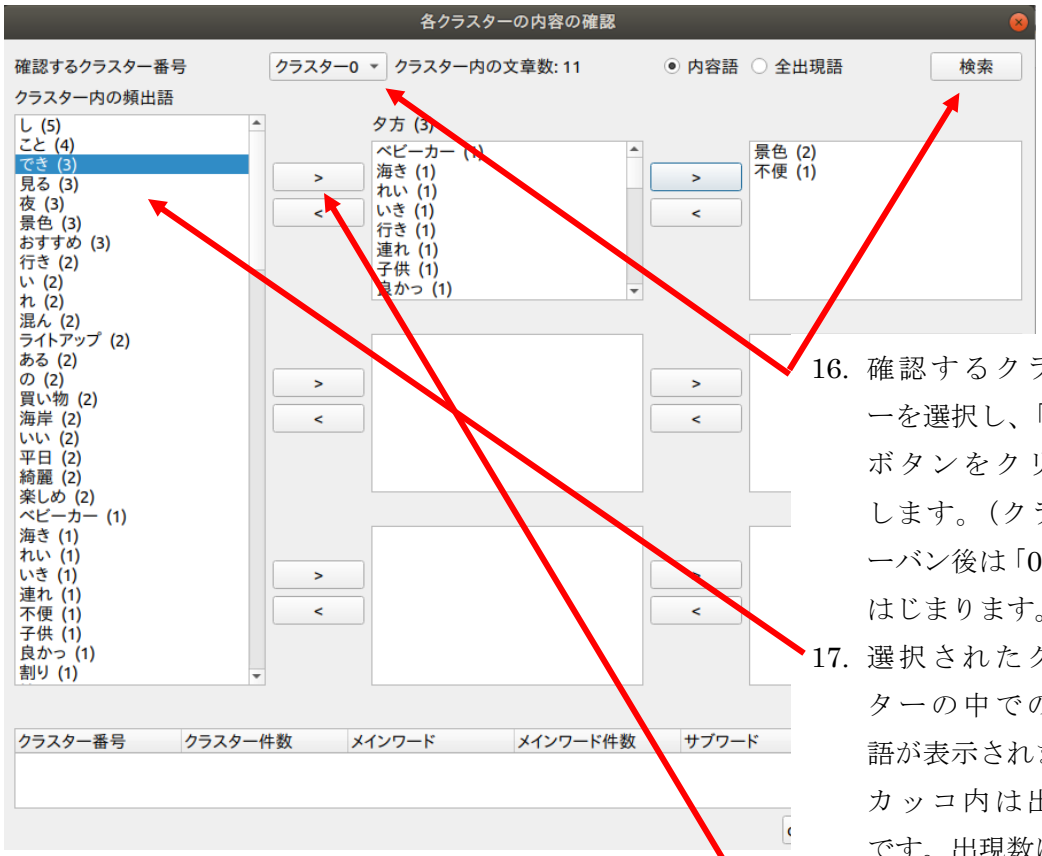
12. エルボー図を確認することで、不一致だったテキストをいくつに分類するかを検討します。エルボー図の作成には k-means 法を用いています。



13. この図では 5 から 8 クラスターに分類することが妥当と思われます。



14. クラスター数を入力し、「クラスタリング」ボタンをクリックすると、それぞれのクラスターに分類されたテキスト数が表示されます。ここでクラスター数を判断することができます。
15. さらに「確認」ボタンを押して、各クラスターの中身を頻出語と、その頻出語と共起する単語の関係で確認することができます。



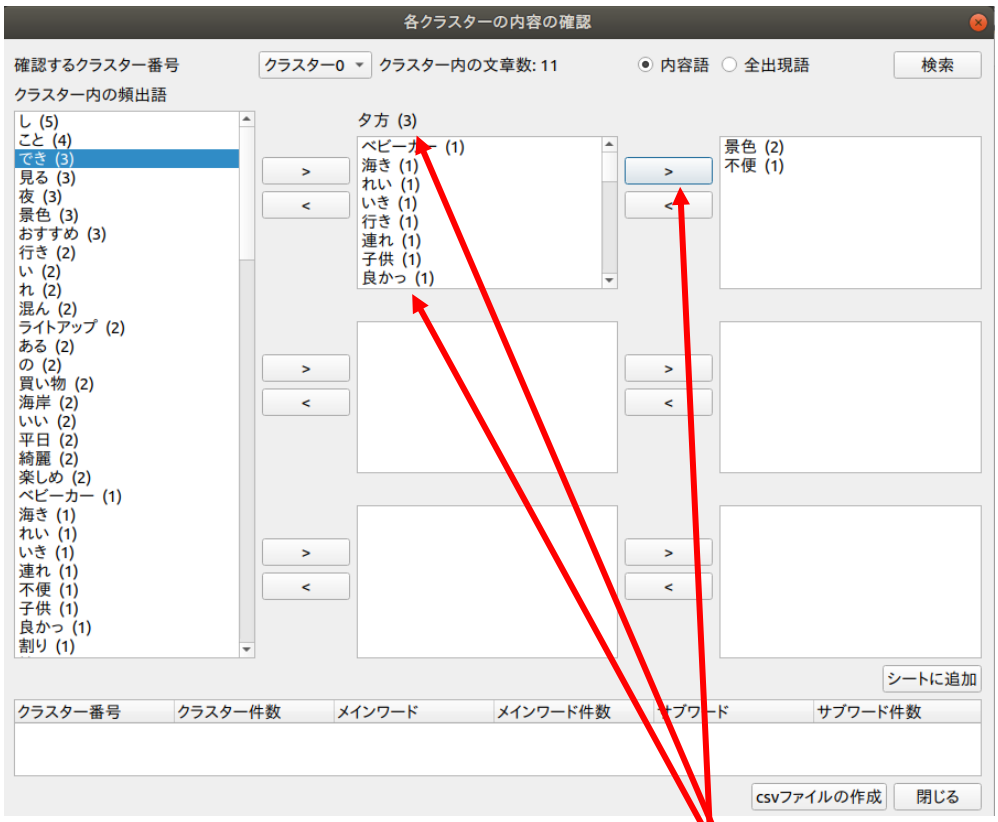
16. 確認するクラスターを選択し、「検索」ボタンをクリックします。（クラスターバン後は「0」からはじまります。）
17. 選択されたクラスターの中での頻出語が表示されます。カッコ内は出現数です。出現数には総出現回数ではなく、出現テキスト数を用いています。
18. 頻出語から単語を選択し「＞」ボタンをクリックすると、その頻出語と共起する単語が表示されます。間違って「＞」を押した単語は「＜」で戻すことができます。

出現回数のカウント方法

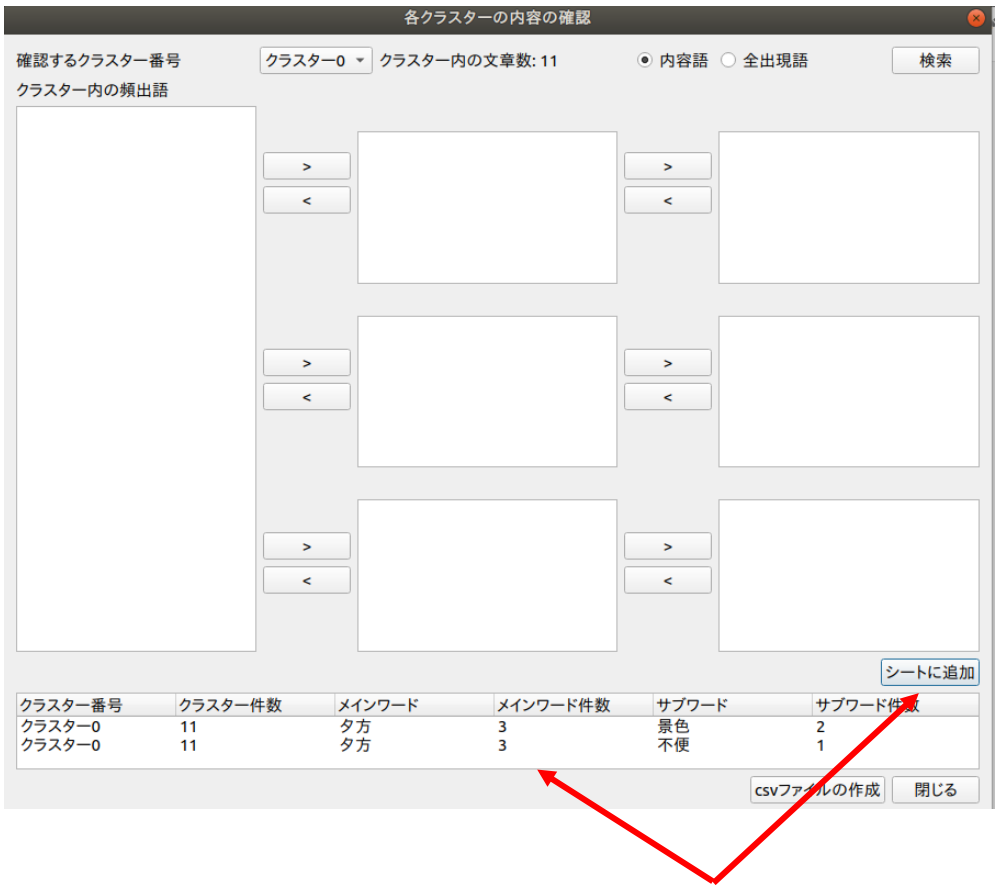
ひとりの少女が、緋のマントをメロスに捧げた。メロスは、まごついた。佳き友は、気をきかせて教えてやった。

（「走れメロス」より）

この場合、「メロス」という単語の総出現回数は 2 回ですが、ひとつのテキスト（段落）中なので 1 回と数えます。



19. 頻出語として選ばれた単語(「夕方」)と共起する語が一覧で表示されます。
20. 「＞」をクリックすると、選択された共起語が右欄に移ります。
21. 上記図の場合、頻出語として「夕方」が選択され、「夕方」と共起する語として「景色」と「不便」が選択されています。



22. 「シートに追加」ボタンをクリックすると下の表に、選択した内容が移動します。
23. 確認したいクラスターの内容を確認し終えたのちに「csv ファイルの作成」ボタンをクリックすると表の内容を csv ファイルとして保存することができます。

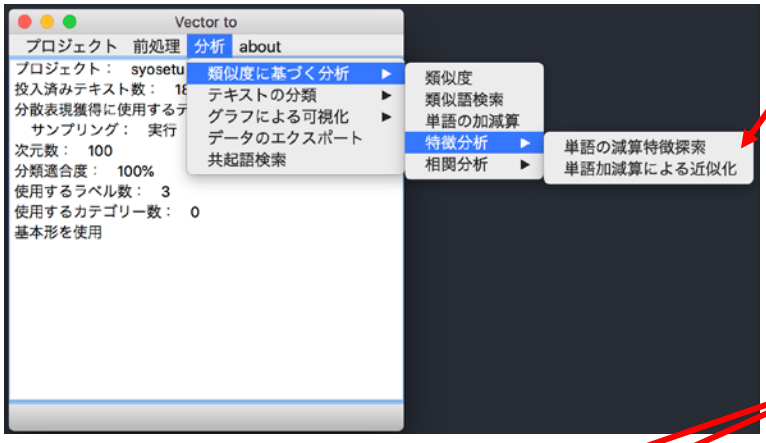
類推問題（analogy task）を解くことによる特徴の可視化

fastText など分散表現を用いた自然言語処理の基本的な機能として類推問題を解くというものがあります。類推問題とは「王様」－「男性」＋「女性」は？という単語間の関係を推測するもので、上記の答えは「女王」になります。Vector to にはこの類推問題を解くことから単語の特徴を可視化する機能が 2 つあります。

これまで使用してきた小説データを用いて行った事例が次のものです。

単語の加減算による近似化

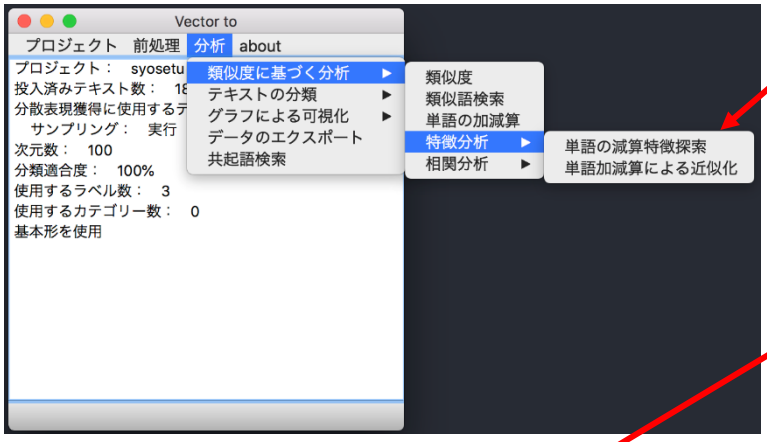
ひとつめの類推問題を用いた特徴の可視化では、単語を二つ指定し、第一語が第二語に近づくためには、どのような単語を加算したり減産すればよいかを探索します。



1. 「分析」から「類似度に基づく分析」「特徴分析」「単語の加減算による近似化」と進んでください。
2. 今回は第一語に「メロス」を第二語に「セリヌンティウス」を入れて分析しました。
3. この結果から、「メロス」という概念が「セリヌンティウス」という概念になるためには「メロス」から「それ」を減算し、さらに「上」を減算することで「セリヌンティウス」と 0.99 の類似度を得る概念になることがわかります

単語の減算による近似化

この機能は、指定した単語らしさをもっとも失わせる概念は何かを探索するものです。



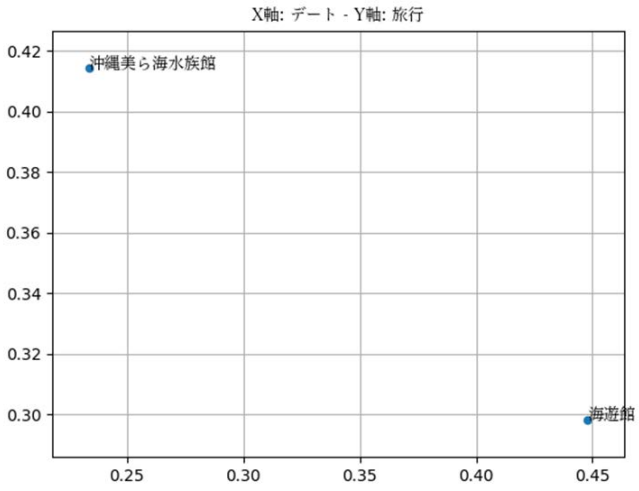
1. 「分析」から「類似度に基づく分析」「特徴分析」「単語の減算特徴探索」と進んでください。
2. 特徴を探索する語に「メロス」と入力し「探索」をクリックします。
3. ここでは「メロス」から「待つ」と「殺す」という概念を減算すると「メロス」から最も遠ざかることがわかります。

代理変数を用いた特徴の抽出

これまでの計量に基づくテキストマイニングでは、分析対象の特徴を抽出するために、単語の出現回数（例：「大きい」よりも「小さい」という単語の登場回数が多ければ、その分析対象の特徴として「小さい」が強い）や共起関係（〇〇という単語と一緒に登場している）、用いることでアプローチしてきました。分散表現テキストマイニングでは、これまでとは異なる方法で分析対象の特徴にアプローチすることができます。

この新しいアプローチである相関分析の説明に入る前に、その前提となる特徴の代理変数化について説明します。方法としては、文章中に登場する商品やサービスの特徴を、特徴となる語と商品・サービス名の類似度で代理する、つまり商品・サービスの持つ特徴の代理変数として類似度を用いる手法を採用するというものです。具体的にはテキスト中に登場するサービス名や施設名と様々な出現語の類似度をサービスや施設の特徴の代理変数として使用することで、施設の特徴を明確にする手法として採用します。ここで特徴とは「近い」や「安い」、「デート」、「旅行」など文章中に出現する単語を指しています。

たとえば、後述の外形的データを併用した分析の節で使用する学習済みモデルを用いると、「海遊館」という単語と「デート」という単語の類似度は 0.45 で、同「海遊館」と「旅行」の類似度は 0.30 です。また「沖縄美ら海水族館」と「デート」の類似度は 0.23 で、同「沖縄美ら海水族館」と「旅行」との類似度は 0.41 です。

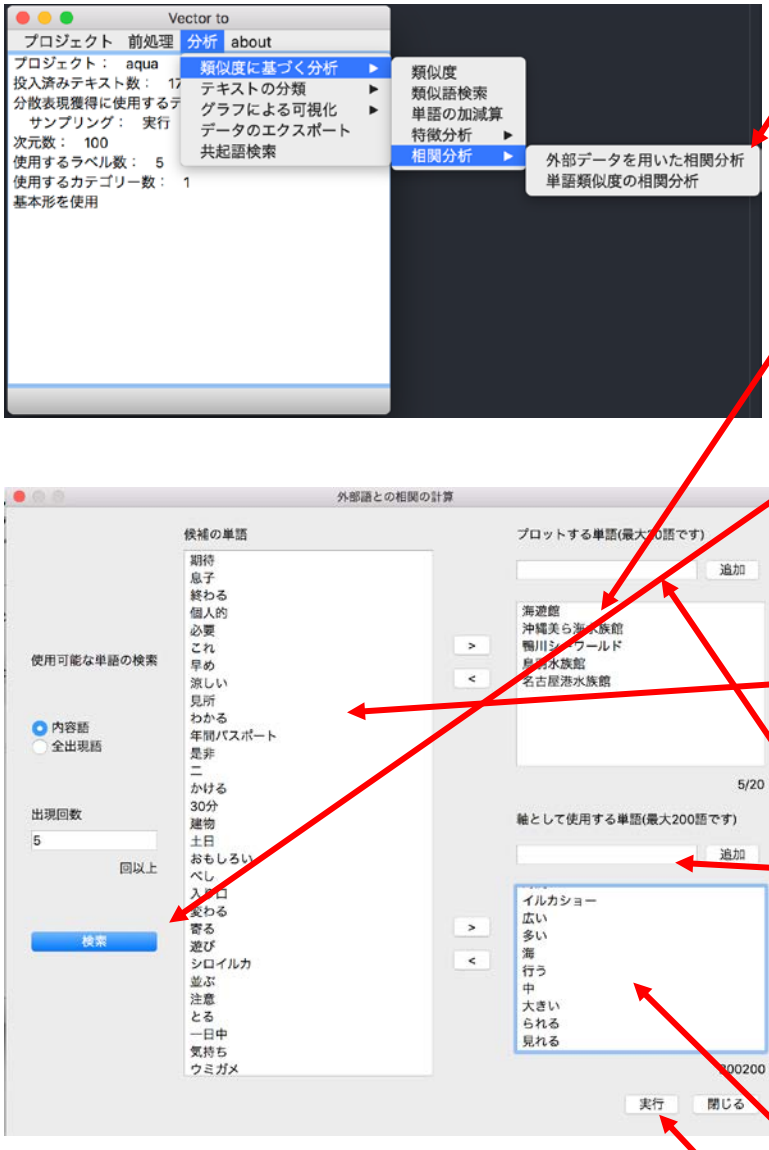


この計算結果から、海遊館と沖縄美ら海水族館の相対的な特徴としては、沖縄美ら海水族館が海遊館と比較して旅行客が訪れる施設という特徴があり、海遊館がデートで訪れられる施設という特徴があるという可能性を推測することができます。このような分析は、単語が様々な成分から構成されており、それら成分と、その影響の程度によって意味が決まっていることによって可能となります。

Vector to では、このように取得された様々な特徴と、他の要素との相関を計算することができます。例えば、竹岡(2018b)および竹岡(2019)では 5 つの水族館の様々な特徴を上記手法で測定し、それらの特徴と「入場者数」、「延べ床面積」、「飼育種類数」という外形的データとの相関を計算することで、消費者が水族館というサービスを消費するにあたってがどのような要因が影響を与えているのかを分析しました。

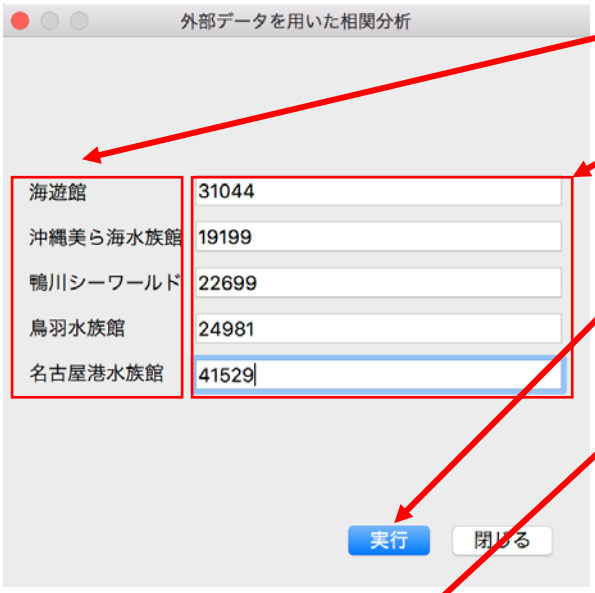
外部データを用いた相関分析

今回は、これまで使用してきた小説では困難な分析なので、後述の「外形的データを併用した分析」の事例を行った際の手順を説明します。

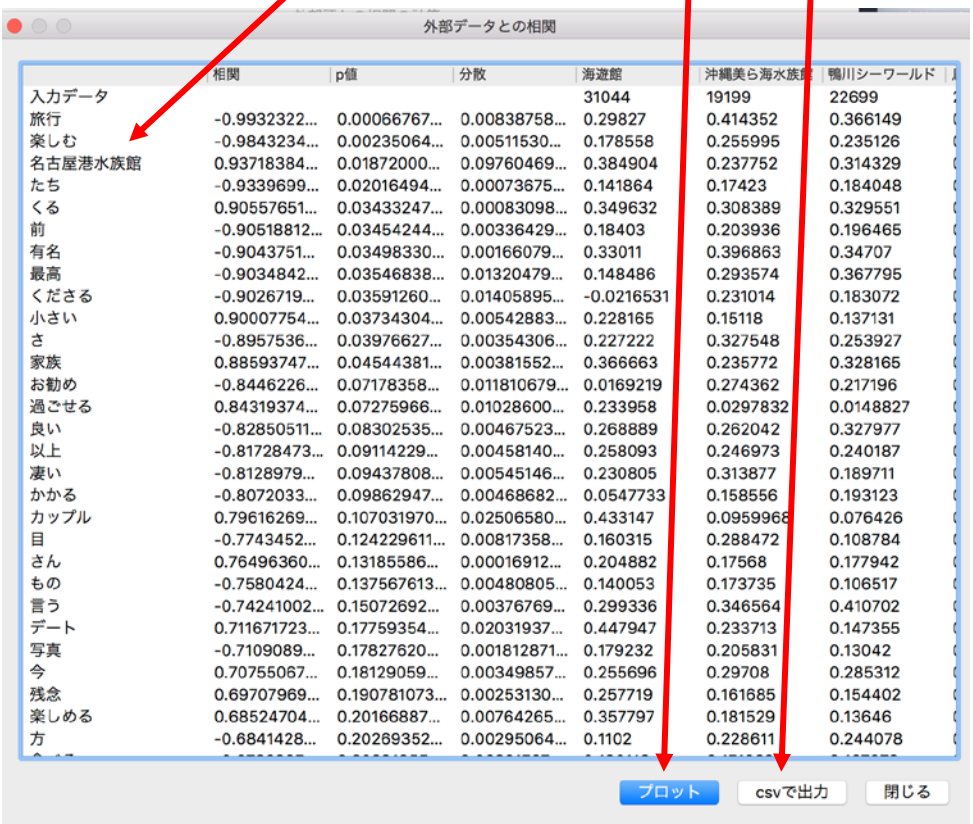


4. 「分析」から「相関分析」と進み、「外部データを用いた相関分析」をクリック。

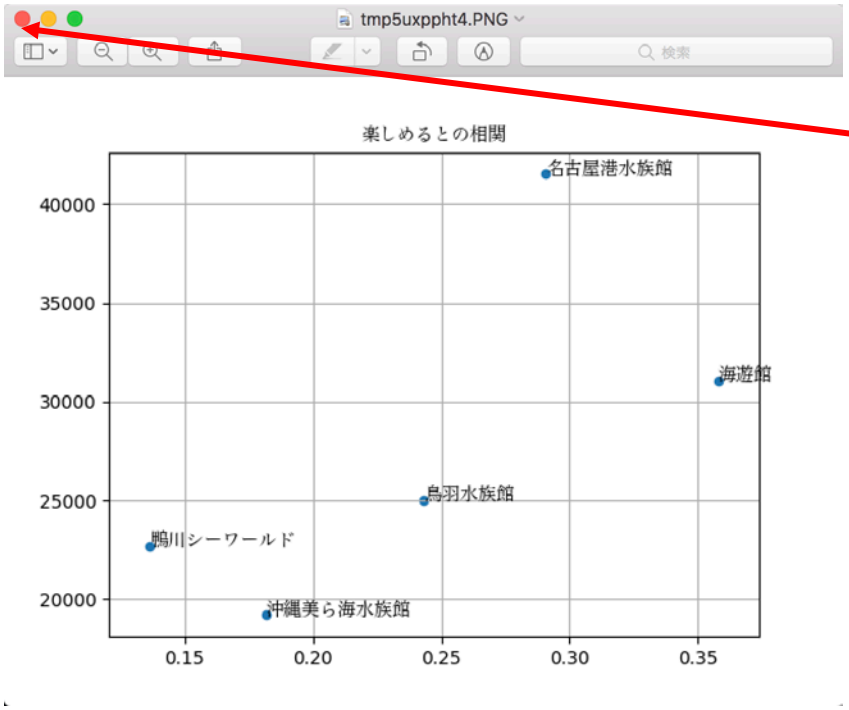
1. 特徴を知りたい施設名やサービス名を登録します。
   - 「検索」をクリックすると学習に使用した単語を一覧で取得できます。
   - 候補の単語を選択し、矢印ボタンで追加できます。
   - 直接入力することもできます。複数の語を直接入力する場合はスペースを入れてください。
2. 特徴として抽出する概念を登録します。
3. 「実行」をクリックします。



5. 特徴を知りたい施設名などが表示されますので、その横にデータを入力してください。今回は各施設の「延べ床面積」を入力しました。
6. 「実行」をクリックします。
7. 相関、あるいは逆相関の強い順にリスト表示されます。
8. 入力データとの関係を散布図として表示する際には、単語を選択し、「プロット」をクリックしてください。
9. この表は csv 形式で保存することができます。



	相関	p値	分散	海遊館	沖縄美ら海水族	鴨川シーワールド
入力データ				31044	19199	22699
旅行	-0.9932322...	0.00066767...	0.00838758...	0.29827	0.414352	0.366149
楽しむ	-0.9843234...	0.00235064...	0.00511530...	0.178558	0.255995	0.235126
名古屋港水族館	0.93718384...	0.01872000...	0.09760469...	0.384904	0.237752	0.314329
たち	-0.9339699...	0.02016494...	0.00073675...	0.141864	0.17423	0.184048
くる	0.90557651...	0.03433247...	0.00083098...	0.349632	0.308389	0.329551
前	-0.90518812...	0.03454244...	0.00336429...	0.18403	0.203936	0.196465
有名	-0.9043751...	0.03498330...	0.00166079...	0.33011	0.396863	0.34707
最高	-0.9034842...	0.03546838...	0.01320479...	0.148486	0.293574	0.367795
くださる	-0.9026719...	0.03591260...	0.01405895...	-0.0216531	0.231014	0.183072
小さい	0.90007754...	0.03734304...	0.00542883...	0.228165	0.15118	0.137131
さ	-0.8957536...	0.03976627...	0.00354306...	0.227222	0.327548	0.253927
家族	0.88593747...	0.04544381...	0.00381552...	0.366663	0.235772	0.328165
お勧め	-0.8446226...	0.07178358...	0.011810679...	0.0169219	0.274362	0.217196
過ごせる	0.84319374...	0.07275966...	0.01028600...	0.233958	0.0297832	0.0148827
良い	-0.82850511...	0.08302535...	0.00467523...	0.268889	0.262042	0.327977
以上	-0.81728473...	0.09114229...	0.00458140...	0.258093	0.246973	0.240187
凄い	-0.8128979...	0.09437808...	0.00545146...	0.230805	0.313877	0.189711
かかる	-0.8072033...	0.09862947...	0.00468682...	0.0547733	0.158556	0.193123
カップル	0.79616269...	0.107031970...	0.02506580...	0.433147	0.0959968	0.076426
目	-0.7743452...	0.124229611...	0.00817358...	0.160315	0.288472	0.108784
さん	0.76496360...	0.13185586...	0.00016912...	0.204882	0.17568	0.177942
もの	-0.7580424...	0.137567613...	0.00480805...	0.140053	0.173735	0.106517
言う	-0.74241002...	0.15072692...	0.00376769...	0.299336	0.346564	0.410702
デート	0.711671723...	0.17759354...	0.02031937...	0.447947	0.233713	0.147355
写真	-0.7109089...	0.17827620...	0.001812871...	0.179232	0.205831	0.13042
今	0.70755067...	0.18129059...	0.00349857...	0.255696	0.29708	0.285312
残念	0.69707969...	0.190781073...	0.00253130...	0.257719	0.161685	0.154402
楽しめる	0.68524704...	0.20166887...	0.00764265...	0.357797	0.181529	0.13646
方	-0.6841428...	0.20269352...	0.00295064...	0.1102	0.228611	0.244078

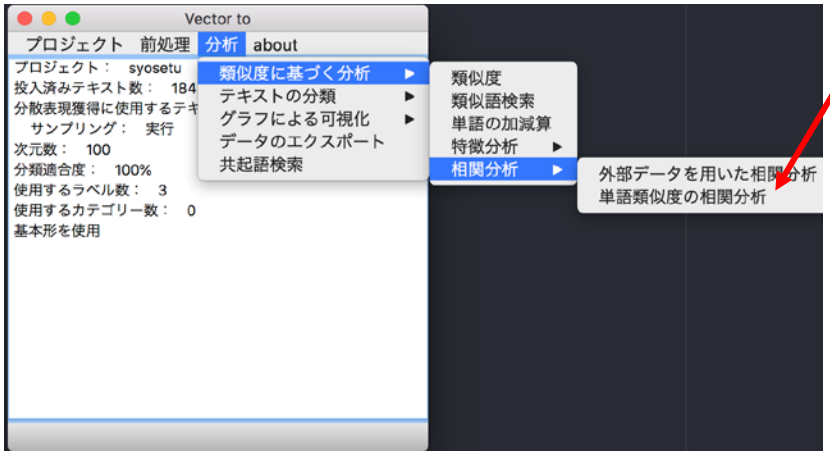


10. 散布図が得られました。
11. 散布図の確認が終わりましたら、必ずウィンドウを閉じてください。

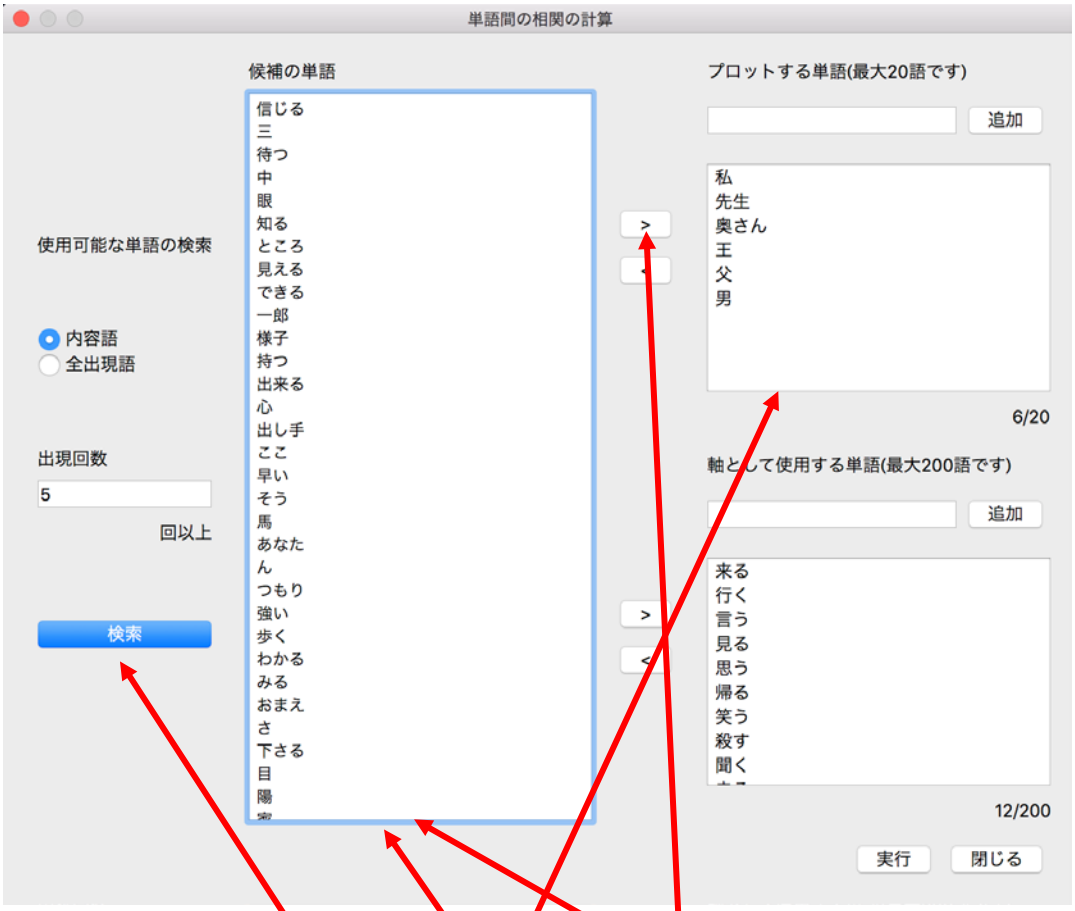
出現語間の相関分析

続いて、代理変数を用いた特徴分析の中でも、出現語間の相関分析について説明します。これは、様々な特徴間には、「かわいい」と「好き」のように関連のあるものが存在します。このような特徴間の関係を相関分析によって明らかにするのが、この機能です。

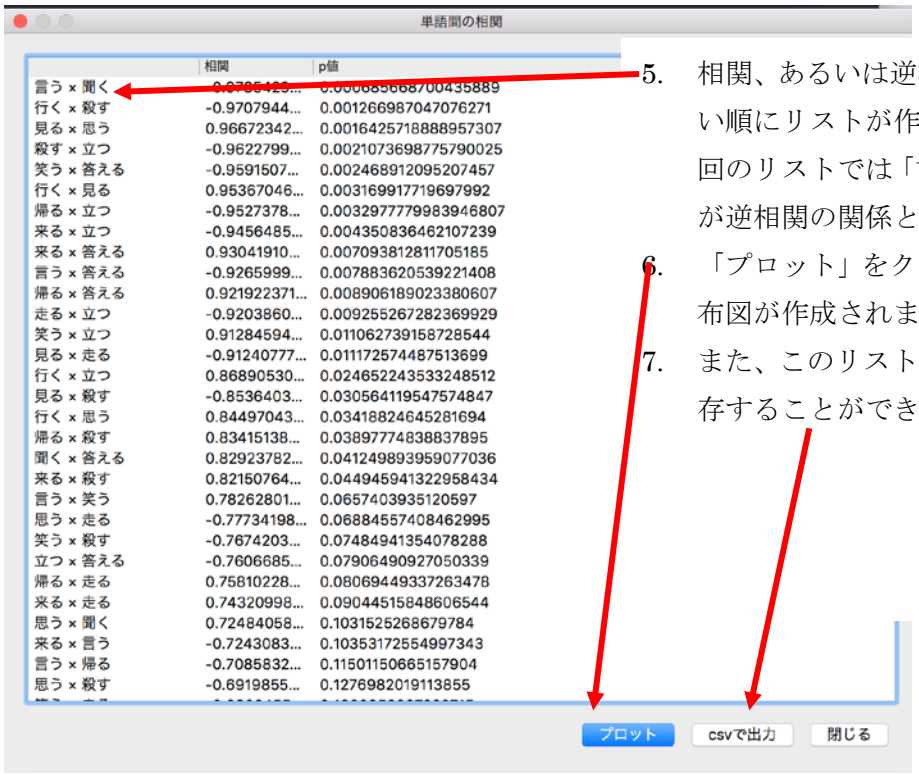
今回もチュートリアルの小説データを用いて説明します。



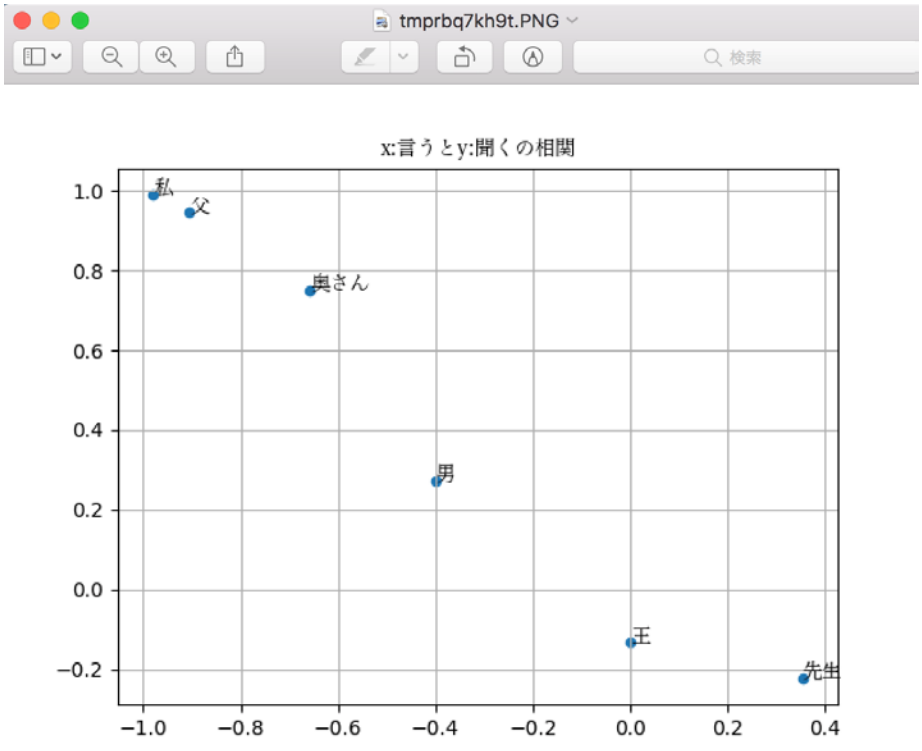
1. 「分析」から「相関分析」、「単語類似度の相関分析」と進んで、クリックしてください。



2. 今回は小説中に登場する人称名詞の特徴を分析してみたいと思いますので、検索をクリックして候補の単語の一覧を取得します。
3. 人称名詞をプロットする単語として選択し、「＞」で「プロットする単語」として登録してください。
4. 人称名詞の特徴となる語としては名詞、形容詞と動詞があると思われますが、今回は人称名詞がどのような動作をする設定で扱われているのかを見たいと思いますので、動詞を候補の単語から選択し「＞」で「軸として使用する単語」として登録してください。



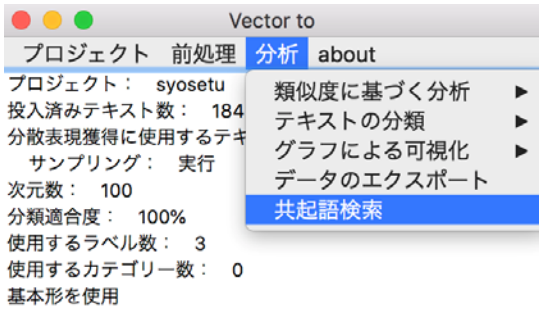
5. 相関、あるいは逆相関の関係の強い順にリストが作成されます。今回のリストでは「言う」と「聞く」が逆相関の関係となっています。
6. 「プロット」をクリックすると散布図が作成されます。
7. また、このリストは csv 形式で保存することができます。



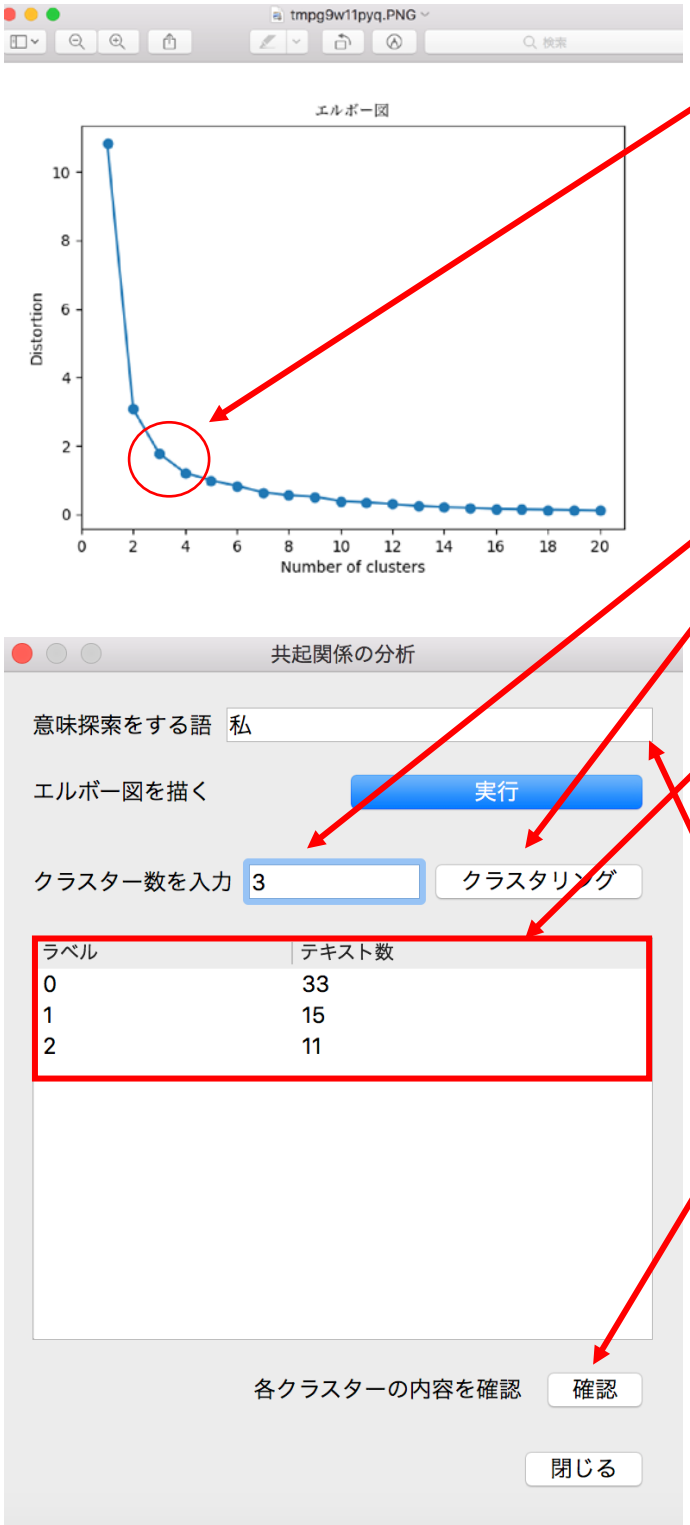
単語の共起関係に注目した分析

テキストマイニングの基本的な考え方の一つに共起関係への注目があります。Vector to では単純な共起関係への注目は行わず、注目語を指定したうえで、その後が出現する文章だけを抽出、K-means 法によって抽出された文章の分類を行い、その分類結果に出現する共起関係を分析します。

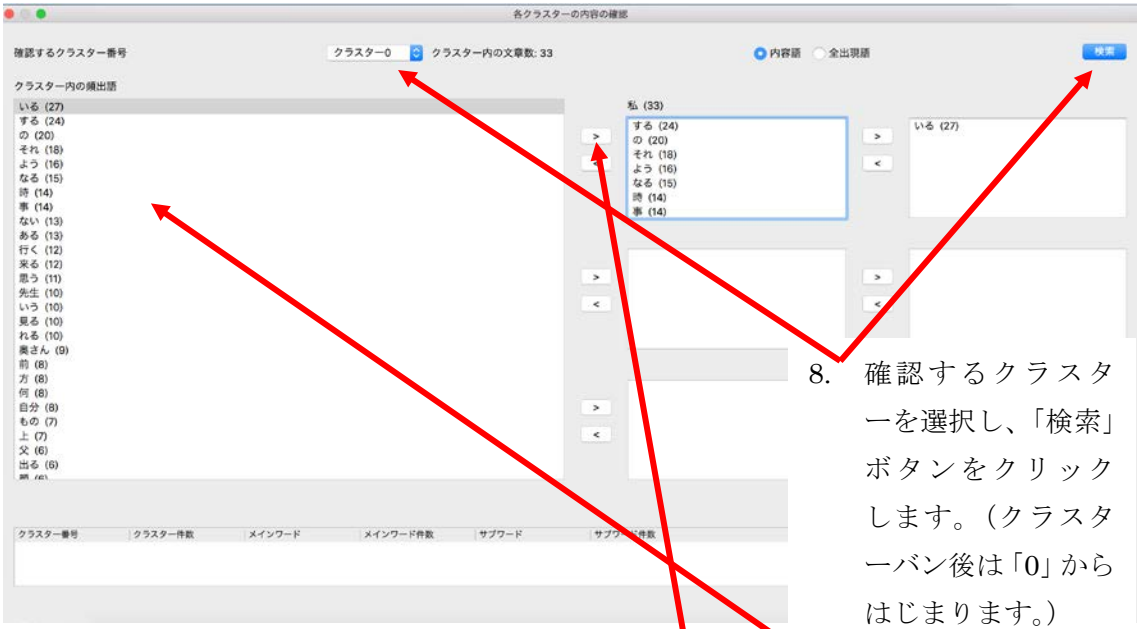
今回はチュートリアルで使用した小説の学習済みモデルを使用します。



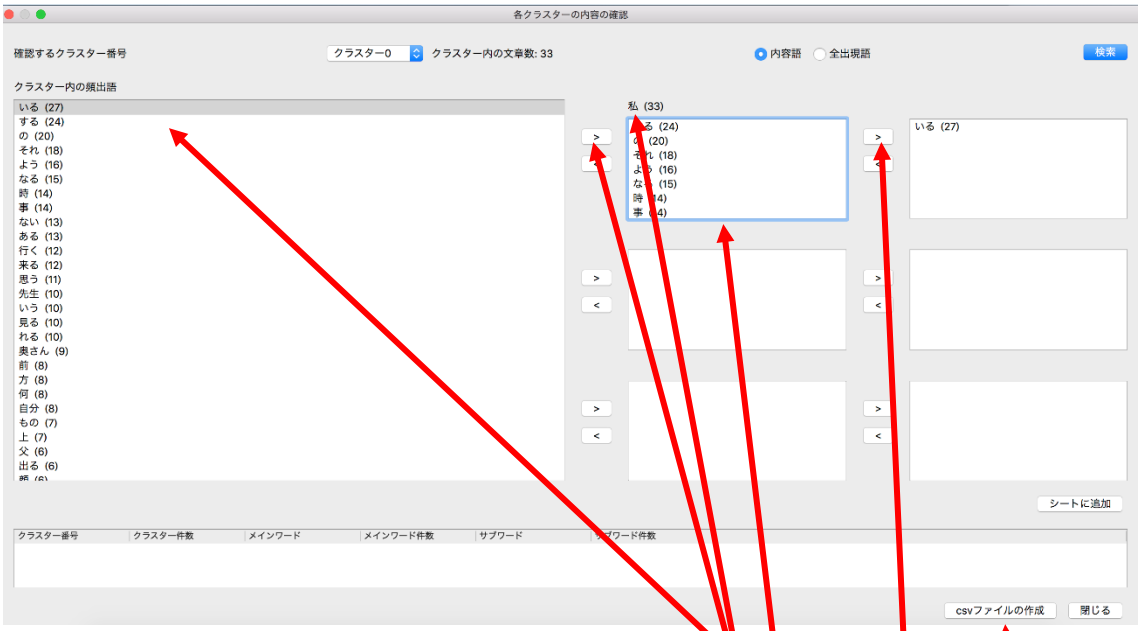
1. 「分析」から「共起語検索」をクリックします。
2. 意味探索したい語を入力します。
3. エルボー図を描くを「実行」すると先ほど指定した語が登場する文章だけがバックグラウンドで抽出され、分類するクラスター数を確認することができます。(エルボー図の作成は必須ではありません。不要な場合は次のステップに進んでください。)



4. エルボー図を確認するとクラスター数は 3 or 4 あたりが良さそうです。
5. 今回はクラスター数を 3 で分析を進めたいと思います。クラスター数を 3 にし、「クラスタリング」をクリックします。
6. それぞれのクラスターに分類された文章数が表示されます。（ラベルが 0 からはじまりますので注意してください。）
7. 「確認」をクリックすると各クラスターに含まれる出現語を確認することができます。ここで確認することができる語は、「意味探索する語」を必ず含んでいますので、「意味探索する語」と共起する語となります。



8. 確認するクラスターを選択し、「検索」ボタンをクリックします。（クラスターバン後は「0」からはじまります。）
9. 選択されたクラスターの中での頻出語が表示されます。カッコ内は出現数です。出現数には総出現回数ではなく、出現テキスト数を用いています。今回は「意味探索する語」として「私」を指定していますので「私」が最も多い語として登場します。
10. 頻出語から単語を選択し「>」ボタンをクリックすると、その頻出語と共起する単語が表示されます。間違って「>」を押した単語は「<」で戻すことができます。



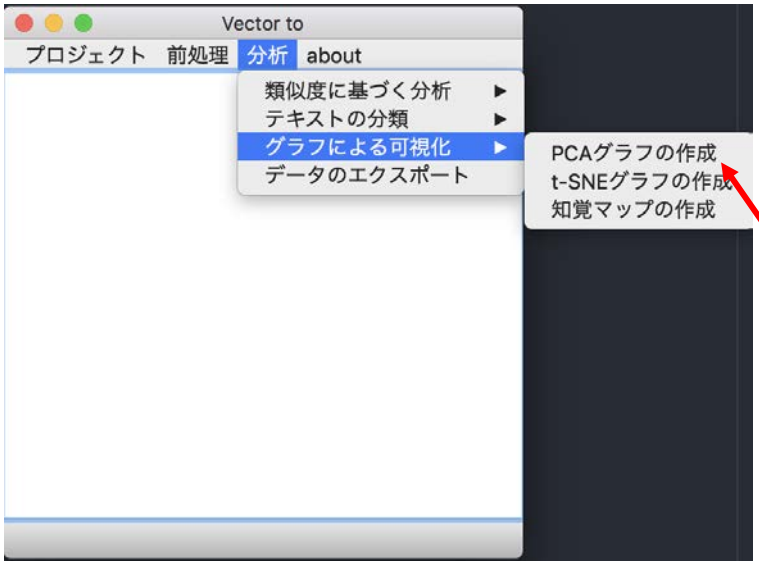
11. 今回は「私」という単語が含まれる文章の特徴を知りたいので「私」を選択し、「>」をクリックします。
12. 頻出語として選ばれた単語(「私」)と共起する語が一覧で表示されます。
13. 「>」をクリックすると、選択された共起語が右欄に移ります。
14. さらに単語を選択し絞り込むことができます。
15. 絞り込みの結果を csv 形式で保存することができます。

グラフによる可視化

Vector to では fastText を使って単語を数次元のベクトル表現に変換しています。これによって単語間の関係を類似度として計算することができるようになっています。しかし、このままでは単語 A と単語 B の関係など少数の単語間の関係しか把握することができず、複数の単語間の関係を俯瞰的に確認することができません。そこで、Vector to では多次元のベクトル表現を 2 次元に削減し、散布図を描く機能を実装しています。

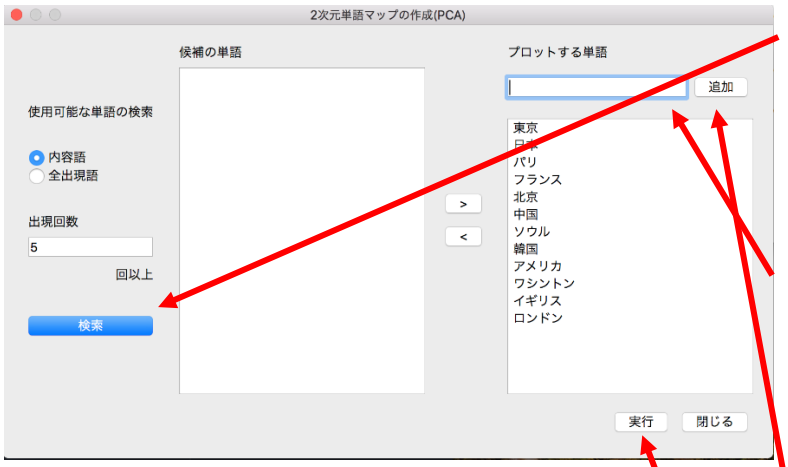
PCA(主成分分析)グラフの作成

PCA(主成分分析)で作成されたグラフの特徴は、単語群 A(King, Queen)と単語群 B(Man, Woman)の間に明確な関係がある場合、またその関係性が分散を最大化するにあたって影響の強いものである場合、単語間のオフセット関係が比較的明確に出現します。たとえば、“King” - “Man” + “Woman”という単語の加減算ができるということは、それぞれの単語間の関係にオフセット関係があるということです。

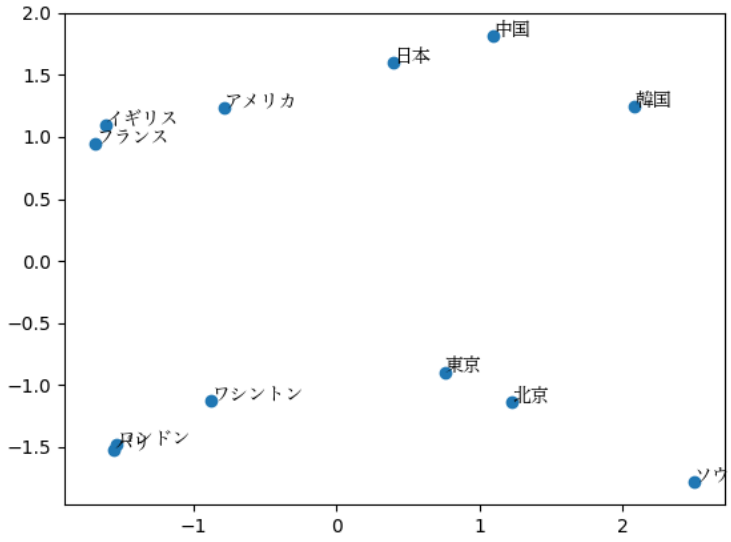


以下では、Vector to ホームページで公開している Wikipedia の学習済みモデル(Vector to-2)を使用しています。

1. メニューバーの「分析」から「グラフによる可視化」→「PCA グラフの作成」を選択してください。



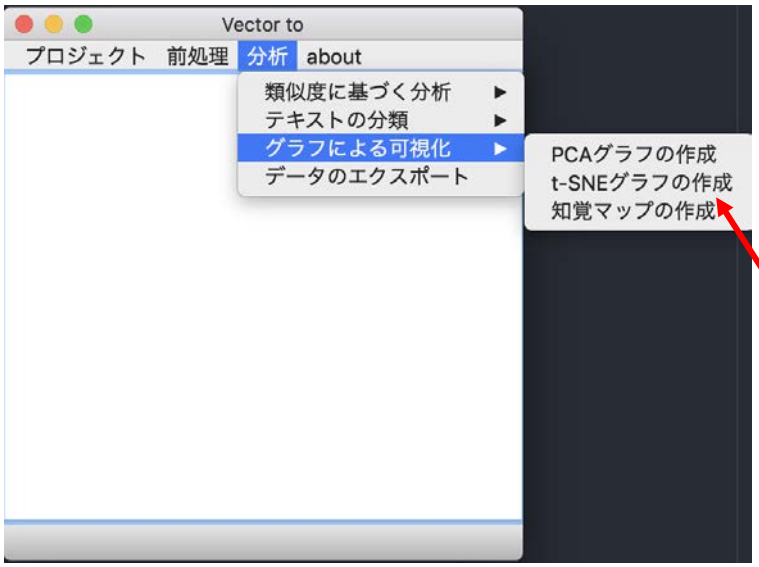
2. ご自身で学習モデルを作成した場合はプロットする単語を検索することができます。
3. また、直接プロットする単語を入力することもできます。複数の単語を入力する際は、スペースで単語間をつないでください。
4. 「追加」をクリックするとプロットする単語の一覧に単語が追加されます。
5. 「実行」をクリックしてください。PCAで次元削減を行った結果が作図されます。
6. このグラフから、国名と首都のオフセット関係が明確にわかります。



t-SNE グラフの作成

t-SNE では、元の多次元の単語間の関係をできる限り残したまま次元の削減を行います。そのため、t-SNE を用いたグラフでは類似度の高いものは近くにプロットされ、低いものは遠くにプロットされます。このような性質から、t-SNE で作成されたグラフには単語間の距離関係に基づくクラスターが形成されます。つまり、t-SNE を用いることで単語間の類似度に基づくクラスターを確認することができます。

しかし、t-SNE には PCA ではなかった数々のパラメーターが存在し、また初期値にランダムに生成された情報を使用しています。そのため、同じ結果を得ることが難しく、またパラメーターをうまく設定しなければ、十分な結果を得ることはできません。



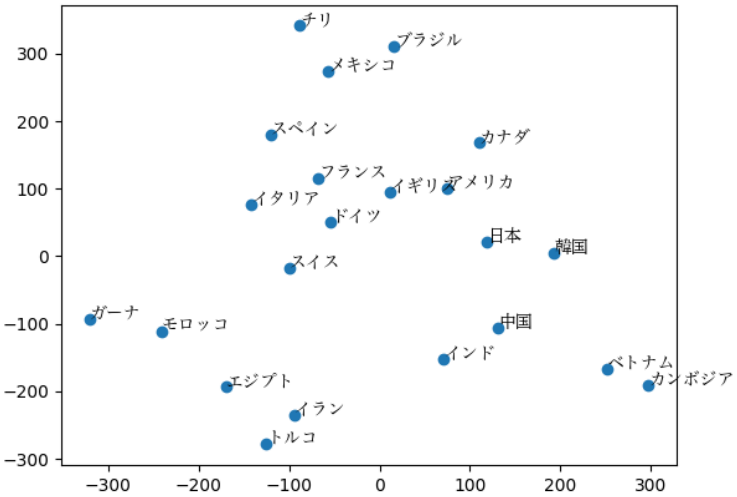
以下では、Vector to ホームページで公開している Wikipedia の学習済みモデル(Vector to-2)を使用しています。

1. メニューバーの「分析」から「グラフによる可視化」→「t-SNE グラフの作成」を選択してください。

※ 初期値にランダムに生成された情報を使用しているため、同じ結果にはなりません。

基本的な使用方法は PCA と同じです。以下では、ハイパーパラメーターの説明を行います。

- perplexity
  - 各点に近接する点の数を推測する指標。プロットする単語の数よりも少なく設定してください。クラスターが明確に分離されない場合は perplexity を高く設定するとうまくいくことがあります。 [推奨値：5-50]
- 学習率
  - 学習ごとにモデルをどの程度更新するのかの設定。大きいほど最適解に早く近づくことができるが、最適解を飛ばしてしまう可能性もある。[推奨値：10-1000]
- 学習回数
  - 学習回数。250 以上に設定してください。
- 高速化
  - barnes_hut 法を用いることで解析速度を高速化します。しかし、barnes_hut 法を用いない(チェック無の)ほうが正確な結果が得られます。
- 次元の削減
  - t-SNE 計算を行う scikit-learn の機能ではなく、Vector to の機能です。
  - PCA を用いてあらかじめ次元を削減し、その後に t-SNE 法でグラフを削減します。あらかじめ次元を削減することで、余計なノイズを削減することができます。



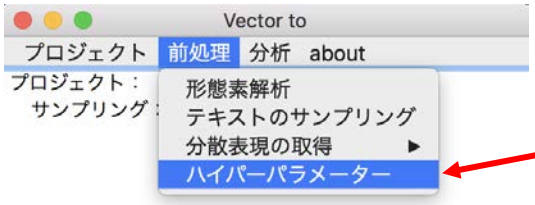
ハイパーパラメーターを設定します。

2. perplexity：5
   学習率：100
   学習回数：2000
   高速化：なし
   次元の削減：0

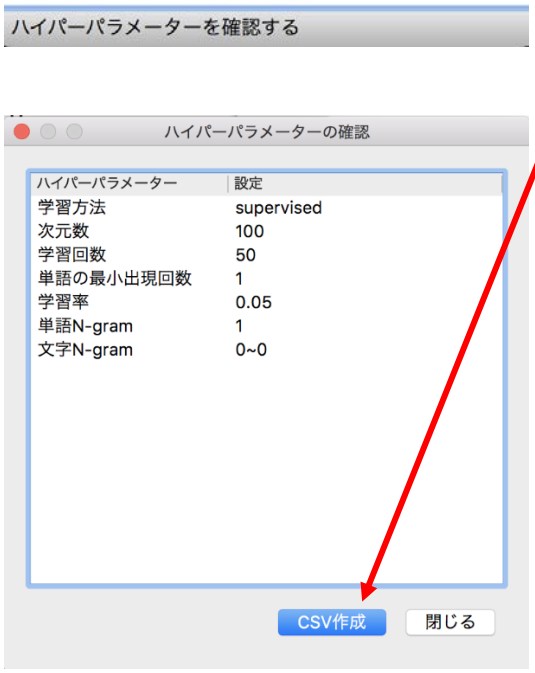
この図から地域的なクラスターが見て取れます。

ハイパーパラメーターの確認

学習に際して設定したハイパーパラメーターを確認することができます。

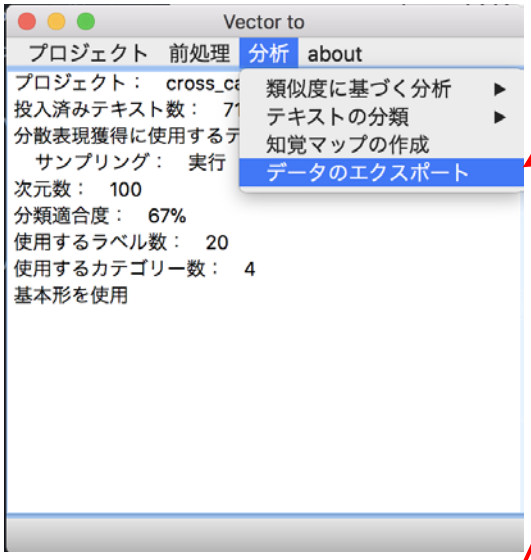


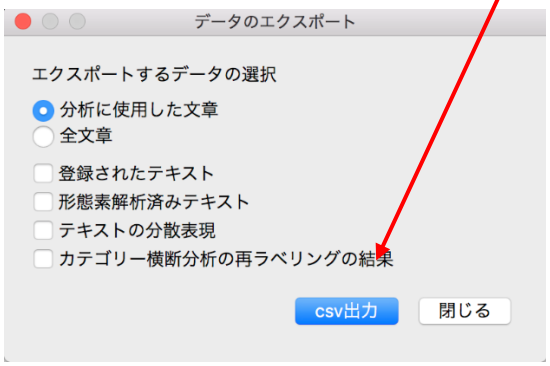
1. メニューバーの「前処理」から「ハイパーパラメーター」をクリック
2. CSV ファイルとしてエクスポートすることができます



データのエクスポート

分析に使用しているテキスト、形態素解析済みのテキスト、分散表現化された文章のベクトル、カテゴリー横断分析で再ラベリングを行った際の各文章の結果を出力することができます。





1. メニューバーの「分析」から「データのエクスポート」を選択します。
2. 「csv 出力」をクリック

- エクスポートするデータは分散表現の獲得に使用した「分析に使用した文章」と「全文章」を選択できます。Vector to では登録した文章をすべて形態素解析していますので、形態素解析の結果を使って他の分析を行う場合にはこちらで「全文章」を選択することで、エクスポートすることができます。
- 「登録されたテキスト」はインポートしたすべての文章です。
- 「形態素解析済みテキスト」は形態素解析時に長音記号の修正などを行っている場合、それらを反映したものです。
- 「テキストの分散表現」は、各文章をベクトル表現にしたものです。
- 「カテゴリー横断分析の再ラベリングの結果」はカテゴリー横断分析で再ラベリングを行った際の結果です。0 から 3 が出力されます。
  0：分析には使用していないテキスト、1：ラベルが一致したもの、2：ラベルは一致しなｋったが、カテゴリーが一致したもの、3：ラベルもカテゴリーも不一致のもの

プロジェクトの削除

不要になったプロジェクトはメニューバーのプロジェクトから削除できます。

ここで削除を行うと、登録しているデータや分析に使用した条件等すべて削除されますが、登録するにあたって使用したファイルは削除されません。

以上が Vector_to の使用方法です。

機械学習を用いたテキストマイニングの分析事例

分析の対象と環境

以下では、www 上のクチコミデータを用いて機械学習を活用したテキストマイニングを行います。使用したデータはじゃらん net に投稿された「テーマパーク・レジャーランド(東京ディズニーランド、東京ディズニーシー、ユニバーサル・スタジオ・ジャパン、横浜・八景島シーパラダイス、ナガシマスパーランド)」、「水族館(沖縄美ら海水族館、鳥羽水族館、鴨川シーワールド、海遊館、名古屋港水族館)」、「動物園・植物園(アドベンチャーワールド、旭川市旭日山動物園、神戸市立王子動物園、東山動植物園、上野動物園)」「アウトレットモール(神戸三田プレミアム・アウトレット、三井アウトレットパーク ジャズドリーム長島、三井アウトレットパーク 木更津、三井アウトレットパーク 滋賀竜王、三井アウトレットパーク マリンピア神戸)」の 4 つのカテゴリーのランキング上位 5 施設、計 20 施設に関するクチコミ計 71218 件です。じゃらん net に投稿されたクチコミ情報は www 上に投稿されている各施設に関するクチコミの一部でしかなく、本来であれば www 上に投稿されているすべての情報を分析する必要がありますが竹岡・高木(2017)に従い、今回はじゃらん net の情報だけを分析の対象としています。データの収集は 2017 年 10 月 9 日から同 26 日にかけて行いました。実際に分析したデータは、各施設のクチコミ数に偏りがあり、このような不均衡データを用いたクラス分類の結果には偏りが生じることが考えられることから、各施設のクチコミから 1000 件をサンプリングし、計 20000 件で行っています。

クチコミデータに関してはカタカナを全角に、アルファベットと数字を半角に変換、当該施設名に関わるものは、分析の中心的単語であることを考慮し表記ゆれの修正といった前処理を行っています。fastText はすべてデフォルト設定で使用しています。

分散表現(distributed representation)の分散(vaiance)に基づく知覚マップの作製 [19]

分散表現を用いれば、意味の類似する単語を、コサイン類似度を算出することで抽出することができます。たとえば今回使用しているデータの中で東京ディズニーランドといくつかの施設のコサイン類似度を算出すると、東京ディズニーシーが 0.73436、ユニバーサル・スタジオ・ジャパンが 0.41962、三井アウトレットパーク木更津が-0.58749 です。

これを応用すれば各施設を構成する要素とその構成の程度を近似した値として使用することができます。つまり、単語は様々な成分、たとえば、東京ディズニーランドの場合はパレードやアトラクションなどの成分からなり、それぞれが東京ディズニーランドという単

[19] 分析事例は Vector to の α 版とも呼べる CUI 環境で行ったものです。環境の詳細は次の通りです。OS：Ubuntu 16.04.3 LTS、形態素解析：mecab 0.996、辞書：mecab-ipadic-neologd 2.7.0、fasttext：“431c9e2a9b5149369cc60fb9f5beba58dcf8ca17”、プログラミング言語：python 3.5.2、使用モジュール：pyfasttext 0.4.4 (fastText のモデルに接続)、scikit-learn 0.19.1(k-means によるクラスタリングに使用)。

語に対して影響を与えることで、東京ディズニーランドという単語の意味が構成されていることから、類似度の高い単語はこれらの成分と影響の程度が似たものであるといえます。

たとえば、「東京ディズニーランド」と「パレード」の類似度は 0.994496、「アトラクション」の類似度は 0.923629 であり、「ショッピング」の類似度は-0.58031 です。この結果から東京ディズニーランドはパレードやアトラクションがその特徴として消費者に認識されており、ショッピングを行うところとは認識されていないことが推測されます。このコサイン類似度を基に作成した知覚マップが図 1 です。

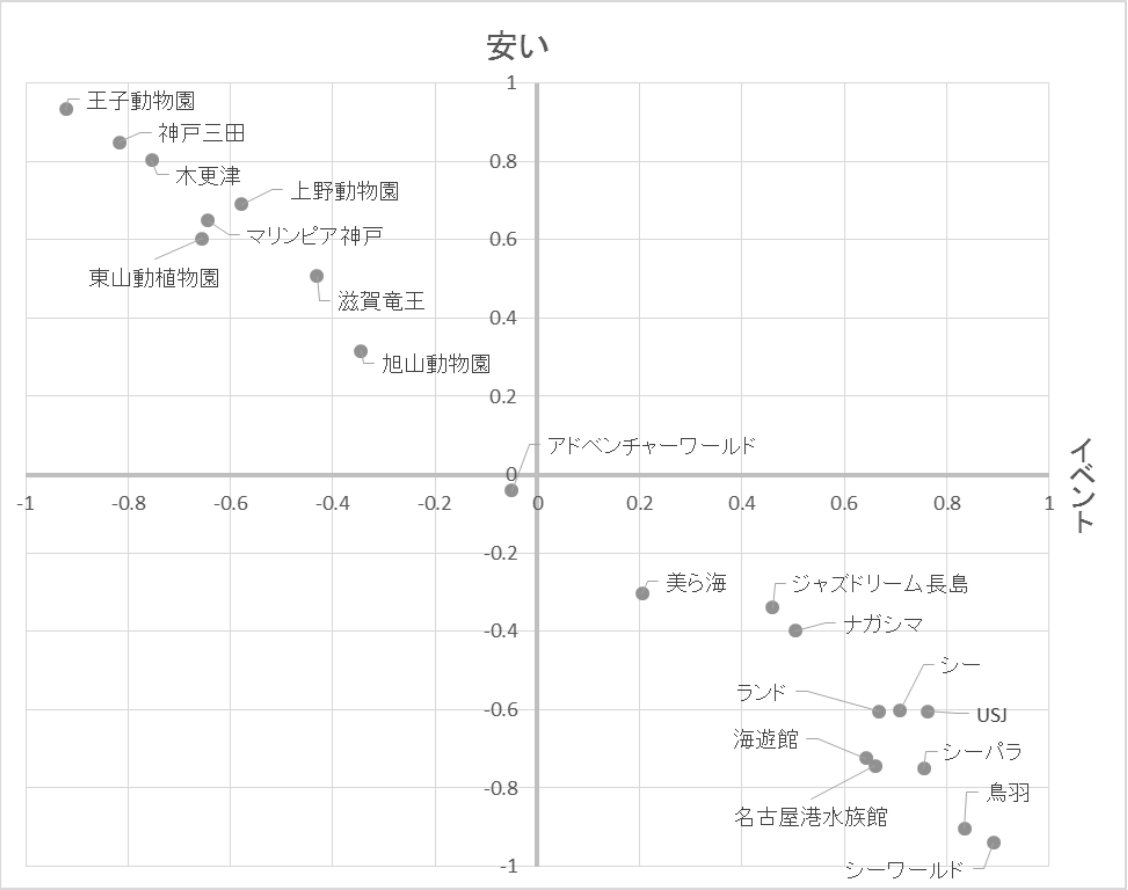


図 1　「イベント」と「安い」を 2 軸にとった知覚マップ [20]

この知覚マップの作成に当たっては、クチコミ中の名詞、動詞、形容詞を合わせた上位頻出 200 語を対象に、それらの語と各施設名のコサイン類似度を算出、それらの分散を計算し、分散の大きなもの(「安い」の分散は 0.425845421、「イベント」の分散は 0.416727697)を二つの軸にとって各施設をプロットしています。たとえば、グラフ中左上にある王子動物園は「神戸市立王子動物園」と「安い」のコサイン類似度が 0.933182063 と高く、「イベント」とのコサイン類似度は-0.92163741 と低いことをあらわしています。

[20] 施設名には略語を使用しています。

図 2 は図 1 と同様に分散の大きい「お土産(分散 : 0.377282719)」と「楽しかっ[21] (分散 : 0.417025259)」を 2 軸にとり、それら 2 単語と各施設名の類似度をもとに、各施設名をプロットしたものです。このグラフからショッピングモールの多くはお土産との類似度が低いことや、「お土産」との類似度が高い施設ほど「楽しかっ」との類似度が高いことがわかります。

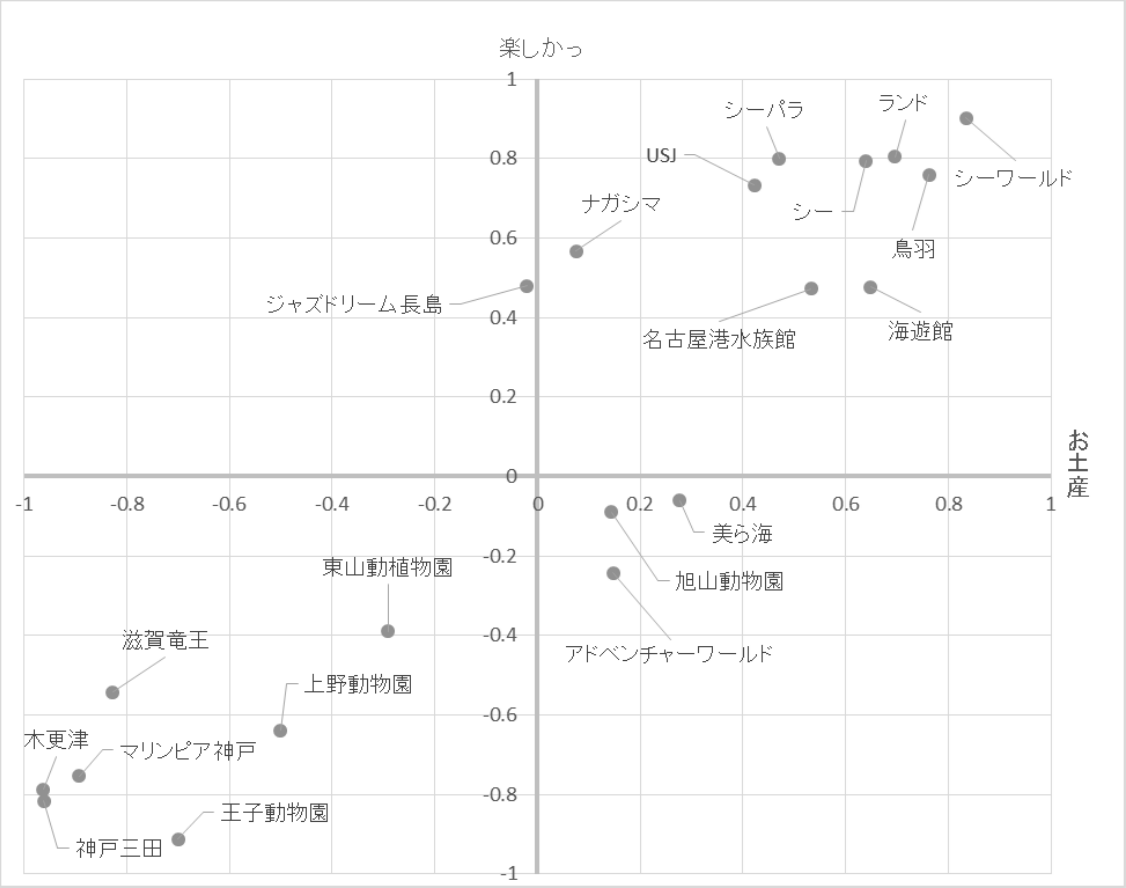


図 2　「お土産」と「楽しかっ」を 2 軸にとった知覚マップ

このように、分散表現とそれに基づく類似度を用いれば知覚マップを容易に作成するこ

[21] 今回の分析の中では「見」や「行っ」のように活用語尾の無い動詞が登場しています。例えば計量テキスト分析用ソフトウェアとして多方面で活用されている KH Coder の場合にはこのような語は基本形に直したうえで分析に使用しているので、これらの語は「見る」と「行く」として集計されていると考えられますが、今回の分析ではその後の分析に使用することなども考慮し、基本形には変換せず使用しています。そのため、同じ「見る」という単語であっても「見」、「見る」、「見れ」といった形で登場し、異なる語としてカウントされています。なお、形態素解析に使用した Mecab の辞書には基本形が登録されており、形態素解析を行う際に簡単に変換することもできますし、Vector to では前処理の段階でこの事例の用に表層形を使用するのか、それとも基本形を使用するのかを選ぶことができます。

とが可能です。また、アンケートによるデータ収集のように最初に決められた概念に関する集計結果を用いて作成するのとは異なり、何らかの形で単語がクチコミなどのデータ内に出現していれば、このような形で知覚マップを作成することが可能です。

また、今回は知覚マップ中に広く分布するように分散の大きな単語を用いましたが、極論すれば、データ中に登場するすべての単語をこのような形で分析することも可能であり、これまでと比較して、容易に素早く消費者の観点から見た分析対象の市場におけるポジショニングを確認することができます。

文章分類に基づくカテゴリー横断分析

次に、自然言語を対象とした機械学習の中でも広く利用されているテキスト分類の機能と、これまでの計量テキスト分析の手法を用いて各施設に共通して言及されている特徴を可視化したいと思います。

機械学習におけるテキストの分類は教師あり学習で行われることが多く、各テキストにラベルを張り、それをもとにテキストの特徴を抽出、学習するというプロセスがとられます。そして、このプロセスで学習された結果をもとにして、新たにテキストが投入された場合には、そのテキストがどのラベルに当てはまるのかを判定し、分類することが可能になります。通常はこのように用いられるのですが、ここでは学習に使用したデータを再度 fastText に投入し、分類器によって判定、その中で分類器が誤判定したラベルから、各施設に共通している特徴について分析を行いたいと思います。

表 1　分類器による再ラベリングの結果

	一致	同一カテゴリー	不一致
ディズニーランド	7	894	99
ディズニーシー		930	70
ユニバーサル・スタジオ・ジャパン	837	99	64
横浜八景島シーパラダイス	631	262	107
ナガシマスパーランド	401	505	94
沖縄美ら海水族館		52	948
鳥羽水族館	7	14	979
鴨川シーワールド	44	10	946
海遊館		19	981
名古屋港水族館	8	12	980
アドベンチャーワールド	115	660	225
旭川市旭山動物園	727	170	103
神戸市立王子動物園	174	718	108
東山動植物園	539	296	165
上野動物園		882	118
神戸三田プレミアムアウトレット		976	24
三井アウトレットパークジャズドリーム長島		927	73
三井アウトレットパーク木更津		965	35
三井アウトレットパーク滋賀竜王	966		34
三井アウトレットパークマリンピア神戸		955	45

表 3 では、分類器の判定結果が元のラベルと一致する場合には一致、ラベルは一致しなかったが、同一カテゴリーの施設のラベルが張られた場合には同一カテゴリー、カテゴリーも一致しなかった場合には不一致として、それらの結果を集計しています。

表 3 の分類器による再ラベリングの結果から、「ユニバーサル・スタジオ・ジャパン」や「三井アウトレットパーク滋賀竜王」のように、分類器がかなりの程度判定に成功しているクチコミもあれば、「沖縄美ら海水族館」、「鳥羽水族館」、「鴨川シーワールド」、「海遊館」、「名古屋港水族館」の各水族館のように、他のカテゴリーの施設と判定されているものもあります。このような結果からは、「ユニバーサル・スタジオ・ジャパン」や「三井アウトレットパーク滋賀竜王」の場合には、他の施設と比較して何らかの特徴的なものが存在し、それが分類器の判定を正確なものとしていることが推測されます。

以下では不一致カテゴリーに注目します。このカテゴリーに含まれているクチコミは先述のとおり分類器によって再ラベリングした際に、異なるカテゴリーのサービス施設名がラベリングされたクチコミです。このようなクチコミに注目する理由としては、このようなクチコミにはサービスの業態を超えた共通点があると考えられるからです。つまり、ラベルが一致したクチコミを分析すればその施設の特徴が、同一カテゴリーに含まれたクチコミを分析すればそのサービスカテゴリーの特徴が、そして不一致カテゴリーを分析すれば今回の分析対象となった 4 つのサービスカテゴリーに共通する特徴が抽出できると考えられるからです。

しかし、このラベルが不一致のクチコミは全体で 6198 件存在します。これらを直接分析することは困難です。そこで、これらのクチコミを共通の特徴に沿って複数のクラスターに分割し、それを分析することで、消費者の意見を可視化したいと思います。先ほどの fastText を用いたクラスタリングでは教師データを用いてクラスタリングを行いましたが、今回はこれらのラベル(施設名)を用いても意味はありません。なぜなら、クチコミを施設ごとに分類するのが目的ではなく、クチコミ内に登場する何らかの特徴に基づいて分類することが目的となるからです。そのため、教師データのない状態でクラスタリングを行う必要があります。

今回は教師データなしでのクラスタリングに k-means 法を用います。k-means 法では fastText が算出した文章の分散表現を用いてクラスタリングを行いました。図 3 のエルボー図より、クラスター数は 8 から 15 が適正と推測されますので、今回は全クラスター内のクチコミ数が 1000 以下となる分岐点、11 クラスターに分類するこにします。

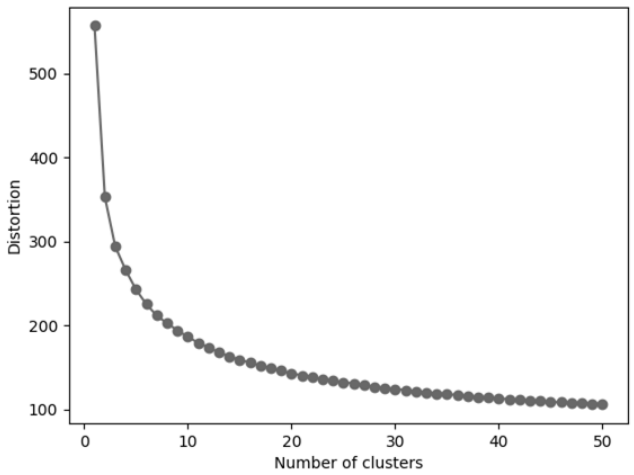


図 3　エルボー図

ここまでで共通話題のクラスタリングはできましたが、この各クラスターの中でどのような内容が書き込まれているのかを分析する必要があります。しかし、fastText では文章を分散表現として要約してはいますが、分散表現は今回の分析の場合 100 次元の座標で表現されており、人の目でこれら座標から中身を判断することは困難です。そこで、以下では計量テキスト分析の手法を用いて、各クラスターの中でどのような話題が投稿されているのかを見ていくことにします。

図 4 は各クラスターの頻出語(名詞、動詞 、形容詞)と出現回数を示しています。図中の各表の左上にはクラスター名を、右上には各クラスターに分類されたクチコミ件数を、第 1 列は頻出語上位 3 語を、第 2 列にはその出現回数を、第 3 列には各語との共起頻度の高い上位 3 語を、そして第 4 列には共起語の共起数を記しています。たとえばクラスター0 の出現回数 1 位は「ショー」で全 655 件のクチコミ中 420 件で登場しており、この「ショー」と共起する形で「見」が 153 件登場しています。

クラスター0　655 件

ショー	420	見	153
		イルカ	142
		シャチ	138
水族館	235	ショー	112
		見	75
		行っ	60
イルカ	199	ショー	142
		シャチ	88
		見	77

クラスター1　594 件

水槽	446	ジンベイザメ	182
		大きな	140
		見	125
ジンベイザメ	238	水槽	182
		大きな	67
		見	62
水族館	164	水槽	112
		見	55
		ジンベイザメ	49

クラスター2　115 件

パンダ	84	行っ	16
		弁当	14
		平日	14
動物	26	見	6
		パンダ	5
		思い	5
		行っ	5
弁当	18	パンダ	14
		食べる	9
		持っ	8

クラスター3　267 件

ショー	246	シャチ	164
		イルカ	129
		水族館	108
シャチ	178	ショー	164
		イルカ	101
		水族館	79
イルカ	137	ショー	129
		シャチ	101
		水族館	59

クラスター4　361 件

水族館	249	水槽	141
		ジンベイザメ	79
		魚	58
水槽	240	水族館	141
		ジンベイザメ	87
		魚	76
ジンベイザメ	118	水槽	87
		水族館	79
		迫力	39

クラスター5　658 件

水族館	346	思い	76
		行っ	69
		行き	68
ショー	136	水族館	34
		思い	31
		イルカ	31
楽しめ	126	水族館	59
		イルカショー	28
		ショー	27

クラスター6　657 件

思い	141	行き	33
		行っ	33
		見	27
見	131	時間	29
		思い	27
		行っ	27
行っ	124	思い	33
		見	27
		行き	24

クラスター7　895 件

見	280	ショー	100
		行っ	87
		行き	80
行っ	253	ショー	78
		行き	70
		思い	68
ショー	246	見	100
		思い	81
		行っ	78

クラスター8　478 件

駐車場	91	行く	17
		車	16
		無料	16
思い	75	楽しめる	15
		駐車場	14
		良い	14
行き	69	思い	12
		駐車場	11
		アウトレット	11

クラスター9　587 件

ショー	482	シャチ	219
		イルカ	168
		水族館	141
シャチ	270	ショー	219
		イルカ	88
		見	85
水族館	207	ショー	141
		シャチ	56
		行っ	54

クラスター10　931 件

行き	278	行っ	62
		思い	61
		子供	60
行っ	232	行き	62
		子供	40
		思い	40
子供	161	行き	60
		行っ	40
		楽しめる	40

図 4　各クラスターの頻出語と共起語 [22]

この結果から、単語の出現数には偏りがあること、つまり各クラスターの特徴は抽出されていることがわかります。たとえば、クラスター1 は大きな水槽について、クラスター2 はパンダについて、クラスター0 や 5、9 は水族館でのショーについて、クラスター8 は駐車

[22] 図 4 を見るとクラスター0、1、3、4、5、9 は人であれば水族館に分類するような単語の出現の傾向であり、11 クラスター中の半数の 6 クラスターがこれにあたります。表 3 では水族館の各施設が不一致カテゴリーに分類される件数が多かったですが、その結果の偏りがこのようなクラスターの偏りを生んだと考えられます。このように fastText が適切に文章を分類できなかった理由としては、ラベル数が 20 と多いこと、またかなり類似する施設を分類していること、そして最大の理由としてデフォルト設定で使用したことがあげられます。デフォルト設定で使用する場合でも、たとえば、「良い」と「悪い」という評価を学習させて、それをもとに分類させれば、そこで出現する単語の分布の違いは明確であり、今回とは異なる結果になると考えらます。また、水族館という施設が他のサービス施設と類似するサービスを提供しており、それへの言及への多さからこのような結果になったとも考えられます。

場について、クラスター10 は子供を連れての行き先として言及されていることが推測されます。

しかし、これらの特徴はそのクラスターの最大公約数的な特徴であり、細かな特徴をつかみきることはできていません。たとえば、クラスター7 では、車いすに関する言及が 5 回、ベビーカーに関する言及が 30 回、アレルギーに関する言及が 2 回あり、またクラスター10 では観覧車やサンタマリア、明石海峡、繁華街など近隣施設への言及と思われる単語が複数登場していました。これらはそれぞれのクチコミを人の目で確認することで抽出可能な特徴といえます。

類推問題を用いた特徴分析 [23][24]

ここで扱うデータは、「じゃらん net」に掲載されているクチコミ東京ディズニーランド、東京ディズニーシー、ユニバーサル・スタジオ・ジャパン、横浜・八景島シーパラダイス、ナガシマスパーランドという 5 つのテーマパークに関するクチコミです。実際に分析したデータは、各施設のクチコミ数に偏りがあり、このような不均衡データを用いた学習の結果には偏りが生じることが考えられることから、各施設のクチコミから 1500 件をサンプリングした計 7500 件です。

まず、単純に単語間の類似度をもとに 5 つの施設の類似度を計算すると、各施設の関係は下記表 3 のようになります。表 3 からは、「ディズニーランド」と「ディズニーシー」の類似度が高いこと、すなわちクチコミの中で類似する意味を持っていることが考えられます。言い換えると、多数いるクチコミ作成者の中では、同じテーマパークであっても、「ディズニーランド」と「ディズニーシー」以外は比較的遠いものとして認識されていることが考えられます。

表 3　施設間の類似度

	ディズニーランド	ディズニーシー	USJ	横浜・八景島シーパラダイス	ナガシマスパーランド
ディズニーランド		0.72463	0.31852	0.17208	0.29442
ディズニーシー			0.26664	0.22879	0.24817
USJ				0.20728	0.25325
横浜・八景島シーパラダイス					0.25377
ナガシマスパーランド					

当然ながら、クチコミの中には施設名以外にも無数の単語が含まれています。以下では、この無数に含まれる単語から類似度の高い単語を抽出し、各施設の特徴を明らかにします。今回は、出現回数上位 200 位までの単語を総当たりで計算し、各施設と類似度の高い単語を抽出しました。その結果が、表 4 です。この結果から、「USJ」と「大阪」、「横浜・八景島シーパラダイス」と「水族館」、「ナガシマスパーランド」と「アウトレットモール」など、単語を入れ替えても意味が通りそうなものが上位にきていることがわかります。すなわち、これらの単語がテキスト作成者の中では代替可能な言葉、類似する意味を持つ言葉として認識されていると考えることができます。

表 4　類似度に基づく特徴の抽出

ディズニーランド	
ディズニーシー	0.724639
雰囲気	0.612989
比べる	0.609007
飲める	0.588506
お酒	0.580588

ディズニーシー	
ディズニーランド	0.724639
お酒	0.651191
飲める	0.645726
雰囲気	0.530008
大人	0.514238

USJ	
年間パスポート	0.545035
Potter	0.513558
Harry	0.509493
大阪	0.5051448
行く	0.473982

横浜・八景島シーパラダイス	
水族館	0.641214
鎌倉	0.605503
シロイルカ	0.561197
金沢八景	0.519127
イルカ	516494

ナガシマスパーランド	
温泉	0.544516
ナガシマ	0.538653
地区	0.531
西	0.518665
アウトレットモール	0.513082

23 ここでのハイパーパラメーターは学習方法：skip-gram、次元数：100、学習回数：30、学習率：0.05、文字 N-gram：2~4 です。

24 ここでの分析結果は竹岡(2018b)および高木・竹岡(2018)で発表されたものを再掲したものです。

この他にも、単語の加減算から各施設の特徴を検討しているのが、表 5 と表 6 です。表 5 は、各施設名からどのような単語を減算すると最も特徴が遠ざかるのかについて、出現回数上位 200 位までの単語で総当たりによって計算を行っています。「ディズニーシー」は「お酒」と「飲める」を減算すると、逆ベクトルの単語になり、「USJ」は「ハリーポッター」がなくなると、逆ベクトルの単語になっていることがわかります。すなわち、テキスト作成者にとって、「お酒が飲めること」こそが「ディズニーシー」の中核的な意味であり、「ハリーポッター」こそが「USJ」の中核的な意味であると考えられます。

表 5　減算による特徴抽出

ディズニーランド	
ディズニーシー, 飲める	−0.422241
ディズニーシー, お酒,	−0.412928
ディズニーシー, 雰囲気,	−0.395229
ディズニーシー, 違う,	−0.36092
夢の国, ディズニーシー	−0.335172

ディズニーシー	
お酒, 飲める	−0.334084
お酒, ディズニーランド	−0.326443
飲める, ディズニーランド	−0.319303
お酒, クリスマス	−0.20991
飲める, クリスマス	−0.209498

横浜・八景島シーパラダイス	
水族館, イルカ	0.0241484
水族館, 海	0.0308357
イルカ, 海	0.0367161
ここ, イルカ	0.119278
ここ, 海	0.1281773

ナガシマスパーランド	
アウトレット, スチール	−0.019824
スチール, ドラゴン	−0.016169
アウトレット, ドラゴン	0.0190883
プール, スチール	0.045261
ジェットコースター, スチール	0.050169

USJ	
Potter, Harry	−0.168214
ハロウィン, Harry	−0.161839
Potter, ハロウィン	−0.160334
訪れる, Harry	−0.090942
Potter, 訪れる	−0.087906

表 6 は、各施設に何を追加すれば、「ディズニーランド」に近似するのかを、出現回数上位 200 位までの単語で総当たりによって計算を行っている。その結果、雰囲気などの言葉が多いことから、テキスト作成者が認識するディズニーランドの最も特徴的な部分であり、他の施設と最も違う部分は、物的なものではなく雰囲気という非常に曖昧なものではないかと考えることができる。

表 6　特徴の近似化

ディズニーシー	
ディズニーシー + 思う + 雰囲気	0.7940271
ディズニーシー + 思う + 違う	0.7886179
ディズニーシー + 大人 + ディズニー	0.7877395
ディズニーシー + ディズニー + 違う	0.7877291
ディズニーシー + ディズニー + 雰囲気	0.7876899

USJ	
USJ + 雰囲気 + ディズニーシー	0.7415832
USJ + 大人 + ディズニーシー	0.7340913
USJ + ディズニーシー + 飲める	0.7241877
USJ + 違う + ディズニーシー	0.7241855
USJ + 気 + ディズニーシー	0.7176755

横浜・八景島シーパラダイス	
横浜・八景島シーパラダイス + ディズニーシー + 雰囲気	0.6634625
横浜・八景島シーパラダイス + ディズニーシー + 違う	0.6515939
横浜・八景島シーパラダイス + ディズニー + ディズニーシー	0.649789
横浜・八景島シーパラダイス + ディズニーシー + 飲める	0.6444271
横浜・八景島シーパラダイス + ディズニーシー + お酒	0.6360012

ナガシマスパーランド	
ナガシマスパーランド + ディズニーシー + 雰囲気	0.7373017
ナガシマスパーランド + ディズニーシー + お酒	0.725242
ナガシマスパーランド + ディズニーシー + 飲める	0.7249891
ナガシマスパーランド + ディズニーシー + 違う	0.7165409
ナガシマスパーランド + 夢の国 + ディズニーシー	0.7108018

以上で示した分析結果は驚くべきものでは決して無いものです。むしろ、当然の結果であり、この結果自体には何ら面白みも新しさもないでしょう。しかし、裏を返せば、テキストデータを Vector to に投入するだけで、かなりの程度現実をうまく写像した結果が得られるということがわかります。さらには、このような結果が得られる過程においては、分析者の意図は介在せず、にもかかわらず、分析結果を得ることができるというプロセスです。

類推問題を解くという分析は、分散表現を用いたテキストマイニングにおいては最も単純なものです。にもかかわらず、これだけの結果を得られるという点が、分散表現テキストマイニングの強みかもしれません。

外形的データを併用した分析 [25][26]

ここでは、水族館 5 施設(沖縄美ら海水族館、鳥羽水族館、鴨川シーワールド、海遊館、名古屋港水族館)に分析対象をしぼり、これまでの分散表現に加えて外形的データ(仕様など)を用いた、つまり単語の類似度と外形的データを併用した分析をおこないます。これによって、消費者の体験によって構築される意味と、客観的仕様から各施設の特徴を可視化したうえで文えせ記することが可能になります。外形的データとしては Wikipedia に掲載されている水族館の来場者数、飼育種類数、延べ床面積を利用しました [27]。

具体的な方法としては、水族館に関するクチコミ 10000 件の中で上位出現回数 200 位までの単語を抽出(表 7、8、9 中最左列、3 行目以下)、それらと上記 5 施設名の類似度を計算します(同右側の 5 列、3 行目以下)。このように類似度を計算することで各施設の特徴を表す代理変数として扱うことが可能になります。

続いてこの計算された類似度と、各施設の外形的データ(同右側の 5 列、2 行目)の相関係数(同左 2 列目、3 行目以下)を総当たりで計算します。表 7～9 は相関の強い 20 概念を抽出したものです 。

表 7　延べ床面積との相関

	相関	p値	分散	海遊館	沖縄美ら海水族館	鴨川シーワールド	鳥羽水族館	名古屋港水族館
延べ床面積(㎡)				31044	19199	22699	24981	41529
旅行	*-0.993*	0.001	0.008	0.298	0.414	0.366	0.371	0.180
楽しむ	*-0.984*	0.002	0.005	0.179	0.256	0.235	0.194	0.072
一緒	0.935	0.020	0.006	0.231	0.139	0.127	0.121	0.291
利用	-0.909	0.032	0.005	0.191	0.277	0.181	0.212	0.084
前	-0.905	0.035	0.003	0.184	0.204	0.196	0.162	0.062
有名	-0.904	0.035	0.002	0.330	0.397	0.347	0.328	0.284
最高	*-0.903*	0.035	0.013	0.148	0.294	0.368	0.246	0.079
くださる	-0.903	0.036	0.014	-0.022	0.231	0.183	0.157	-0.020
小さい	0.900	0.037	0.005	0.228	0.151	0.137	0.243	0.318
家族	*0.886*	0.045	0.004	0.367	0.236	0.328	0.298	0.395
お勧め	-0.845	0.072	0.012	0.017	0.274	0.217	0.159	0.052
過ごせる	0.843	0.073	0.010	0.234	0.030	0.015	0.114	0.215
良い	-0.829	0.083	0.005	0.269	0.262	0.328	0.266	0.141
以上	-0.817	0.091	0.005	0.258	0.247	0.240	0.270	0.105
凄い	-0.813	0.094	0.005	0.231	0.314	0.190	0.271	0.123
カップル	*0.796*	0.107	0.025	0.433	0.096	0.076	0.180	0.352
目	-0.774	0.124	0.008	0.160	0.288	0.109	0.158	0.042
言う	-0.742	0.151	0.004	0.299	0.347	0.411	0.287	0.252
デート	0.712	0.178	0.020	0.448	0.234	0.147	0.133	0.390
写真	-0.711	0.178	0.002	0.179	0.206	0.130	0.171	0.098

表 7 より、代理変数を用いて算出した各施設の特徴と各施設の延べ床面積から以下のこ

[25] ここでのハイパーパラメーターは学習方法：skip-gram、次元数：100、学習回数：30、学習率：0.05、文字 N-gram：2~4 です。

[26] ここでの分析結果は竹岡(2018c)および竹岡(2019)で発表されたものを再掲したものです。

[27] https://ja.wikipedia.org/wiki/日本の水族館（最終確認日：2018 年 9 月 28 日)。名古屋港水族館の飼育種類数については https://ja.wikipedia.org/wiki/名古屋港水族館(最終確認日：2018 年 9 月 28 日)

とがいえる。

- 「旅行」と延べ床面積は逆相関の関係にある（旅行と延べ床面積の相関係数は-0.993（以下同じ））
  - → 広い水族館は旅行で行くところではない
  - → あるいは、大都市内の水族館の延べ床面積が広い傾向にあるためか
- 「楽しむ」や「最高」と延べ床面積は逆相関の関係にある（楽しむ：-0.984、最高：-0.903）
  - → 広いと楽しめないということか
- 「家族」や「カップル」と延べ床面積は相関関係にある（家族：0.886、カップル：0.796）
  - → 大都市内の水族館なので家族やカップルには手軽ということか

表 8　飼育種類数との相関

	相関	p値	分散	海遊館	沖縄美ら海水族館	鴨川シーワールド	鳥羽水族館	名古屋港水族館
飼育種類数（種）				580	740	800	1200	500
面白い	*0.979*	0.004	0.007	0.065	0.105	0.162	0.244	0.028
ジュゴン	*0.953*	0.012	0.033	0.231	0.236	0.262	0.627	0.171
暑い	−0.948	0.014	0.003	0.201	0.215	0.189	0.091	0.233
見学	0.924	0.025	0.002	0.078	0.140	0.153	0.189	0.096
笑	0.905	0.035	0.003	0.176	0.186	0.159	0.258	0.118
みる	0.899	0.038	0.006	0.177	0.293	0.228	0.368	0.206
すぎる	0.898	0.038	0.003	0.091	0.161	0.191	0.239	0.137
セイウチ	*0.895*	0.040	0.026	0.123	0.108	0.101	0.464	0.094
珍しい	*0.884*	0.046	0.011	0.173	0.188	0.171	0.416	0.186
場所	−0.884	0.047	0.002	0.277	0.240	0.193	0.162	0.244
飼育	0.883	0.047	0.011	0.076	0.131	0.205	0.354	0.165
動物	0.872	0.054	0.012	0.175	0.103	0.257	0.362	0.110
見応え	0.856	0.064	0.001	0.144	0.159	0.154	0.229	0.166
来る	0.838	0.076	0.010	0.293	0.406	0.274	0.480	0.244
ラッコ	0.838	0.076	0.014	0.173	0.091	0.184	0.399	0.150
混雑	−0.835	0.079	0.001	0.146	0.159	0.171	0.098	0.179
大変	0.833	0.080	0.006	0.110	0.078	0.167	0.200	−0.003
食事	0.830	0.082	0.001	0.157	0.124	0.174	0.204	0.119
たくさん	0.828	0.083	0.007	0.283	0.266	0.225	0.428	0.238
デート	−0.824	0.086	0.020	0.448	0.234	0.147	0.133	0.390

続いて、表 8 では外形的データに飼育種類数を用いて相関関係を分析する。

- 「面白い」や「珍しい」は飼育種類数と相関関係にある（面白い：0.979、珍しい：0.884）
  - → 飼育種類数が多いと必然的に珍しい動物を飼育することになり、それが来場者に面白いと認識され、相関関係が高くなっているか
- 「ジュゴン」や「セイウチ」が飼育種類数と相関関係にある（ジュゴン：0.953、セイウチ：0.895）
  - → 日本では鳥羽水族館だけがジュゴンを飼育しているため、鳥羽水族館は値が大きくなっているか
  - → セイウチは鳥羽水族館と鴨川シーワールドで飼育されているが鳥羽水族館だけが類似度が高くなっており、これはショーを行っているかどうかの差があらわれているか

表 9　来場者数との相関

	相関	p値	分散	海遊館	沖縄美ら海水族館	鴨川シーワールド	鳥羽水族館	名古屋港水族館
来場者数（万人）				217	281	80	83	199
スポット	0.981	0.003	0.011	0.313	0.341	0.116	0.117	0.240
近い	*-0.977*	0.004	0.003	0.132	0.069	0.211	0.189	0.136
きれい	0.948	0.014	0.012	0.209	0.313	0.047	0.106	0.268
優雅	0.944	0.016	0.019	0.299	0.421	0.091	0.118	0.186
水槽	*0.932*	0.021	0.025	0.381	0.429	0.061	0.138	0.208
混む	0.914	0.030	0.005	0.240	0.257	0.130	0.084	0.157
サメ	0.913	0.030	0.025	0.307	0.277	-0.032	-0.006	0.108
何度	0.904	0.035	0.004	0.341	0.402	0.230	0.305	0.338
泳ぐ	0.903	0.036	0.016	0.296	0.396	0.127	0.103	0.158
アシカ	-0.896	0.039	0.021	-0.018	-0.002	0.221	0.321	0.150
ジンベイザメ	0.894	0.041	0.043	0.448	0.584	0.139	0.126	0.192
夕方	*0.893*	0.041	0.020	*0.286*	*0.307*	0.092	-0.035	0.136
眺める	0.892	0.042	0.014	0.132	0.266	-0.058	0.067	0.091
動物	-0.879	0.050	0.012	0.175	0.103	0.257	0.362	0.110
巨大	*0.877*	0.051	0.045	0.431	0.509	0.098	0.044	0.129
違う	0.871	0.055	0.002	0.292	0.366	0.246	0.261	0.270
規模	*0.865*	0.058	0.004	0.304	0.300	0.161	0.230	0.300
大きい	*0.859*	0.062	0.008	0.407	0.422	0.192	0.312	0.329
施設	0.855	0.065	0.005	0.295	0.307	0.138	0.205	0.202
入場料	0.842	0.073	0.007	0.280	0.325	0.110	0.193	0.175

最後に表 9 では来場者数との相関を算出したものである。

- 「近い」と来場者数は逆相関の関係にある（近い：-0.977）
  - → この「近い」が家から「近い」ことを指しているのか、あるいは人と動物の距離が「近い」ことを指しているのかはわからない
- 「きれい」や「優雅」と来場者数は相関関係にある（きれい：0.948、優雅：0.944）
- 「巨大」や「規模」、「大きい」と来場者数が相関関係にある（巨大：0.877、規模：0.865、大きい：0.859）
  - → 延べ床面積と「最高」や「楽しむ」は逆相関の関係にあったことから、「巨大」、「規模」、「大きい」は水槽のサイズを指している可能性が高い（「水槽」と来場者数も相関関係にあるため）
- 来場者数は「夕方」と相関関係にある（夕方：0.893）

以上、代理変数を用いて算出した各施設の特徴と外形的データである延べ床面積、飼育種類数、来場者数との相関を算出し、相関、あるいは逆相関の関係が強いものについて考察しました（上記箇条書き➔）。これらの分析の結果（上記箇条書き●）はすべて仮説といえます。しかし、実務家がこれを用いる場合には、この結果を意思決定の際の根拠データとして用いることも可能でしょう。

しかし、単語の意味を文脈なしに特定すること、つまり上記のような相関分析の結果だけでは十分に内容を把握することが難しいこともあります。たとえば、「巨大」や「規模」、「大きい」と来場者数が相関関係にある一方で、延べ床面積と「最高」や「楽しむ」は逆相関の

関係にありました。「not 楽しむ」の施設の来場者数が多いとは思えないため、 「巨大」や「規模」、「大きい」が延べ床面積（施設の広さ）を指しているとは考えられず、「巨大」、「規模」、「大きい」は水槽のサイズを指している可能性が高い（「水槽」と来場者数の相関も強いため）と推測されます。このように、相関関係を算出しただけでは、その結果を十分に把握することは難しいこともあるということには注意を払う必要があります。

出現語間の相関関係を用いた分析 [28][29]

出現語間の相関関係を分析することで、仮説の検証を行ってみたいと思います。まず、水族館プロデューサー中村元氏のテレビ番組での下記の発言 [30] から仮説を構築します。

> 「水族館というのはデートに行くには一番いい場所なんですよ。(中略) たとえば、動物園だったりとか、テーマパークだったりすると、夏は、女性は汗だらだらで、化粧取れるから嫌だとかさ、あとハイヒール履けないから嫌だとか、オシャレできなかったりするじゃないですか。でも、水族館ってなんとなくちょっと知的で健全で涼しくて、オシャレもできるし、という感じで。(後略)」

この発言から検証する仮説として①水族館はテーマパークなどと比較してデートで訪れる場所である、その理由としては汗をかかずに楽しめる、つまり②エアコンが効いているためである、の 2 つを立てました。

表 10 は各施設名と「デート」の類似度を低いものから高いものへ並べたものです。水族館は斜字で記載し、先頭に★記号を付しています。

表 10　各施設名と「デート」の類似度

施設名	デートとの類似度
三井アウトレットパーク木更津	−0.5497
三井アウトレットパーク滋賀竜王	−0.4260
ディズニーランド	−0.3163
★鴨川シーワールド	−0.2496
ディズニーシー	−0.2035
ナガシマスパーランド	−0.1644
旭川市旭山動物園	−0.1003
上野動物園	−0.0934
三井アウトレットパークジャズドリーム長島	−0.0216
神戸市立王子動物園	−0.0003
アドベンチャーワールド	0.0615
神戸三田プレミアム・アウトレット	0.0635
★沖縄美ら海水族館	0.0779
★鳥羽水族館	0.1102
三井アウトレットパークマリンピア神戸	0.1617
USJ	0.1924
東山動植物園	0.2372
横浜・八景島シーパラダイス	0.3909
★名古屋港水族館	0.4638
★海遊館	0.7003

---

[28] ここでのハイパーパラメーターは学習方法：supervised、次元数：100、学習回数：30、学習率：0.05、文字 N-gram：0~0 です。

[29] ここでの分析結果は竹岡(2018d)および竹岡(2019)で発表されたものを再掲したものです。

[30] 株式会社 TBS 制作、2017 年 5 月 30 日放送の「マツコの知らない世界」より。

これを見ると、水族館と「デート」の類似度が高い傾向にある、つまり水族館という施設のデート属性が高いことがわかります。次に表 11 は各施設名と「旅行」との類似度です。

表 11　各施設名と「旅行」の類似度

施設名	旅行との類似度
神戸三田プレミアム・アウトレット	−0.5327
三井アウトレットパークマリンピア神戸	−0.4731
★鴨川シーワールド	−0.3812
東山動植物園	−0.3056
★名古屋港水族館	−0.1993
三井アウトレットパーク滋賀竜王	−0.1738
ディズニーシー	−0.1658
ナガシマスパーランド	−0.1438
三井アウトレットパークジャズドリーム長島	−0.1253
三井アウトレットパーク木更津	−0.0904
神戸市立王子動物園	−0.0364
★海遊館	0.0766
旭川市旭山動物園	0.0887
上野動物園	0.1046
★鳥羽水族館	0.1704
★沖縄美ら海水族館	0.1844
USJ	0.2346
アドベンチャーワールド	0.3403
ディズニーランド	0.4515
横浜・八景島シーパラダイス	0.5264

「旅行」はテーマパークとの類似度は高いが、水族館とはそれほど高くないことがわかります。「デート」と「旅行」の相関係数は 0.179 と低く、これら各施設間、あるいはカテゴリー間にはデート属性と旅行属性において違いがあることが推測されます(図 5)。

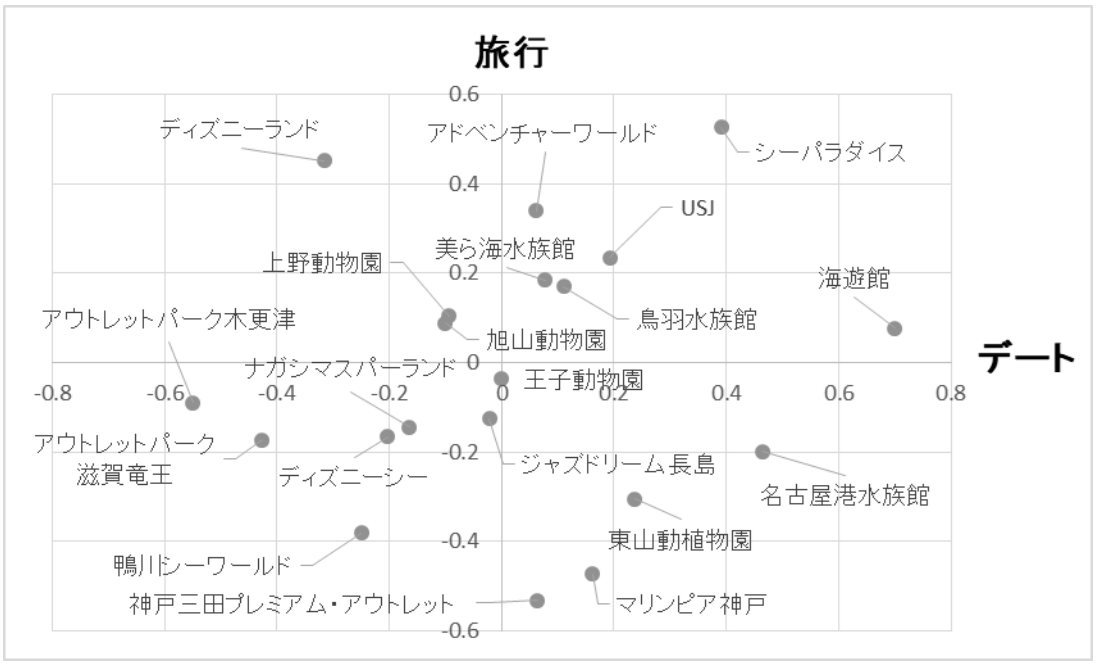


図 5　各施設における「デート」と「旅行」の知覚マップ

次に仮説の②です。夏場に汗をかかないのはエアコンが効いた施設内での行動が多いためと考えられます。表 12 は各施設名とエアコンの類似度です。水族館は全体として類似度が高い傾向にあります。

表 12　各施設名と「エアコン」の類似度

施設名	エアコンとの類似度
三井アウトレットパーク滋賀竜王	-0.588
神戸市立王子動物園	-0.411
ディズニーシー	-0.355
ナガシマスパーランド	-0.328
東山動植物園	-0.316
ディズニーランド	-0.256
三井アウトレットパークジャズドリーム長島	-0.241
三井アウトレットパーク木更津	-0.229
USJ	-0.090
旭川市旭山動物園	-0.083
上野動物園	-0.011
アドベンチャーワールド	0.104
★鴨川シーワールド	0.133
三井アウトレットパークマリンピア神戸	0.258
神戸三田プレミアム・アウトレット	0.321
横浜・八景島シーパラダイス	0.326
★沖縄美ら海水族館	0.329
★海遊館	0.469
★名古屋港水族館	0.771
★鳥羽水族館	0.779

図 6 は各施設名と「エアコン」・「デート」の類似度を知覚マップとしてプロットしたもの

です。「エアコン」と「デート」の相関係数は 0.650、p 値 0.002 と強い相関関係にあることがわかります（同様に、「エアコン」と「旅行」の相関を計算したところ 0.144、また「エアコン」と「家族」では-0.092 と低いものでした）

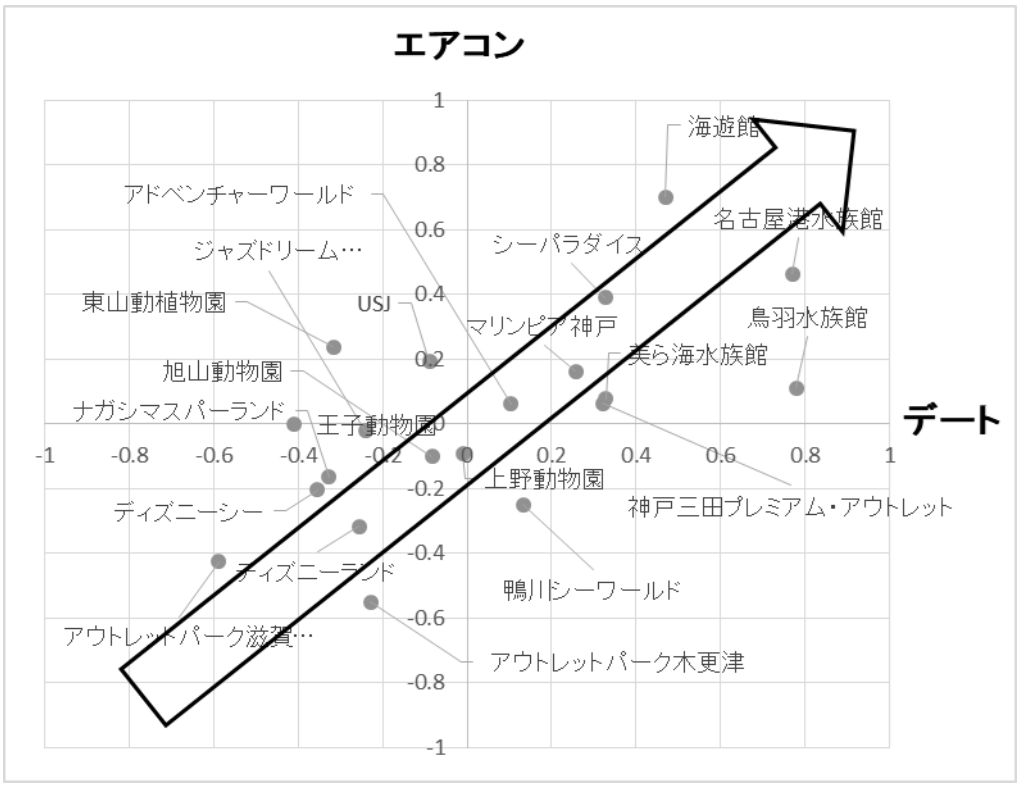


図 6　各施設における「デート」と「エアコン」の知覚マップ

以上から、仮説①および②はその妥当性が高いことが推測されます。

分析結果の活用時の注意

以上、Vector to を用いた分析事例を見てきました。このように、分散表現を用いたテキストマイニングによって、商品カテゴリー・サービスを超えた分析が可能となり、またアンケートを用いた調査のようにあらかじめ項目を決める必要がないため、効率の良い分析が可能になります。しかし、クチコミというデータには、それが消費者の意見のごく一部でしかないという問題も存在します。つまり、クチコミを投稿するのはその商品やサービスの利用者のごく一部であり、またクチコミを投稿する際には、その商品やサービスにおける経験を振り返り、そのすべてを書き込むのではなく、何らかの基準で選択した結果を書き込むことになります。ゆえに、クチコミを分析した結果だけを結論を導き出す道具とすることは控えるべきでしょう。しかし、分析者が本格的な分析を行う前のパイロットスタディとしては大変有用です。これらの手法を用いて可視化された特徴をもとにアンケートを作成し、それを分析すれば、これまで以上に効率よく、様々なことが分析可能になると思われます。

分散表現テキストマイニングについて

テキストマイニングと分散表現テキストマイニング

現在のテキストマイニングの主流は計量テキスト分析と呼ばれるものです。計量テキスト分析とは樋口(2014)によれば「計量的分析手法を用いてテキスト型データを整理または分析し、内容分析(content analysis)を行う方法(p.15)」です。その基本は文章を分かち書きし、出現する単語を集計すること、より進んだ分析としては文章内での単語の共起関係を集計し、その集計値をもとに単語間の関係を共起ネットワークや階層的クラスターとして描いたりすることです。これらの手法は多様な分野で広く用いられており、様々な質的データの分析を定量的に行うことを可能にしています。このように、これまでは困難であった質的データへの量的アプローチを可能にした計量テキスト分析ですが、意味へのアプローチに弱みがありました。

たとえば、「テキスト」という単語があります。この単語からどういう意味を読み取ればよいでしょうか。大学生であれば「教科書」という意味を読み取るかもしれません。テキストマイニングを用いている人は「文」という意味を読み取るかもしれません。単語は文脈によって意味が異なります。しかし、シンプルな計量テキスト分析では、文脈によって異なる意味を持っている単語を、形態素(意味を成す音素の最小単位)が同一であるということで、同一のものみなしていました。ここでシンプルという形容表現を用いたのは、計量テキスト分析自体が意味へのアプローチができない分析手法であると言いたいのではなく、使い方、つまり単純な出現語数の集計によってはアプローチできないという意味だからです。計量テキスト分析では、意味へのアプローチの方法として共起分析を行います。共起分析とは、単語 A と単語 B が共通の文脈で登場するとき、単語 A と単語 B には何らかの関係があり、この頻度や強さを測定することで、単語 A と単語 B の関係性を計測するというものです。この関係性ですが、重要な点としては関係性の強さもありますが、複数の語が出現する共起関係を明らかにすることで、それぞれの単語が持つ意味の広さを制限することができることがあげられます。つまり先ほどの「テキスト」が「経営学」という単語と同じ文章内で登場していれば「教科書」という意味で使われていることが推測でき、また「分析」という語と同じ文章で登場していれば「文」という意味で使われていることが推測できます。このような共起関係の性質を利用して樋口(2011)や竹岡(2016)では複数語による組み合わせでコーディングルールを作成し、分析対象となる文章の意味を分析しています。

たとえばコーディングルールとしては

「メモリカード」　∩　{「ファイル」　∪　「写真・動画」}　∩　保存

のように出現単語の組み合わせを設定し、これに対して「メモリカード保存」というラベルを張る、つまり、メモリカードとファイルあるいは写真・動画、そして保存という概念が文章中に入っていれば、メモリカード保存についての話題に関する文章であると推定すると

いうものです。ここで単語ではなく概念としたのは、「メモリカード」にはメモリカード、SD、SD カード、ミニ SD、マイクロ SD、メモリスティックという単語が含まれるためです。このように頻出する類語に関しては概念としてまとめることで、より詳細な意味の抽出が可能になります。

このような方法によってテキストの持つ意味にアプローチすることは可能になりますが、限界もあります。そもそも計量テキスト分析は後述の one-hot ベクトル表現のようなもので、すべての文意を抽出するためには巨大かつ疎なベクトルデータを用いる必要があり、計算数が出現単語数に合わせて増加するという問題があります[31]。そこで、樋口(2011)や竹岡(2016)では、コーディングルールに使用する単語を頻出語などに限定したうえで、頻出共起関係だけを用いてコーディングルールを作成し、分析を行っています。しかし、このような手法では、コーディングルールとして作成されたラベルの数が少なければ頻出共起関係だけを分析の対象とすることになり、多くの文章が分析の対象外になりかねません。これに対応するためにはコーディングルールを相当数作成する必要がありますが、それは計算数の増大を招くことになります。

このような問題に対するひとつの解決法がベクトルの次元を数百程度に圧縮し分析することを可能にする分散表現を用いたテキストマイニングです。分散表現とは「任意の離散オブジェクトの集合Vに対して、各離散オブジェクト v ∈Vにそれぞれ D 次元のベクトルを割り当て、離散オブジェクトを D 次元ベクトルで表現したもの(鈴木他, 2017, p.58)」とされ、また離散オブジェクトとは「人や物の名前、概念のように、物理的に計測できる量を伴わず、通常、記号を用いて離散的に表現するもの(同上)」です。テキストマイニングの文脈で分散表現を平易に表現すれば、分析対象としてある各文章や各単語を決められた次元のベクトルで表現したものといえます。

[31] たとえば、1 万語の異なり語が登場するテキストを分析する場合には、1 万次元のベクトルを用いる必要があります。

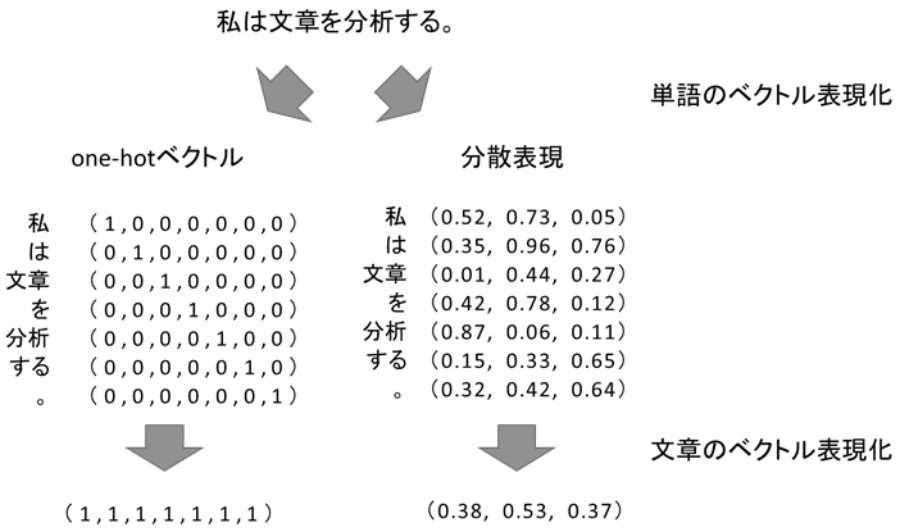


図 5　文章のベクトル化－one-hot ベクトルと分散表現

文章や単語をベクトル化する方法には、局所表現と呼ばれるベクトルのすべての要素の中である要素のみが 1、その他が 0 で表現された one-hot ベクトル表現もありますが、これは分析対象となるすべての文章に登場する異なり語数がベクトルの次元数となるため、巨大かつ疎な行列を計算することになります。

他方、分散表現では、単語のベクトル化には Skip-gram や CBOW(continuous bag of words)といった手法がとられ、これらによって各単語のベクトルが算出されます。これらの手法を用いることによって各単語のベクトルの次元数を通常で 100 次元、wikipedia の全データを用いた場合でも各単語を 300 次元で表現することが可能となり、局所表現と比べて大幅に次元数を圧縮することができます。

分散表現による意味へのアプローチ

分散表現を用いた分析においては単語をベクトル表現に変換する過程で単語に文脈情報を持たすことができます。たとえば、図 2 左のような離散表現の場合、出現回数を集計しただけでは単語間の類似度を測ることはできません。しかし、同図右のように Skip-gram や CBOW[32]を用いてベクトル表現に変換すれば単語間の類似度をコサイン類似度 [33]で測定することができます。

[32] Vector to では CBOW を用いた分散表現への変換はできません。これは CBOW が Skip-gram では可能な subword information (部分語情報)を利用できないためです。Bojanowski et al.(2017)は、subword information を用いることで、まれにしか出現することのないような単語でも信頼できる分散表現を得ることができるとしています。

[33] コサイン類似度は次の式によって計算します。$\cos(A,B)=(A\times B)/|A||B|$

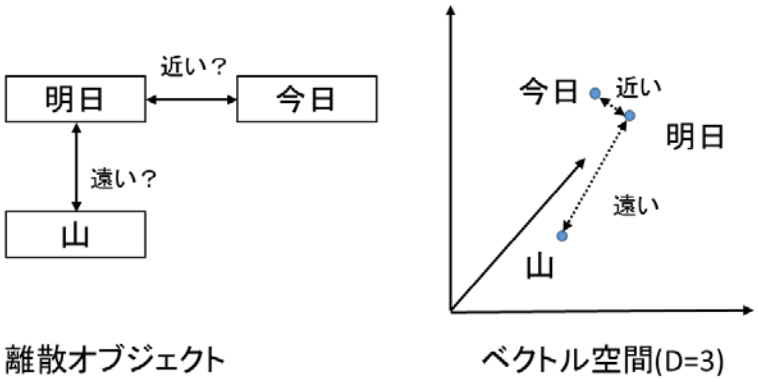


図 6　離散オブジェクトとベクトル空間

(鈴木他, 2017, p.59　図 3.2 を一部改変)

このようなことを可能にしているのが Skip-gram や CBOW の背景にある分布仮説(Harris, 1954, Rubenstein & Goodenough, 1965)です。分布仮説とは類似する文脈で出現する単語は、意味的にも類似しているという考えであり、Skip-gram や CBOW はこの仮説を背景に単語の分散表現、つまりベクトルを算出しています。ゆえに図 6 右のように「今日」と「明日」という類似する文脈で出現する可能性の高い単語はベクトル空間上で近く、「明日」と「山」のように同一文章で登場する可能性は高いが、類似する文脈で登場する可能性の低い単語はベクトル空間上では遠くなります。

先述の通り、分散表現では各単語を Skip-gram などの手法でベクトル表現に変換することで、単語間の類似度をコサイン類似度として計算することができ、また、単語間の関係をベクトルのオフセットで推定することも可能となります(図 7 は二つの単語の単数形と複数形の関係を図示したもの)。

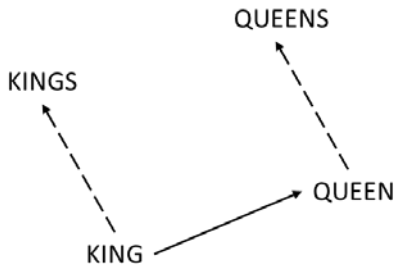


図 7　単語のオフセット

(Mikolov et al., 2013, p.749 figure2 を一部改変)

このような分散表現の最大の特徴は、加法構成性を持つベクトルの加減算による類推にも応用することができる点にあります。たとえば図 8 のように「King」－「Man」+「Woman」

は「Queen」に近似する(Mikolov & yin et al., 2013)といった類推が可能となります。

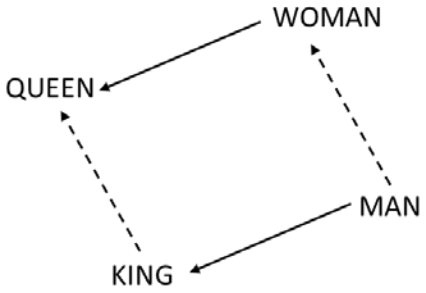


図 8　分散表現の加法構成性

分散表現テキストマイニングでは、先ほどの「テキスト」という単語の中に「教科書」という意味も「文書」という意味も含みこんでいます。ゆえに「テキスト」と「教科書」の類似度、「テキスト」と「文書」の類似度、これらがともに高い値で算出されるはずです(分析対象=学習に使用したテキストによっては異なる可能性もあります)。このように、分散表現テキストマイニングは、計量テキスト分析に比べて意味への接近が容易であるといえます。

以上のように、単語の分散表現には単語間の関係をコサイン類似度で算出したり、ベクトルの加法構成性を利用して意味の類推を行うことができます。ここまで単語の分散表現について説明してきましたが、fastText では学習の成果を用いた文章の分類も可能です。文章のベクトル化 には、文章を構成する単語のベクトルと EOS(end of sentence)「</s>」のベクトルを合計した後に文章を構成する単語数+1(この+1 は EOS)で除したものを用いています。このように計算することですべての文章がベクトル表現化され、計量テキスト分析では困難だったすべての文章の持つ意味を分析の対象にすることができます。なお、文章分類を行う際の単語の分散表現化では Skip-gram や CBOW を使用せず、異なる手法を用いているようです[34]。

分散表現テキストマイニングの問題点－計量テキスト分析との比較の観点から

分散表現テキストマイニングと計量テキスト分析を用いたテキストマイニングの相違点としては、計量テキスト分析が単語の出現回数に基づいて共起分析や階層的クラスター分析を行うのに対して、分散表現テキストマイニングでは単語や文章のベクトルを加法構成性に基づく加減算やコサイン類似度によって分析を行う点にあります。

単語や文章を分散表現にすることで、計量テキスト分析では困難であった単語や文章の持つ意味への接近や、類推、分類が可能になる一方で、人の認識可能な特徴のかなりの部分が失われることにもなります。たとえば、図 1 にあるように、すべての単語はベクトル表現

[34] これについては白木義彦氏が説明したものが下記 web ページにあります。
https://www.slideshare.net/shirakiya/fasttext-71760059

に変換されます。この数字の羅列を見て何かをそこから読み取ることのできる人間がどのくらい存在するでしょうか。他方の計量テキスト分析であれば共起関係の分析過程で出現するデータも集計値であり、大変理解しやすいものになっています。

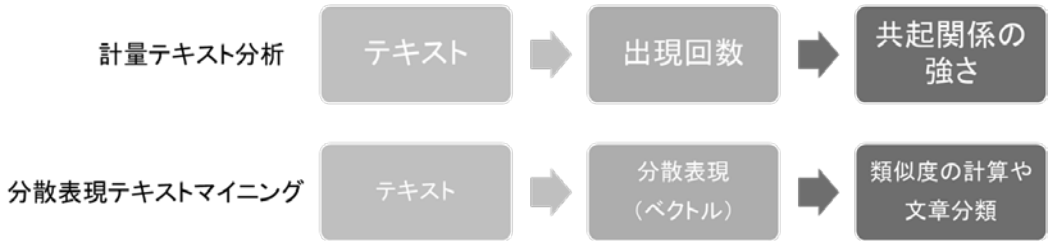


図 9　計量テキスト分析と分散表現テキストマイニングの違い

ゆえに、文章分類に基づくカテゴリー横断分析においては、分類器によって分類された何らかの特徴があると考えられる文章クラスターに対して、従来の計量テキスト分析の中でも最もシンプルな頻度分析を行うことによって、分類器の結果だけではわからない文章クラスターの特徴を抽出しました。しかし、クラスターの持つ特徴を計量テキスト分析するだけでは見落としてしまうものも存在します。なぜなら計量テキスト分析、特に出現頻度に基づく分析は大きな特徴の抽出には向いていますが、ごく少数しか現れない特徴の抽出は苦手としているからです。このような特徴は、共起ネットワークや階層的クラスター分析では足切され、比較的多数の単語を分析の対象として使用することのできる自己組織化マップ(樋口, 2014)においても、分析対象になりうるかどうかはコンピュータの性能に依存するところがあります。

他方で、このような文章クラスターの特徴の抽出に関しては、教師データさえ用意すれば、これまでの計量テキスト分析ではできなかった分類を、文章の分散表現への変換とそのクラスター化によって可能となります。また、機械学習の仕組みを用いることで、計量テキスト分析では人が出現単語を見渡したうえでコーディングルールを作成する必要がありましたが、分散表現テキストマイニングでは直感的にテキストに対してラベルを張りさえすれば、コーディングルール自体はソフトウェアが自動的に作成し、かなりの精度での分類をしてくれます。

今後ますます発展すると思われる機械学習の技術は、これまでには想像のできなかった分析手法を我々に提供し、これまでとは異なる分析、あるいはこれまでと同じ分析であったとしても異なるルートから結果に到達することを可能にすると思われます。しかし、分散表現テキストマイニングは万能ではありません。開発者自身も、別の研究では、現在も KH Coder を使用しています。ゆえに、今後は分散表現テキストマイニングについての応用を考えるとともに、これまでの計量テキスト分析との相互補完的な併用についても検討する必要があると考えます。

分散表現テキストマイニングの問題点—再現性の観点から [35]

分散表現テキストマイニングの最大の問題点は再現性に関する問題です。再現性とは「誰もが、同じ手続きを用いて再び同じ測定ができ、最初の測定結果を追試できる可能性のこと(May, 2001 邦訳書 p.107)」で、ニューラルネットワークを用いる際には、これを担保することが難しいのです。

分散表現テキストマイニングを使用するにあたって、単語を分散表現化（ベクトルに変換）する際、fastText では重みづけに用いる行列を生成する部分でランダムに初期値を生成する処理が行われます(図 9 内 Win と Wout)。そのため、全く同じテキストデータを fastText で処理しても、毎回異なる結果が出力されることになります。ここでの結果とは、ひとつの単語に対して設定した次元でベクトル表現されたものを指します。たとえば、「テキスト」という単語を含むいくつかの文章を fastText で学習させた場合、「テキスト」という単語に対して 1 回目は「(0.52, 0.73, 0.05)」であったものが 2 回目には「(0.35, 0.96, 0.76)」という結果が出力されるということです。このようにベクトル表現化した場合の数値が異なることで、これらを用いて計算する類似度も異なることになります。ある程度は epoch(学習回数)を多く設定することで解消可能な問題ですが、結果を完全に一致させることは困難だと思われます。

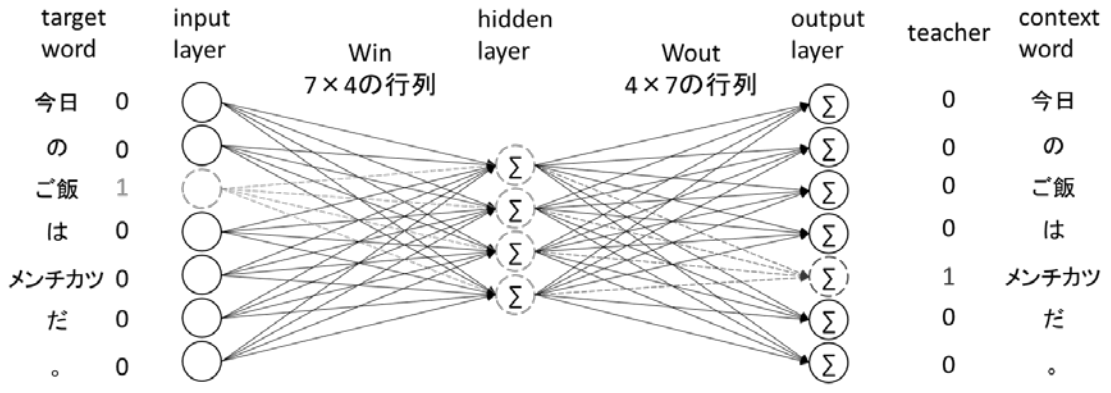


図 10　fastText で用いるニューラルネットワーク図（簡略版）

同じデータを扱っても異なる結果が生じる、つまりその研究に再現性がないということになり、研究結果の信頼性を損なうことになりかねません。このような再現性の問題は、今後、機械学習を含むニューラルネットワークを用いた AI 技術が研究方法に取り込まれる中で、ひとつの論点になる可能性もあります。以下では、この問題に関する考察を行うことで、

[35] ここは竹岡(2019)の内容を一部修正したものです。

今後の研究につなげていきたいと思います。

研究の進め方は研究者や研究対象、そして選択する手法によって異なるが、テキストマイニングにおいては、対象の決定から、考察にかけて、およそ図 11 のようなプロセスで進むと思われます。この中で、分析対象の決定は再現性の対象にはならず、また、考察も再現性の対象にはなりません。なぜなら、すでに決定されている分析対象について、その分析を再現するのが再現性であり、また、分析の結果あらわれた数値などをもとに何らかのインプリケーションを引き出すのが考察だからです。再現性の対象に含まれるのは、データの収集から分析までの過程です。

図 11　テキストマイニングのプロセス

以下が、各プロセスにおける再現のために必要な要素です。

1. データの収集
   - 収集を行った日、場所、データの境界、方法など
2. クリーニング、整形、サンプリング
   - 表記ゆれを修正する場合には「A を B に変換」などの情報
   - 何（記号など）を、どの順番で削除したのか
3. 分かち書き
   - 通常何らかのソフトウェアを用いて行われるため、どのようなソフトウェアを使用したのかといった情報
4. 集計と分散表現化
   - 計量テキスト分析では単語ごとの集計が行われるだけなので特に問題はない
   - 分散表現テキストマイニングでは単語のベクトル表現化が行われるが、ここで上述のランダムに生成される初期値を用いた計算がなされるため、再現性の問題が生じます
5. 分析
   - 再現に必要な情報として分析の詳細な手順など
   - 初期値をランダムに生成する K-means 法のような機械学習技術を用いる場合には再現性の問題が再び生じます

厳密な再現性があるとは、このすべてのプロセスを第三者が研究者と同一に行うと全く同じ結果が得られる状態を指します。分散表現テキストマイニングを行う際に再現性の問

題となるのは、この分散表現化のプロセスです。これは第三者の再現実験だけではなく、研究者自身が行っても、完全に一致する結果は得られません。

この問題に関する結論としては、その他のプロセスは通常のテキストマイニングと同一であり、問題となるのは分散表現化のプロセス、具体的にはこのプロセスで生成される学習済みモデルが試行のたびにごくわずかに変わることのみであること、しかし、分散表現化に使用した全単語およびその際のハイパーパラメーター(学習回数や学習率など、分析者が分散表現化に際して設定する値)は把握可能であり、それゆえに近似することは可能であること、つまりおおよその追試・検証は可能であり、また他社も同一のモデルを用いれば同じ分析の結果を得ることもできること、よってこの手法は、完全な厳密性を要求されるような分析でない場合には、十分に採用する価値のあるものであると考えます。

ライセンスや外部ソフトウェアについて

ライセンスについて

Vector to は開発に python3.5 および 3.6 を使用しています。python は機械学習や AI の発展もあり、現在注目されている言語のひとつです。Vector to のライセンスは GPL です。もし必要な機能が Vector to に存在しない場合には自由に改変することも可能です。特に学生の皆さんにはどんどん Vector to を使用していただき、柔軟な発想力で Vector to を改良していただければと思います。

Vector to が使用している主な外部ソフトウェアおよびモジュール

Vector to は主に下記のソフトウェアやライブラリ等を用いることで機能しています。

- fastText
- Mecab
  - mecab-ipadic-NEologd
- MySQL
- python
  - pyfasttext
  - NumPy
  - Matplotlib
  - PyQt
  - scikit-learn
  - Pandas

おわりに

最近では AlphaGo の発表や、その社会への貢献が目に見えるものとなってきていることから、機械学習をはじめとする AI への関心が高まっています。「AI 人材の不足」という言葉も頻繁にみられます。Vector to の開発前、初めて fastText に触れたときには、開発者も「これからの時代は AI を使えるか使えないかが大きな差になる」と感じました。それほど fastText の出す結果と、その結果への到達の容易さ、そして応用範囲の広さに衝撃を受けました。しかし、Vector to の開発を進める中で「簡単に使用することを可能にするインターフェースがあれば、パラメーターを操作すること自体は容易であり、コンピュータの性能が高い現在であれば、これまでの表計算ソフトを用いたデータ分析とそれほど大きく変わるものではないのではないか」と考えるようになりました。

大規模なデータを使用する場合には専門性が必要とされ簡単にはいかないかもしれませんが、100 万件程度のデータであれば現在のコンピュータでも十分に分析可能です。特に fastText は近年の機械学習を用いたソフトウェアでよく用いられている GPU コンピューティングではなく、CPU を用いて計算を行っており、導入のハードルが低いという特徴があります。このように、様々な側面から使用に向けたハードルを下げることができれば、使用に限ればですが、誰もが使用可能なもの、つまり現在の表計算ソフトによるデータ分析のレベルまでハードルを下げることができると思います。Vector to がこれに貢献できていれば幸いです。

今回 Vector to をフリーソフトウェアとして公開した背景には、フリーソフトウェアとして公開され、現在も様々なアップデートがなされている計量テキストマイニング・ソフトウェア KH Coder の存在があります。Vector to の開発者はこれまで 10 年近くの間、このソフトウェアを使用して様々な研究を行ってきました。KH Coder を使用する中でテキストマイニングという分析手法について深く考えるようになり、様々な経験と考察を行ってきました。その経験と考察が Vector to につながっています。開発者は KH Coder というソフトウェアに育てられ、その延長線上に Vector to が生まれたといってもおおげさではないかもしれません。このような人と道具、学習、知識構築の連鎖が Vector to によっても起こることが研究者としての希望であり、大学教員としての希望でもあります。

Vector to の開発者はプログラマーではありません。経営学系統を専門とする研究者です。また、プログラミングを始めて 1 年の、なかなかの初心者です。そのため本職の方から見れば非常に拙いコードで全体が構成されており(オブジェクト指向がいまだにしっかりと理解できていませんし、独学のため本職の人であればこう書くであろうというところが全く分かっていません)、またインターネット上に存在する多くの方の知見を勝手に拝借することでこのような形で配布することができるようになりました。

今回の開発過程で感じたことは、世界は平坦で、公平で、かつ親切であるということです。様々な知識が無償で提供されており、それらを自由に改変して自分のプログラムの中に組み込むことができる、また、様々な掲示板上では質問と回答のやり取りに加えて、異なる環境下でのテストや教示なども行われており、相互作用的に新たな知識が構成されていく過程を後追いすることができました。Vector to はこのような多くの巨人の無償の知への貢献の上に出来上がり、成り立ったソフトウェアです。この Vector to がこれからの時代の学生たち、企業内のデータサイエンティストや研究者のお役に立つことができれば幸いです。

最後に、先述の通り、Vector to の開発者は経営学領域を専門としており、その分析手法や対象が経営実践における事象の可視化を目的としたものになる傾向があります。今後、様々な分野で活用され、異なる観点からの意見が集まることで、さらなる改良や機能の実装が進むことを切に願います。皆さんのご意見を頂ければ幸いです。

＜謝辞＞
Vector to は科学研究費補助金による助成を受けた研究成果の一部です。末筆ながら感謝申し上げます。

参考文献

鈴木潤、海野裕也、坪井祐太 (2017) 「言語処理における深層学習の基礎」坪井祐太、海野裕也、鈴木潤『深層学習による自然言語処理』pp.43-90 講談社.

竹岡志朗 (2016)「イノベーションの普及過程における非連続性と連続性 - テキストマイニングにおける話題分析 - 」竹岡志朗、井上祐輔、高木修一、高柳直弥『イノベーションの普及過程の可視化 テキストマイニングを用いたクチコミ分析』pp.126-142 日科技連出版社.

竹岡志朗 (2018a) 「機械学習を活用したテキストマイニング ―クチコミを活用した商品・サービスカテゴリーの横断分析」『桃山学院大学経済経営論集』第 59 巻第 4 号 pp.101-122.

竹岡志朗 (2018b)「機械学習を活用したテキストマイニングー特徴抽出の方法に関する検討―」『日本情報経営学会第 76 回全国大会予稿集』pp.155-158.

竹岡志朗 (2018c)「機械学習を活用したテキストマイニングー外形的データを併用することによる特徴分析―」『日本経営学会第 92 回大会報告要旨集』pp.143-146.

竹岡志朗 (2018d)「機械学習を活用したテキストマイニングー概念間の相関分析による特徴の確認―」『日本情報経営学会第 77 回全国大会予稿集』pp.161-164.

竹岡志朗 (2019) 「機械学習を活用したテキストマイニング(2) ―仮説の発見と検証―」『桃山学院大学経済経営論集』第 60 巻第 4 号 pp.121-143.

竹岡志朗・高木修一 (2017) 「インターネットを用いた情報探索に関する検索エンジンと web リンクの観点からの考察」日本情報経営学会第 75 回大会発表資料.

高木修一、竹岡志朗 (2018) 「経営学におけるテキストマイニングの可能性ー仮説構築志向の利用方法ー」『富大経済論集』第 64 巻 2 号, 印刷中.

西内啓 (2013)『統計学が最強の学問である』ダイヤモンド社.

樋口耕一 (2011)「現代における全国紙の内容分析の有効性 ー社会意識の探索はどこまで可能か―」『行動計量学』Vol.38-1 pp.1-12.

――――― (2014)『社会調査のための計量テキスト分析』ナカニシヤ出版.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017) "Enriching Word Vectors with Subword Information," in *Transactions of the Association for Computational Linguistics*, Vol.5, pp.135-146.

Harris Z., (1954) "Distributional structure," in *Word* Vol. 10, No. 23, pp. 146-162.

May, T. (2001) *Social Reseach 3rd edition,* Open Univesity Press (中野正大監訳 (2005)『社会調査の考え方 論点と方法』 世界思想社).

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013) "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, pp. 3111-3119.

Mikolov, T., Yih, W. T., Zweig, G. (2013) "Linguistic regularities in continuous space word

representations," in *Proceedings of Naacl-HLT 2013*, pp. 746-751.

Rubenstein, H., & Goodenough, J. B. (1965) "Contextual correlates of synonymy," in *Communications of the ACM*, 8(10), pp.627-633.