金丸 洋平

店舗販売員

自己紹介

現在、小売りの販売員として日々商品を販売しています。
売り場のレイアウト変更や商品の陳列が得意です。とてもキレイで気持ちのいい売り場に仕上がっていると思います。
売上前年比は直近３年の平均で１１５％達成。

この先、新しい仕事にチャレンジするために自己学習を続けています。

スキル

プログラミング(Python,R,その他)
統計学基礎知識
データ集計・探索的データ分析

趣味

景観を眺めること
音楽を聴くこと
データ、プログラミングの勉強

連絡先

携帯：080-3001-4457
Email：YouNat.Kana@outlook.com
skyrocket_xvii@icloud.com

参考書籍

統計学・数学関連

統計学入門　はじめての統計学　ゼロから学ぶ統計解析　基礎統計学Ⅰ
統計学基礎(2級対応)　統計学(1級対応)　現代数理統計学の基礎
ゼロから学ぶ微積分　ゼロから学ぶ線形代数　最適化数学　多変量解析がわかる
多変量解析のはなし　バイオサイエンスの統計学　医療を志す人のための基礎数学
やさしく学ぶ機械学習を理解するための数学のきほん
人工知能プログラミングのための数学　その問題、数理モデルが解決します

機械学習・ＡＩ関連

入門Python3　Pythonではじめる機械学習
Pythonではじめるデータラングリング　ゼロから作るDeep Learning
Pythonクローリング&スクレイピング　Pythonによるデータ分析入門
Pythonデータサイエンスハンドブック　あたらしい機械学習の教科書
機械学習のエッセンス　データを読み解くアルゴリズムの技法　Kaggleで勝つ
Scikit-learnによる実践機械学習　RとPythonで学ぶデータサイエンス&機械学習
Python機械学習プログラミング　Python実践100本ノックシリーズ4冊

Ｒ関連その他

Rによるやさしい統計学　Rプログラミング入門　Rではじめるデータサイエンス入門　データ解析のための統計モデリング入門　Rで学ぶデータサイエンス(一般化線形モデル)　Rによる多変量解析入門　効果検証入門　SAS入門　実用SAS生物統計ハンドブック　Rによるテキストマイニング入門　実践Rによるテキストマイニング

データ分析関連

やってみようテキストマイニング　社会調査のための計量テキスト分析
調査系論文の読み方　戦略的データサイエンス入門　データ分析のための数理モデル入門　分析者のためのデータ解釈学入門　マーケティング・リサーチ入門
社会調査の考え方(上)

参考サイト

我楽多頓陳館(http://www.snap-tck.com/room04/c01/stat/stat.html)
米国データサイエンティストのブログ(https://datawokagaku.com/)
米国データサイエンティストがやさしく教えるデータサイエンスのためのPython講座(udemy)
キカガク流脱ブラックボックスコース(https://www.kikagaku.ai/)
SIGNATE Quest(https://quest.signate.jp/quests) 9課題/全16課題 実施済み
DjangoBrothers(https://djangobrothers.com/tutorials/blog_app/)
⇒DjangoBrosBook購入 実施済み
SQL攻略(http://sql.main.jp/)
TECHPROjin(https://tech.pjin.jp/blog/2016/12/05/sql練習問題-一覧まとめ/)
⇒全75問実施済み

学習まとめ

その１．社会やビジネスにおけるさまざまなデータと指標を知る。
・量的データ：間隔尺度、比例尺度
・質的データ：名義尺度、順序尺度
・相関：相関、自己相関、偏相関、疑似相関など
・データ標準化：ＭｉｎＭａｘ、Ｚ得点、中心化(平均値を引く)など
・交絡と中間因子
・多重共線性：ＶＩＦ
・データのバイアス：セレクションバイアス、選択バイアスなど
・データのバイアスを取り除く：ＲＣＴ、ＯＶＢ(脱落バイアス)、傾向スコア、ＤＩＤ、回帰不連続など

その２．データ前処理を実施する。
・サンプルサイズの確認(分析する上で十分なサイズであること)
・表記ゆれの補正
・欠損値、外れ値の確認と処置(グラフによる可視化)
・質的データの処置(ダミー変数化)
・各データの単位を揃える(無名単位化)
・必要であれば、新しいデータを作成する(データ+データ、データ×データ、(データ×データ)/総データ数など)

その３．データを正しく解釈して、目的に適う分析手法を選択する。
・回帰と分類
　線形・非線形回帰モデル、決定木モデル、勾配ブースティングモデル、ロジスティック回帰モデル、クラスタリング、ディープラーニングなど
・多変量解析
　線形・非線形重回帰、パス解析、因子分析、共分散構造分析、一般化線形モデル(対数線形モデル、ロジスティック回帰)、クラスタリング、コレスポンデンス分析など

その４．分析結果を評価する。
・各分析モデルについての評価方法
　予測の正解率、分類の正解率(適合率、再現率、調和平均)、可視化(ＲＯＣ曲線)、偏回帰係数の検定、因子負荷の検定、適合度の検定(カイ二乗値)、モデルの当てはまりの良さ(ＡＩＣ、ＢＩＣ)、構成概念の信頼性と妥当性の評価など

その５．分析結果を考察する。
オッズ比(ロジット関数)による効果指標、より重要な変数の特定、因果関係の特定　⇒　新たなＫＰＩ策定、経営リソースの再配分など

⇓⇓⇓

"初手として、その１～その３を実務レベルで修得する。"
"リサーチデザインについて学習する。"

実践１．頻出単語の抽出

データの種類：質的データ(非構造化データ)
データの対象：東野 圭吾著作「白鳥とコウモリ」の読書レビュー文(全５３９件)
具体的方法：
①Ｐｙｔｈｏｎを用いたＷｅｂスクレイピングにより読書レビュー文を取得。ＣＳＶ形式でローカルＰＣ環境に保存。
②ＣＳＶファイルの文字コードをＵＴＦ８からＳＪＩＳに変換。
③Ｒを用いてＣＳＶファイルを読み込み。以下の手順はすべてＲを用いて実施。
④ＲＭｅＣａｂライブラリのｄｏｃＤＦ関数によりＣＳＶファイルの形態素解析を実行。
⑤品詞を「名詞、動詞、形容詞」に限定する。
⑥名詞、動詞、形容詞それぞれの品詞細分類の内容を確認して、さらに絞り込む。
⑦各品詞ごとの頻出単語上位１０個を表示。

出力表示：

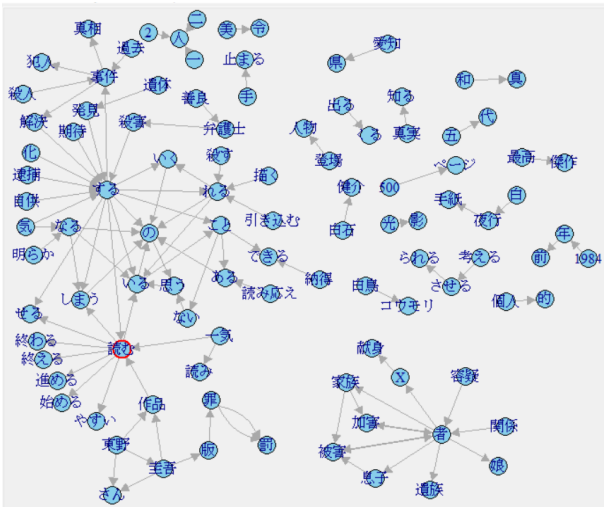
TERM	POS1	POS2	book_reviews_csv_sjis.csv	TERM	POS1	POS2	book_reviews_csv_sjis.csv	TERM	POS1	POS2	book_reviews_csv_sjis.csv
事件	名詞	一般	377	する	動詞	自立	1009	ない	形容詞	自立	187
東野	名詞	固有名	320	読む	動詞	自立	411	面白い	形容詞	自立	127
被害	名詞	一般	251	いる	動詞	非自立	348	良い	形容詞	自立	50
作品	名詞	一般	245	ある	動詞	自立	324	切ない	形容詞	自立	49
圭吾	名詞	固有名	234	なる	動詞	自立	320	すごい	形容詞	自立	43
家族	名詞	一般	203	思う	動詞	自立	294	深い	形容詞	自立	36
加害	名詞	一般	202	しまう	動詞	非自立	183	分厚い	形容詞	自立	34
罪	名詞	一般	175	いく	動詞	非自立	164	いい	形容詞	自立	33
人	名詞	一般	144	できる	動詞	自立	97	長い	形容詞	自立	32
犯人	名詞	一般	144	くる	動詞	非自立	94	多い	形容詞	自立	30

頻出単語であるため、ユニークな単語はないものの、「事件、被害、犯人、東野、圭吾、分厚い、長い」から長編ミステリー小説であることが示唆される。
また、「面白い、良い」が頻出上位であることから、この作品の評価は高いと思う一方、「ない」が形容詞の頻出最上位であるため、もしかすると「面白くない」のかもしれない。

実践２．バイグラムとネットワークグラフの作成

実践１と同様のデータに対して、単語同士の繋がりをグラフにする。
具体的方法：
実践１の①～⑤までの手順は同様。
①ＲＭｅＣａｂライブラリのｄｏｃＤＦ関数の引数に「nDF = 1, N = 2」を与える。
N = 2 ⇒ バイグラムの指定。
nDF = 1 ⇒ バイグラムとして抽出した２つの単語を別々のカラムに出力する。
②出現頻度が１６以上のものに絞る。
③ネットワークグラフ作成用のｉｇｒａｐｈライブラリのｇｒａｐｈ_ｆｒｏｍ_ｄａｔａ_ｆｒａｍｅ関数により、バイグラムをネットワークオブジェクトに変換。
④ｔｋｐｌｏｔ()によりネットワークグラフを描画。

出力表示：



ネットワークグラフでは、単語同士の繋がりがよくわかる。
動詞の頻出単語で最も多かった「する」は、「逮捕、解決、自供、発見、殺害」などの単語と強い繋がりがあり、レビュアーは本作品の内容の要約を書く際に「する」を多用したことがわかる。
また、「する」の次に多かった「読む」は、「終わる、進める、やすい、一気、作品」などの単語と強く繋がっており、本作品の感想を書く際に「読む」を多用したものと思われる。
その他、名詞に関しては、本作品の登場人物の名前や著者の名前と作品のタイトル、著者の他作品のタイトルなどが出力されており、動詞の場合と同様に、本作品のあらすじと感想を記述する際に多用される単語であることがわかる。
名詞の場合は、頻出単語のみでもある程度レビューの内容を推察可能だが、動詞(とくに「する」)の場合は、バイグラムの方が明確にレビューの内容を推し量ることができる。

実践３．センチメント分析

実践１と同様のデータに対して、形態素解析した各単語の感情極性値を算出する。
具体的方法：
　実践１の①～③までの手順は同様。
　①単語感情極性対応表をダウンロードして読み込み。各単語の感情極性値：－１．０～１．０(離散値)
　②重複行を出力。同じ綴りと読みの単語については、それらの感情極性値の平均値を使用する。
　③読み込んだレビュー文を句点ごとの一文に分割する。
　④ｔｉｄｙｖｅｒｓｅライブラリのｔｉｂｂｌｅ関数により、データフレームに変換。それぞれの一文に対してＩＤ付与。
　⑤ＲＭｅＣａｂライブラリのＲＭｅＣａｂＣ関数により、一文ごとに形態素解析を実行。
　⑥形態素(単語)ごとに単語感情極性対応表の感情極性値を参照し、一文(ＩＤ)ごとにその合計を求める。
　⑦Ｘ：ＩＤ(一文)、Ｙ：ＥＭ(感情極性値)として、グラフを描画。

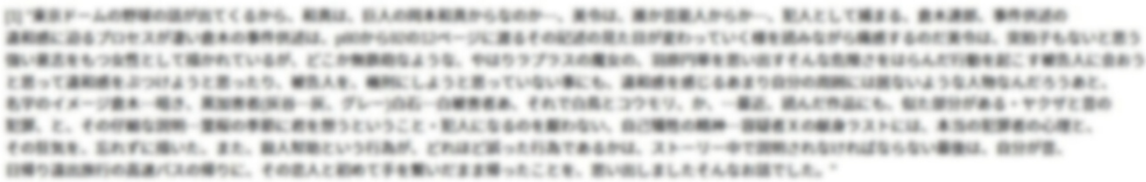
出力表示：

(１)形態素解析した各単語の感情極性値(EM)のサマリー

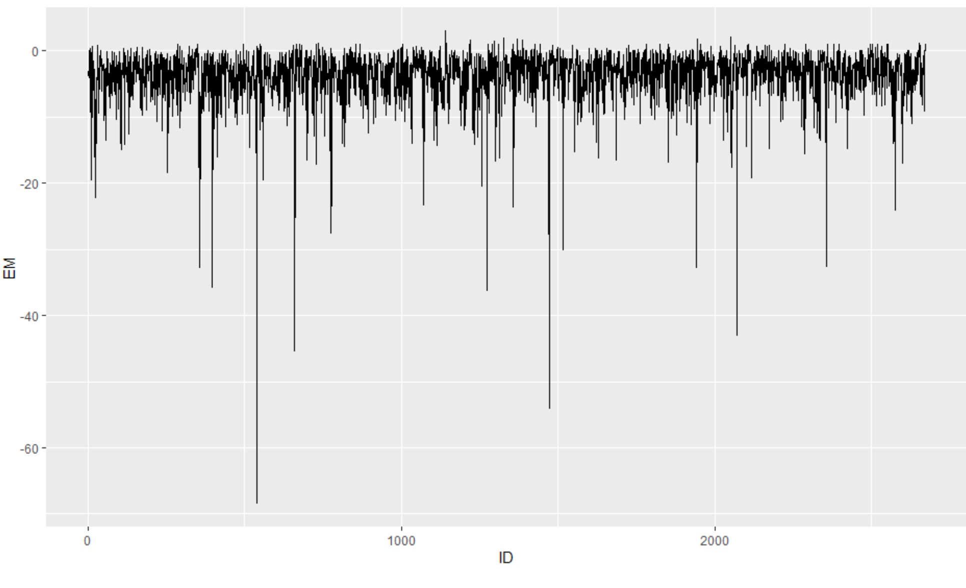
```
        EM
Min. :-68.408
1st Qu.: -5.249
Median : -2.911
Mean : -3.750
3rd Qu.: -1.239
Max. : 2.984
```

⇒平均値が-3.750 とマイナスになっており、最小値に関しては-68.408と極端に小さな値になっている。
極端な最小値により平均値がマイナスの値に引っ張られたかもしれない。

(２)感情極性値が最小の文章を抽出（レビュー内容は念のためぼかしを入れています。以降も同様。）
em_reviews %>% filter(EM == min(EM)) %>% left_join(sent_book_reviews) %>% select(S) %>% pull()



(３)感情極性値の描画１



ほとんどの一文がマイナス値になっている。ミステリー小説の場合、全体的に暗めのトーンで物語が展開されることが多いため、レビュー文も自ずとマイナス感情が多くなるのではないだろうか？
マイナス値が大きくてもブクログにおけるレビュー全体の評価平均は４．２２点と非常に高い評価を得ている作品である。
小説のような文芸作品は、感情極性値とレビュアーの評価との間に正の相関があったり、負の相関があったりするのかもしれない。
実用書のレビュー文との比較をしてみるとその違いが明確になるかもしれないので、今後の課題とする。

(４)感情極性値が１より大きい文章を抽出

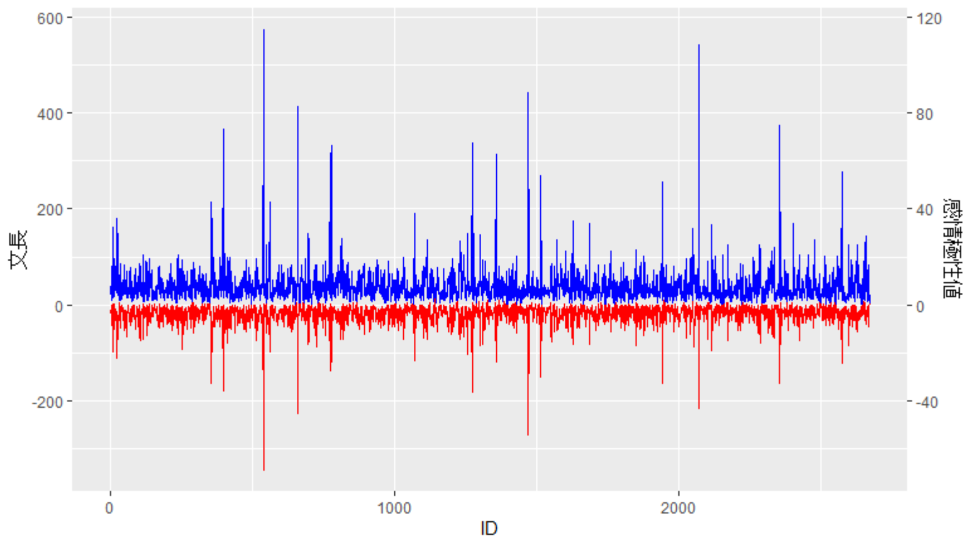
```
em_reviews %>% filter(EM > 1) %>% left_join(sent_book_reviews) %>% select(S) %>% pull()
```

(５)感情極性値が−３５より小さい文章を抽出

```
em_reviews %>% filter(EM < -35) %>% left_join(sent_book_reviews) %>% select(S) %>% pull()
```

感情極性値が１より大きい一文はとても短く、反対に感情極性値が−３５より小さい一文はとても長いことがわかる。そこで、感情極性値と一文の長さを左右のY軸に取り、２つのグラフを同時に描画してみる。

(６)感情極性値の描画２



こうしてみると、一文の長さの長短により感情極性値が変化していることがわかるが、一文が短くてもやはり感情極性値はマイナスの値を示しているため、本作品が全体的にシリアスな内容になっていることが示唆される。

実践４．ポジネガ分類

実践１と同様のデータに対して、形態素解析した各単語のポジティブ値とネガティブ値を算出する。
具体的方法：実践１の①〜③までの手順は同様。

①日本語極性辞書をダウンロードして読み込み。
②日本語極性辞書のＶＡＬＵＥ列に対して、ｎ⇒−１、ｐ⇒１、ｅ⇒０のそれぞれの値を入力する。
③読み込んだレビュー文を句点ごとの一文に分割する。
④ｔｉｄｙｖｅｒｓｅライブラリのｔｉｂｂｌｅ関数により、データフレームに変換。それぞれの一文に対して、ｓｔａｔｕｓ_ｉｄを付与。
⑤ＲＭｅＣａｂライブラリのＲＭｅＣａｂＣ関数により、一文ごとに形態素解析を実行。
⑥形態素(単語)ごとに日本語極性辞書のＶＡＬＵＥ列を参照し、一文(ｓｔａｔｕｓ_ｉｄ)ごとにｎとｐの合計を求める。
⑦Ｘ：’ＰＳまたはＮＥ’＋ｓｔａｔｕｓ_ｉｄ(一文)、Ｙ：ＶＡＬＵＥ(ポジネガ値)として、それぞれの高い値上位２０個を描画。

出力表示：

(１)形態素解析した各単語の感情極性値(EM)のサマリー

```
    VALUE
Min. :-15.000
1st Qu.: -1.000
Median : 0.000
Mean : -0.311
3rd Qu.: 0.000
Max. : 10.000
```

⇒平均値が-0.311 とわずかにネガティブ値が勝っている。
また、最小値は-15、最大値は10と単語感情極性対応表の感情極性値を用いた場合とは異なり、バランスの取れた結果になっている。

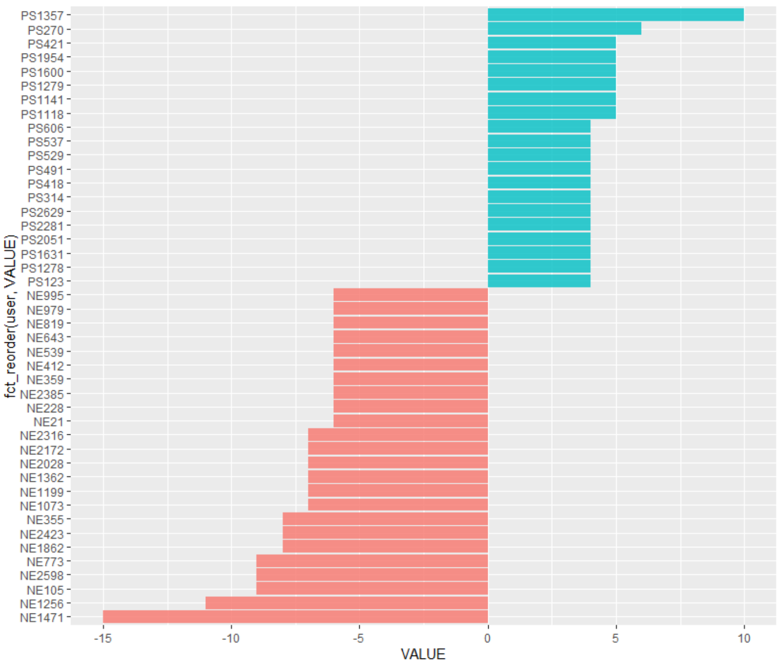
(２)ポジネガ合計値が最小の文章を抽出

```
dat2_calc %>% filter(VALUE == min(VALUE)) %>% left_join(sent_book_reviews) %>% select(S) %>% pull()
```

(３)ポジネガ合計値が最大の文章を抽出

```
dat2_calc %>% filter(VALUE == max(VALUE)) %>% left_join(sent_book_reviews) %>% select(S) %>% pull()
```

(４)ポジネガバーチャート描画



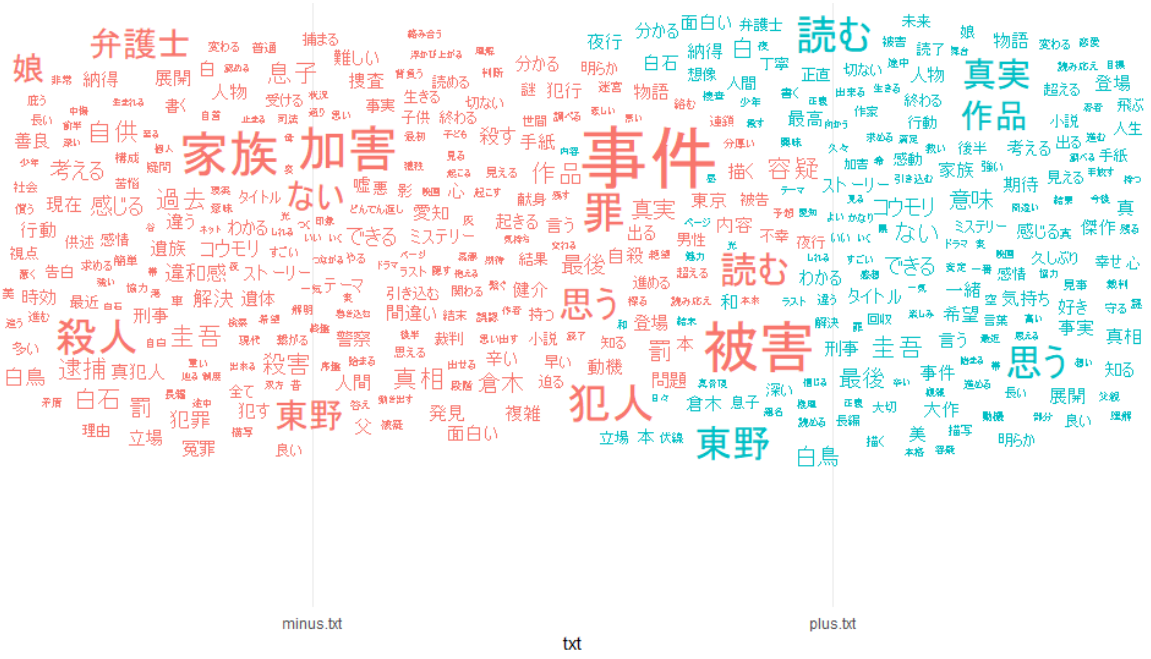
全体的にネガティブ値が高い傾向にあることがわかる。日本語極性辞書を基にポジティブ＋１、ネガティブ−１として計算しても傾向としてはネガティブ値が有意であった。

実践５．ワードクラウド作成

実践４で作成したｓｔａｔｕｓ_ｉｄ、ＶＡＬＵＥ(ポジネガ合計値)、Ｓ(一文)からなるデータフレームからワードクラウドを作成する。
具体的方法：

①実践４で作成したデータフレームからＶＡＬＵＥ＞０、ＶＡＬＵＥ＜０でＳ(一文)をそれぞれｐｌｕｓ．ｔｘｔとｍｉｎｕｓ．ｔｘｔに書き出す。
②ｐｌｕｓ．ｔｘｔ、ｍｉｎｕｓ．ｔｘｔについて、ＲＭｅＣａｂライブラリのｄｏｃＤＦ関数で形態素解析を実行。
③ストップワード＋αを除去。
④ｇｇｗｏｒｄｃｌｏｕｄライブラリのｇｅｏｍ_ｔｅｘｔ_ｗｏｒｄｃｌｏｕｄ関数を使ってワードクラウドを描画。

出力表示：



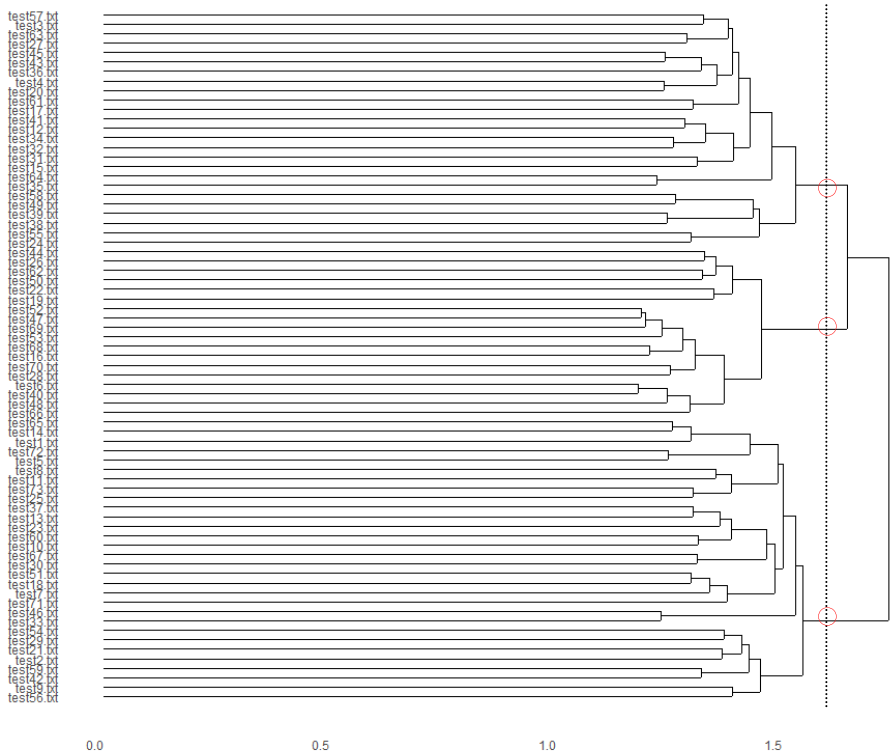
ＶＡＬＵＥ(ポジネガ合計値)によって、それぞれの一文をｐｌｕｓ．ｔｘｔとｍｉｎｕｓ．ｔｘｔに分けているため、青字(ｐｌｕｓ．ｔｘｔ)と赤字(ｍｉｎｕｓ．ｔｘｔ)の両方に出現する単語も存在するが、このワードクラウドは赤字の「事件、加害、被害、家族、犯人、殺害、罪」の出現頻度の高さを示している。
したがって、多くのレビュアーはこれらの単語を用いてレビューを書いているため、全体としてネガティブなレビューになっていると予想できる。

実践６．クラスタリング

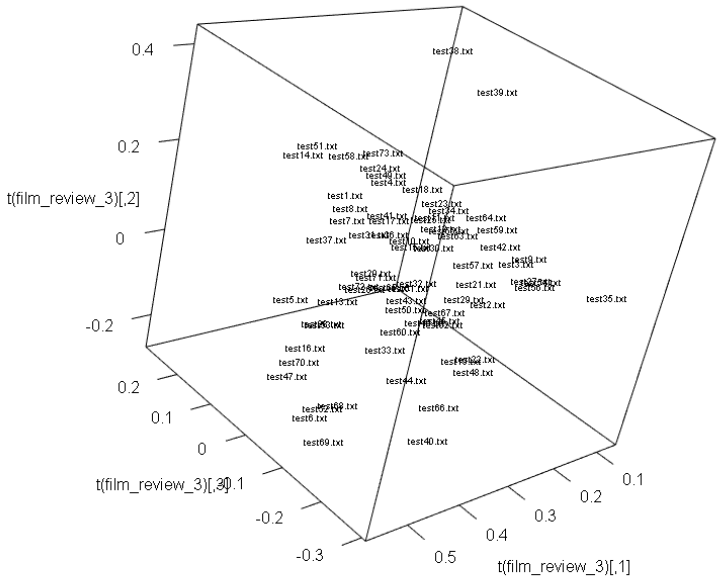
データの種類：質的データ(非構造化データ)
データの対象：ＢＯＮＥＳ原作「ＥＵＲＥＫＡ 交響詩篇エウレカセブン　ハイエボリューション」の映画レビュー文(全７３件)
具体的方法１：
①Ｐｙｔｈｏｎを用いたＷｅｂスクレイピングにより映画レビュー文を取得。ＣＳＶ形式でローカルＰＣ環境に保存。
②ＣＳＶファイルの文字コードをＵＴＦ８からＳＪＩＳに変換。
③Ｒを用いてＣＳＶファイルを読み込み。以下の手順はすべてＲを用いて実施。
④１レビューごとに別名のテキストファイルに保存。(全７３ファイル)
⑤ＲＭｅＣａｂライブラリのｄｏｃＭａｔｒｉｘ２関数によりテキストファイルの形態素解析を実行。
⑥品詞を「名詞、動詞、形容詞」に限定する。
⑦ｄｉｓｔ関数で各単語同士の距離を計算し、ｈｃｌｕｓｔ('ｗａｒｄ.Ｄ２')関数でクラスタリングを実行。
⑧ｇｇｄｅｎｄｒｏライブラリのｇｇｄｅｎｄｒｏｇｒａｍ関数で描画。

具体的方法２：具体的方法１の①～⑥は同様に実施。
①特異値分解により、単語のベクトル表現したものを３次元に圧縮。３×７３(単語(３次元圧縮)×テキストファイル(７３ファイル))の行列作成。
②ｒｇｌライブラリのｒｇｌ関数を用いて３次元プロット。

出力表示：



今回のデータは、データ取得元の映画.comのレビューページにおいて、５段階の評価が均等にバラついている作品を選んだ。
各々の評価はあくまでレビュアーの主観に基づくものだが、目安として大体５つのクラスターに分類されることが予想できる。
表示されているデンドログラムを見てみると、ざっくりとではあるが、大きく３つに大別(「良い・ふつう・悪い」)されているのがわかる。どの山が良い評価の塊でどの山が悪い評価の塊なのか(評価はレビュアーの主観によるものであるため、混在している場合もある)は、 テキストファイルの中身を参照しながら細かく見て行く必要がある。



試しに３次元空間上に各テキストファイルを付置してみたが(画像がボヤけてすみません)、部分的なクラスターはあっても全体的にバラついており、きれいに分類できていないことがわかる。
一つの決められた映画についてのレビューであっても、レビュアーが７３名いると、レビュー内容にバラつきが生じることは自然なことではある。文書分類には他の手法も多く存在するため、いろいろ試してみたいと思う。

実践７．文書の分類１

データの種類：質的データ(非構造化データ)
データの対象：ＢＯＮＥＳ原作「ＥＵＲＥＫＡ 交響詩篇エウレカセブン　ハイエボリューション」の映画レビュー文(全７３件)
具体的方法１：Ｐｙｔｈｏｎを使用。
　①Ｐｙｔｈｏｎを用いたＷｅｂスクレイピングにより映画レビュー文および評価点数を取得。ＣＳＶ形式でローカルＰＣ環境に保存。以下、Ｐｙｔｈｏｎを用いて実施。
　②学習済みのＤｏｃ２Ｖｅｃを'doc2vec/jawiki.doc2vec.dbow300d.tar/jawiki.doc2vec.dbow300d.model'からロード。
　③ＣＳＶファイルをデータフレームで読み込み。
　④データフレームのレビュー文を形態素解析したものをＤｏｃ２Ｖｅｃでベクトル化。
　⑤Ｋｍｅａｎｓ法でベクトル化したレビュー文を５つのクラスターに分類する。

具体的方法２：具体的方法１の①～③は同様に実施。以下、Ｐｙｔｈｏｎを使用。
　①データフレームのレビュー文を形態素解析したものをＴｆＩｄｆでベクトル化。
　②Ｋｍｅａｎｓ法でベクトル化したレビュー文を５つのクラスターに分類する。
　③ＴｆＩｄｆでベクトル化したものをＳＶＤ(特異値分解)で次元削減。
　　３１８８次元⇒１０００次元、３１８８次元⇒１０次元
　④Ｋｍｅａｎｓ法で次元削減したレビュー文を５つのクラスターに分類する。

出力表示：　表頭：０～４のグループ、表側：評価点数

Doc2Vec

評価点数	0	1	2	3	4
0.5	3	0	3	0	0
1	4	0	1	1	1
1.5	1	0	3	0	0
2	7	0	4	0	0
2.5	5	1	0	0	0
2.8	2	0	1	0	0
3	3	0	1	0	0
3.5	5	0	7	0	1
4	4	1	0	1	0
4.5	1	0	4	0	0
5	6	1	1	1	0

Tfldf

評価点数	0	1	2	3	4
0.5	2	1	0	0	3
1	1	0	0	0	6
1.5	2	0	0	0	2
2	3	0	1	0	7
2.5	2	0	0	1	3
2.8	1	0	0	0	2
3	2	0	0	0	2
3.5	4	0	0	0	9
4	3	0	0	0	3
4.5	0	0	0	0	5
5	5	0	0	0	4

Tfldf 1000次元

評価点数	0	1	2	3	4
0.5	5	0	1	0	0
1	7	0	0	0	0
1.5	4	0	0	0	0
2	9	0	0	1	1
2.5	6	0	0	0	0
2.8	3	0	0	0	0
3	4	0	0	0	0
3.5	12	1	0	0	0
4	5	1	0	0	0
4.5	5	0	0	0	0
5	7	1	0	1	0

Tfldf 10次元

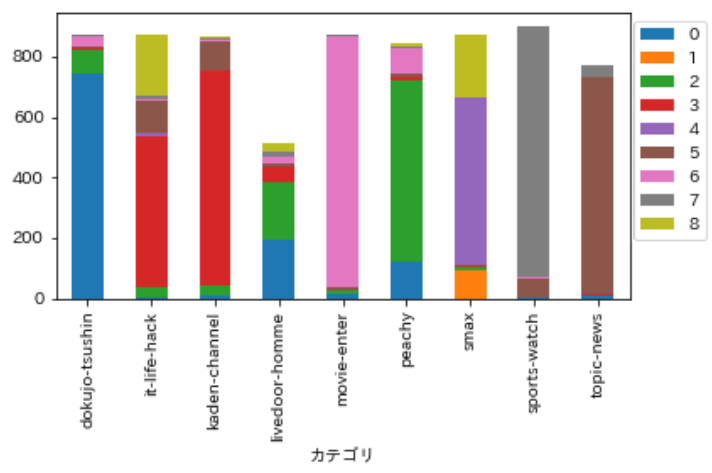
評価点数	0	1	2	3	4
0.5	0	0	3	0	3
1	0	0	6	0	1
1.5	0	0	2	0	2
2	1	1	5	2	2
2.5	1	0	3	0	2
2.8	1	0	2	0	0
3	1	1	1	0	1
3.5	3	1	6	0	3
4	3	0	2	0	1
4.5	0	0	3	0	2
5	4	0	4	1	0

どの方法もレビュアーの評価点数通りにきれいに分類されているとは到底言えない結果になった。
各評価点数のサンプル数を均等に増やして(1000個以上)、再度実施してみる必要がある。
十分なサンプル数で分類すれば、きれいに分類できなくても、評価点数の高いレビュー文、低いレビュー文の何かしらの傾向はわかるかもしれない。
また、ＴｆＩｄｆは単語の出現数のみにフォーカスしてベクトル化しており、単語間のつながりや意味については考慮されていない。したがって、サンプル数を増やしても評価点数通りに分類できるかは疑問である。

実践８．文書の分類２

データの種類：質的データ(非構造化データ)
データの対象：ライブドアニュースコーパスのテキスト文書(全７３７６件)
具体的方法：Ｐｙｔｈｏｎ使用。
　①ライブドアニュースコーパスのダウンロード
　②ダウンロードしたテキストファイルをＰｙｔｈｏｎを用いて「カテゴリ」と「内容」からなるデータフレームにする。
　　⇒「カテゴリ」：各テキストファイルのカテゴリ名。「内容」：各テキストファイルの中身、文章。
　③学習済みのＤｏｃ２Ｖｅｃを'doc2vec/jawiki.doc2vec.dbow300d.tar/jawiki.doc2vec.dbow300d.model'からロード。
　④データフレームの「内容」を形態素解析したものをＤｏｃ２Ｖｅｃでベクトル化。
　⑤Ｋｍｅａｎｓ法でベクトル化した「内容」を９つのクラスターに分類する。※「カテゴリ」が９つのため。

出力表示：



９つのカラーのため、大変見づらいですが、かなりよい精度で分類できているように見える。
livedoor-hommeはカテゴリ０と２で半々に分類されており、カテゴリ０、２の多くはそれぞれdokujyo-tsushin、peachyに分類されている。
実際にテキストの内容を見てみると、「キャリア、フィットネス」ではdokujyo-tsushinと内容が重なり、「ファッション、ライフスタイル」ではpeachyと内容が重なるように思う。
it-life-hackとkaden-channelは同じカテゴリ３に多く分類されているが、おそらく取り上げるテーマが近いために同じカテゴリに分類されたのではないだろうか。
サンプル数が十分であったのと、各カテゴリ(テーマ)に沿った文章が書かれているため、精度の高い分類ができたのだと思う。
ただし、Ｋｍｅａｎｓ法は本来教師なし学習に当たる機械学習アルゴリズムであるため、精度の評価には工夫が必要である。

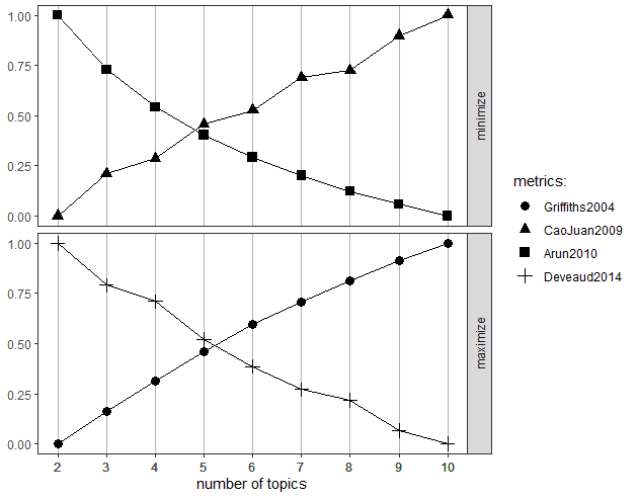
実践９．トピックモデル

データの種類：質的データ(非構造化データ)
データの対象：ライブドアニュースコーパスのdokujyo-tsushinのテキスト文書(全８７０件)
具体的方法１：
①事前にＰｙｔｈｏｎを用いてdokujyo-tsushinのテキスト文書をＵＴＦ８からＳＪＩＳに変換。
②ＲＭｅＣａｂライブラリのｄｏｃＤＦ関数によりテキストファイルの形態素解析を実行。以下、Ｒを使用。
③品詞大分類を「名詞、形容詞」に限定し、さらに品詞細分類を「一般、自立」に絞り込む。
④重複行を削除。
⑤ｔｉｄｙｒライブラリのｇａｔｈｅｒ関数により、データフレームを縦持ちにする。
⑥ｔｉｄｙｔｅｘｔライブラリのｃａｓｔ＿ｄｔｍ関数により、縦持ちデータを文書単語行列に変換。
⑦ｌｄａｔｕｎｉｎｇライブラリのｆｉｎｄＫ関数により、文書単語行列から最適なトピック数を推定。
⇒トピック数４と５でそれぞれ実行。
⑧再度、単語文書行列を作成したあと、ｔｍライブラリのＤＴＭ関数により、文書単語行列に変換。
⑨ｔｏｐｉｃｍｏｄｅｌｓライブラリのＬＤＡ関数により、文書単語行列とトピック数を指定してトピックを抽出。
⇒ｔｏｐｉｃｍｏｄｅｌｓライブラリによるＬＤＡ実行。
⑩ＤＴＭ関数により文書単語行列に変換したものをｌｄａライブラリのｄｔｍ２ｌｄａｆｏｒｍａｔ関数により、さらに変換。
⑪ｌｄａライブラリのｌｄａ．ｃｏｌｌａｐｓｅｄ．ｇｉｂｂｓ．ｓａｍｐｌｅｒ関数により、トピックを抽出。
⇒ｌｄａライブラリによるＬＤＡ実行。

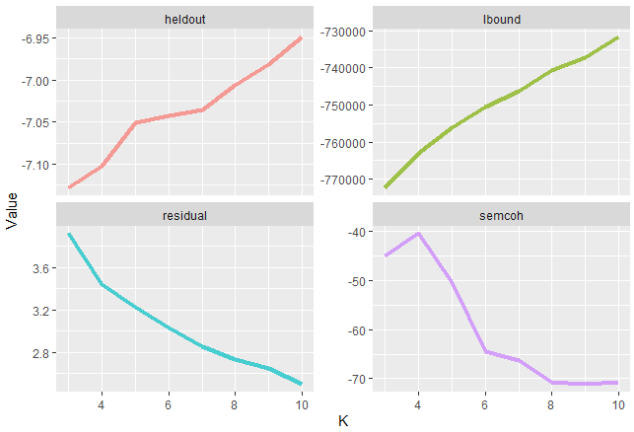
具体的方法２：具体的方法１の①～④を同様に実施。＋αでＰｙｔｈｏｎを用いてテキストファイル名に年代を追加して、ｓｊｉｓ形式で保存。これより下はＲを使用。
①データフレームの単語列を行に変えて、単語文書行列を作成。
②ｔｍライブラリのＤＴＭ関数で文書単語行列に変換。
③ｓｔｍライブラリのｒｅａｄＣｏｒｐｕｓ関数で文書単語行列を縦持ちデータに変換。
④テキストファイル名から「年代」を取得して、縦持ちデータに新たに追加。
⑤ｓｔｍライブラリのｓｅａｒｃｈＫ関数により、縦持ちデータから最適なトピック数を推定。
⇒トピック数４と５でそれぞれ実行。
⑥ｓｔｍライブラリのｓｔｍ関数により、線形モデルを実行。
説明変数：テキストファイル名、単語、トピック数、共変量：年代(Year)。
⑦ｓｔｍライブラリのｅｓｔｉｍａｔｅＥｆｆｅｃｔ関数により、各トピックと年代(Year)との関係を確認。
⑧ｓｔｍライブラリのｓｔｍ関数により、非線形モデルを実行。
説明変数：テキストファイル名、単語、トピック数、共変量：年代(Year)(スプライン指定)。
⑨ｓｔｍライブラリのｅｓｔｉｍａｔｅＥｆｆｅｃｔ関数により、各トピックと年代(Year)との関係を確認。

---

(1)トピック数の推定



ｌｄａｔｕｎｉｎｇライブラリのｆｉｎｄＫ関数によるトピック数推定の結果、それぞれの指標がクロスするときのトピック数を推定値とする。推定値は５個であるが、今回はｓｅａｒｃｈＫ関数によるトピック数推定の結果を考慮して、４個と５個の２つの場合で実施した。



ｓｔｍライブラリのｓｅａｒｃｈＫ関数によるトピック数推定の結果(横軸Ｋ：トピック数)、ｈｅｌｄｏｕｔはトピック数４から５の間における値の変化が大きく、ｒｅｓｉｄｕａｌはトピック数３から４の間における値の変化が大きい。ｓｅｍｃｏｈはトピック数４で最大値になっている。
今回はｆｉｎｄＫ関数の結果も考慮して、４個と５個の２つの場合で実施した。

(2)トピックの抽出結果(トピック数４)

(a)ｔｏｐｉｃｍｏｄｅｌｓライブラリのｌｄａ関数によるトピック抽出結果

```
     Topic 1   Topic 2 Topic 3 Topic 4
 [1,] "女"      "人"    "多い"  "女性"
 [2,] "女子"    "自分"  "女性"  "人"
 [3,] "自転車"  "女"    "人"    "男性"
 [4,] "映画"    "ない"  "月"    "女"
 [5,] "女性"    "女性"  "いい"  "ない"
 [6,] "男"      "男性"  "ない"  "多い"
 [7,] "人"      "相手"  "女"    "自分"
 [8,] "ない"    "多い"  "野菜"  "気"
 [9,] "漫画"    "いい"  "香り"  "モテ"
[10,] "自分"    "男"    "商品"  "美容"
```

⇒上位１０単語から読み取ると、、
Topic 1：女性の「趣味」についてのトピック
Topic 2：女性の「恋愛」についてのトピック
Topic 3：女性の「健康」についてのトピック
Topic 4：女性の「美容」についてのトピック

このような感じだろうか。

(b)ｌｄａライブラリのｌｄａ．ｃｏｌｌａｐｓｅｄ．ｇｉｂｂｓ．ｓａｍｐｌｅｒ関数によるトピック抽出結果

```
      [,1]       [,2]    [,3]      [,4]
 [1,] "自転車"   "男"    "自分"    "人"
 [2,] "漫画"     "男性"  "占い"    "会社"
 [3,] "肌"       "自分"  "映画"    "自分"
 [4,] "hoge"     "モテ"  "相手"    "夫婦"
 [5,] "夏"       "婚"    "恋"      "社員"
 [6,] "マンガ"   "女"    "心"      "男性"
 [7,] "女子"     "独身"  "美容"    "友人"
 [8,] "商品"     "友達"  "男性"    "夫"
 [9,] "by"       "相手"  "気持ち"  "両親"
[10,] "野菜"     "女子"  "男"      "仮名"
```

⇒左から順にTopic1, 2, 3, 4とする。
Topic 1：女性の「趣味、遊び」についてのトピック
Topic 2：女性の「結婚」についてのトピック
Topic 3：女性の「恋愛」についてのトピック
Topic 4：女性の「夫」についてのトピック

このような感じだろうか。

(c)ｓｔｍライブラリのｓｔｍ関数によるトピック抽出結果とｅｓｔｉｍａｔｅＥｆｆｅｃｔ関数による各トピックと年代の関係

```
Topic 1 Top Words:
  Highest Prob: 女性, 自転車, 多い, 人, ない, 肌, 美容
  FREX: 自転車, ドロンジョーヌ, ホルモン, ブラ, ヨネスケ, 腸, 花粉
  Lift: EICO, IKEA, MISSION, イミダペプチド, エクスチェンジ, オーガニック, カリウム
  Score: 自転車, ドロンジョーヌ, ヨネスケ, 菌, 椀, 花粉, スッピン
Topic 2 Top Words:
  Highest Prob: 人, 自分, ない, 占い, 多い, いい, 月
  FREX: 占い, nifty, カード, モード, Twitter, アプリ, 結納
  Lift: アウトレットパーク, メンタルトレーニング, 開き, 歳暮, 消防, 人恋しい, 道徳
  Score: モード, 占い, nifty, 占い師, カード, タロット, マネー
Topic 3 Top Words:
  Highest Prob: 女, 女子, 映画, 自分, 女性, 人, ない
  FREX: 主人公, 少女, ロードショー, 映画, ストーリー, ヒロイン, バイキング
  Lift: バイキング, ラブストーリー, 原作, BeeTV, CIA, CLS, Company
  Score: 映画, 主人公, 少女, 作品, マンガ, 男, ロードショー
Topic 4 Top Words:
  Highest Prob: 女, 人, 女性, 男性, 自分, ない, 多い
  FREX: ボーダー, 仮名, 先輩, 婚, 既婚, 嘘, 独身
  Lift: PACS, SM, VS, Zune, topics, あて, あり方
  Score: 男, 仮名, 夫婦, 既婚, ボーダー, 婚, 娘
```

⇒Topic 1:女性の「美容、健康」についてのトピック
Topic 2:女性の「占い」についてのトピック
Topic 3：女性の「映画」についてのトピック
Topic 4：女性の「恋愛、結婚」についてのトピック

４つの指標で頻出単語が出力されるので、ある程度の予想はできる。

各指標の直感的意味
Highest Prob：単語の出現確率
FREX：特徴的な単語
Lift：特に現れやすい単語
Score：TF-IDFに近い指標

各トピックと年代(Year)との関係(線形モデル)

Topic 1:

```
Coefficients:
     Estimate Std. Error t value Pr(>|t|)
(Intercept) 18.748578 34.411466 0.545  0.586
Year    -0.009214 0.017110 -0.538  0.590
```

Topic 2:

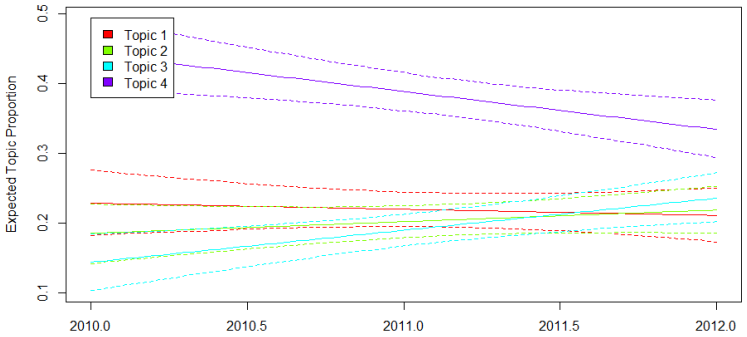
```
Coefficients:
     Estimate Std. Error t value Pr(>|t|)
(Intercept) -35.32268 31.03839 -1.138  0.255
Year     0.01766  0.01543 1.145  0.253
```

Topic 3:

```
Coefficients:
     Estimate Std. Error t value Pr(>|t|)
(Intercept) -92.88481 30.65715 -3.030 0.00252 **
Year     0.04628  0.01524 3.036 0.00247 **
---
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Topic 4:

```
Coefficients:
     Estimate Std. Error t value Pr(>|t|)
(Intercept) 108.85972 37.37555 2.913 0.00368 **
Year    -0.05394  0.01858 -2.902 0.00380 **
---
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```



⇒Topic 3と4について、「年代(Year)」の影響が統計的に有意という結果が出ている。つまり、Topic 3の「映画」、Topic 4の「恋愛、結婚」のトピックは年代によって話題に上る割合が有意に変化する、言い換えると話題性に富むものであるということが言える。

グラフを見てみると、Topic 3は２０１０年～２０１２年にかけて右上がりになっており、Topic4は反対に右下がりになっている。
Topic2も右上がりになっているが、Topic３と比べると緩やかである。

各トピックと年代(Year)との関係(非線形モデル)

Topic 1:

```
Coefficients:
     Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.22821  0.02767 8.247 6.02e-16 ***
s(Year, 4)1 1.18683 99.42729 0.012  0.990
s(Year, 4)2 -0.03766 62.56445 -0.001  1.000
s(Year, 4)3 -1.14322 98.59354 -0.012  0.991
s(Year, 4)4 -0.01813  0.03494 -0.519  0.604
---
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Topic 2:

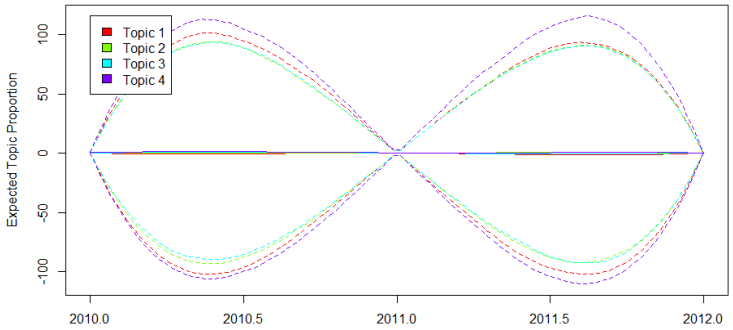
```
Coefficients:
     Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.18499  0.02524 7.328 5.36e-13 ***
s(Year, 4)1 1.70074 90.29597 0.019  0.985
s(Year, 4)2 -0.41075 57.32973 -0.007  0.994
s(Year, 4)3 -0.82117 90.03507 -0.009  0.993
s(Year, 4)4 0.03465  0.03150 1.100  0.272
---
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Topic 3:

```
Coefficients:
     Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.11887  0.02408 4.936 9.54e-07 ***
s(Year, 4)1 -2.19580 87.40543 -0.025 0.979963
s(Year, 4)2 0.69997 55.40148 0.013 0.989922
s(Year, 4)3 1.17533 87.18752 0.013 0.989248
s(Year, 4)4 0.10196  0.03071 3.320 0.000939 ***
---
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Topic 4:

```
Coefficients:
     Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.46768  0.02997 15.607 < 2e-16 ***
s(Year, 4)1 -2.99056 107.12880 -0.028 0.97774
s(Year, 4)2 0.83403 68.27795 0.012 0.99026
s(Year, 4)3 0.91851 107.97578 0.009 0.99321
s(Year, 4)4 -0.11773  0.03749 -3.140 0.00175 **
---
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```



⇒線形モデルと同様にTopic 3と4には何らかの時間的要因が関係していそうであるが、グラフを見る限り、どのトピックも同じように変化の推移を辿っているように見える。ただ、Topic4は他と比べると少しではあるが、変化が大きい。

５年～１０年またはそれ以上の長い期間であれば、顕著な変化があるであろうが、年代が２０１０年～２０１２年の３年間という短い期間では、４つのトピックに大した変化は起きないのかもしれない。

(3)トピックの抽出結果(トピック数５)

(a)ｔｏｐｉｃｍｏｄｅｌｓライブラリのｌｄａ関数によるトピック抽出結果

	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
[1,]	"女性"	"自分"	"女"	"女"	"人"
[2,]	"自転車"	"人"	"人"	"男"	"女性"
[3,]	"多い"	"映画"	"女性"	"ない"	"自分"
[4,]	"肌"	"占い"	"男性"	"人"	"多い"
[5,]	"美容"	"ない"	"ない"	"自分"	"男性"
[6,]	"人"	"女"	"自分"	"漫画"	"ない"
[7,]	"女"	"女性"	"男"	"母"	"相手"
[8,]	"体"	"恋"	"いい"	"女子"	"会社"
[9,]	"気"	"気持ち"	"友達"	"女性"	"女"
[10,]	"ない"	"相手"	"多い"	"娘"	"いい"

⇒上位１０単語から読み取ると、、
Topic 1：女性の「美容、健康」についてのトピック
Topic 2：女性の「恋占い、恋愛映画」についてのトピック
Topic 3：女性の「恋愛」についてのトピック
Topic 4：女性の「結婚」についてのトピック
Topic 5:Topic 3と多くの単語が重なるが、女性の「社内恋愛」についてのトピック

このような感じだろうか。

(b)ｌｄａライブラリのｌｄａ．ｃｏｌｌａｐｓｅｄ．ｇｉｂｂｓ．ｓａｍｐｌｅｒ関数によるトピック抽出結果

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	"夫"	"自転車"	"男"	"相手"	"会社"
[2,]	"hoge"	"肌"	"映画"	"男性"	"香り"
[3,]	"漫画"	"美容"	"女性"	"占い"	"ネット"
[4,]	"クリスマス"	"女性"	"男性"	"自分"	"情報"
[5,]	"父"	"体"	"女"	"女性"	"家"
[6,]	"母"	"野菜"	"婚"	"友達"	"商品"
[7,]	"娘"	"髪"	"恋"	"男"	"店"
[8,]	"結婚式"	"夏"	"活"	"ない"	"ブログ"
[9,]	"妹"	"お腹"	"相手"	"女"	"バブル"
[10,]	"子ども"	"ケア"	"モテ"	"会社"	"おばさん"

⇒左から順にTopic1, 2, 3, 4,5とする。
Topic 1：女性の「夫」についてのトピック
Topic 2：女性の「美容、健康」についてのトピック
Topic 3：女性の「恋活、婚活」についてのトピック
Topic 4：女性の「恋占い」についてのトピック
Topic 5：女性の「ネット情報」についてのトピック

Topic 5がよくわからなかったが、こんな感じだろうか。

(c)ｓｔｍライブラリのｓｔｍ関数によるトピック抽出結果とｅｓｔｉｍａｔｅＥｆｆｅｃｔ関数による各トピックと年代の関係

```
Topic 1 Top Words:
  Highest Prob: 女性, 美容, 多い, 女, 野菜, 肌, ない
  FREX: ヨネスケ, ヨーグルト, ごはん, あやしい, 椀, コラーゲン, みそ汁
  Lift: MISSION, あさり, きなこ, ごぼう, すそ, ほうれん草, まつ毛
  Score: ヨネスケ, 椀, みそ汁, ブラ, 菌, スッピン, 弁当
Topic 2 Top Words:
  Highest Prob: 自転車, 女子, 映画, 月, 女性, 女, T
  FREX: 自転車, ドロンジョーヌ, モード, エイ, ショウ, ロードショー, セール
  Lift: COME, D, Flowers, Over, PHONE, TRUE, blog
  Score: モード, 自転車, ドロンジョーヌ, エイ, ショウ, 作品, キャンペーン
Topic 3 Top Words:
  Highest Prob: 女, 人, 女性, 自分, 男性, ない, 男
  FREX: 隙, モテ, 仮名, 二股, 彼氏, 男, グチ
  Lift: PACS, Sony, VS, YOU, うそ, おばあさん, お構い
  Score: 相手, 男, モテ, 仮名, 年上, 隙, 年下
Topic 4 Top Words:
  Highest Prob: 自分, 人, ない, 多い, 占い, 女, 母
  FREX: 占い, nifty, 家計, 間取り, スキー, 老人, 結納
  Lift: IKEA, Zune, しきたり, のし紙, イノベーション, ウェディング, ガソリン
  Score: 占い, nifty, 間取り, 家計, 占い師, 収入, 子ども
Topic 5 Top Words:
  Highest Prob: 人, ない, 多い, 気, 女性, 夏, 漫画
  FREX: め, ボーダー, うえ, 痔, ストッキング, 八重歯, 匂い
  Lift: ストッキング, EICO, PONT, SDN, V, iPhone, presented
  Score: 痔, うえ, 八重歯, ボーダー, 抱っこ, 花粉, ストッキング
```

⇒Topic 1：女性の「健康、美容」についてのトピック
Topic 2：女性の「映画」についてのトピック
Topic 3：女性の「恋愛」についてのトピック
Topic 4：女性の「結婚生活」についてのトピック
Topic 5：不明..

Topic 5はわかりませんでした。

⇓⇓⇓

上記の結果からトピック数は４個が妥当ではないか。
４つのトピックスは、「健康、美容」、「恋愛、結婚」、「趣味、遊び」、「占い」に大別できると考える。