青空文庫の小説を使った作家分類

概要

- 青空文庫から、小説のデータを収集してクラスタリングを行った。
- クラスタリング手法は k-means を試した。
- 特徴量として、以下のものをそれぞれ試した。

1. 品詞情報
2. TF-IDF
3. TF-IDF + 特異値分解による次元圧縮

- 作家の名前をラベルとして、クラスタリング結果を考察した。
- 発表用のスライドの後に、より詳細な捕捉スライドあり。

目的

- データ収集から分析を通して、自然言語によるクラスタリングの練習とする。
- 複数種類の特徴量を用いて小説データをクラスタリングすることで、各作家の特徴がクラスタに現れるか、現れた場合どのような特徴があるか考察する。

データの概要

- データの収集・前処理はPythonで行った。
- 青空文庫(https://www.aozora.gr.jp/)から、有名な作家の小説を適当に収集
    - 夏目漱石、芥川竜之介、宮沢賢治の3名の小説を収集
- 1つのデータが5000文字前後となるように、長い小説は分割して調整した。
- 各作家 150 ずつデータを集めた。(データは全部で150×3=450個)

データの特徴量抽出について

- 形態素解析機として、Megagon LabsのGiNZA+spacyを用いた。
  - https://megagonlabs.github.io/ginza/
- 品詞情報: 片仮名率、名詞率、句読点の数等 7次元のデータ
  - 参考: 文体診断ロゴーン http://logoon.org/
- TF-IDF: 扱う単語の数の次元のデータ、今回はおよそ 34000次元
  - 単語の出現頻度をベースにしたベクトル化手法
  - その文書の中でより多く登場するが、他の文書にあまり登場しないような単語に大きな値を割り当てる。
- TF-IDF + 次元圧縮: 特異値分解でデータの次元を圧縮することで、意味のある特徴量を得られる場合がある

品詞情報を用いたクラスタリング

表 1　品詞情報を用いたクラスタリング

	k-menas	
	標準化なし	標準化あり
Entropy	0.6807983	0.6128075
Purity	0.6666667	0.7666667

Entropy、Purity両方について標準化を施した方が良い結果が得られている。
7次元のデータのうち、ひらがな率・カタカナ率は文字を、名詞率・その他は単語を数えているのでスケールが違う。

品詞情報を用いたクラスタリング

表 2　品詞情報を用いたクラスタリング
混同行列: 標準化なし

	夏目漱石	芥川竜之介	宮沢賢治
1	37	80	7
2	106	68	29
3	7	2	114

表 3　品詞情報を用いたクラスタリング
混同行列: 標準化あり

	夏目漱石	芥川竜之介	宮沢賢治
1	5	20	124
2	32	108	10
3	113	22	16

標準化する前はクラスタ1, 2に均等に芥川竜之介の小説が分類されている。

黒: クラスタ1 (宮沢 多い)
赤: クラスタ2 (芥川 多い)
緑: クラスタ3 (夏目 多い)

- ひらがな率は黒>緑>赤
- カタカナ率はクラスタの属性に関係なさそう

黒: クラスタ1 (宮沢 多い)
赤: クラスタ2 (芥川 多い)
緑: クラスタ3 (夏目 多い)

- 赤は名詞率が高い
- 緑は動詞率が高い
- 緑は補助記号が少ない

TF-IDFを用いたクラスタリング

表 4　TF-IDF を用いたクラスタリング

	k-menas					
	次元圧縮なし		16 次元に圧縮		3 次元に圧縮	
	標準化なし	標準化あり	標準化なし	標準化あり	標準化なし	標準化あり
Entropy	0.6995073	0.979904	0.9108703	0.7184841	0.7334936	0.8181523
Purity	0.62	0.3533333	0.4644444	0.6044444	0.6111111	0.5755556

Entropy, Purityともに、品詞情報を使った場合より悪い。
次元圧縮なしかつ標準化なしが一番良くなった。
特異値が大きい成分は作家の文体等に関係ないものが抽出されている?

TF-IDFを用いたクラスタリング

表 5　TF-IDF を用いたクラスタリング
混同行列: 次元圧縮なし, 標準化なし

	夏目漱石	芥川竜之介	宮沢賢治
1	3	24	96
2	133	76	5
3	14	50	49

表 6　TF-IDF を用いたクラスタリング
混同行列: 次元圧縮なし, 標準化あり

	夏目漱石	芥川竜之介	宮沢賢治
1	150	145	146
2	0	0	4
3	0	5	0

標準化により、ほとんどクラスタ分けができなくなっている。
標準化がTF-IDFの単語に重みを付けるという良さを完全に潰している。
標準化なしのについて、クラスタ1は宮沢賢治が多い。
-> 宮沢賢治は他の作家と違って、使っている語彙に特徴がありそう。

TF-IDF+次元圧縮を用いたクラスタリング

表 7　TF-IDF を用いたクラスタリング
混同行列: 16 次元に圧縮, 標準化なし

	夏目漱石	芥川竜之介	宮沢賢治
1	0	0	23
2	136	100	83
3	14	50	44

表 8　TF-IDF を用いたクラスタリング
混同行列: 16 次元に圧縮, 標準化あり

	夏目漱石	芥川竜之介	宮沢賢治
1	9	108	104
2	0	0	23
3	141	42	23

どちらも、宮沢賢治の小説だけのクラスタがある。
この場合は標準化した方がいい結果になった。
標準化ありだと、クラスタ3に夏目漱石が多いクラスタが出現している。

TF-IDF+次元圧縮を用いたクラスタリング

表 9　TF-IDF を用いたクラスタリング
混同行列: 3 次元に圧縮, 標準化なし

	夏目漱石	芥川竜之介	宮沢賢治
1	129	60	5
2	14	50	49
3	7	40	96

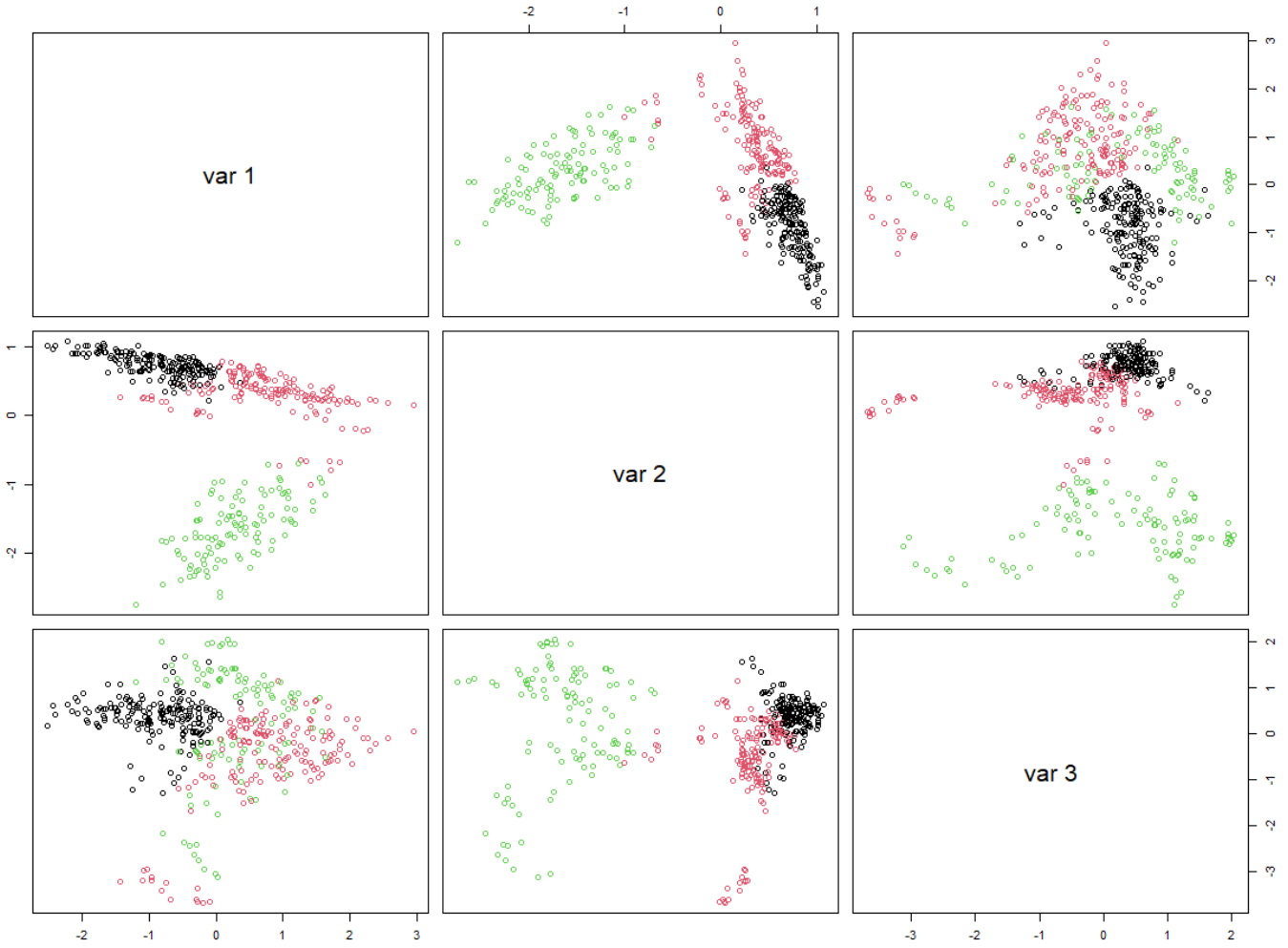
表 10　TF-IDF を用いたクラスタリング
混同行列: 3 次元に圧縮, 標準化あり

	夏目漱石	芥川竜之介	宮沢賢治
1	120	45	15
2	16	55	89
3	14	50	46

この場合は標準化しないほうがいい。
標準化の有無に関わらず芥川竜之介がどのクラスタにも均等に属している。

黒: クラスタ1 (夏目 多い)
赤: クラスタ2 (宮沢・芥川 多い)
緑: クラスタ3 (宮沢 多い)

2つ目の変数についてクラスタ3が綺麗に分離できている。
-> 宮沢賢治の特徴が抽出されている?



クラスタリングの結果まとめ

- Entropy, Purityについて、品詞情報によるクラスタリングが最も良かった。その次に、TF-IDFで次元圧縮も標準化もしない場合が良かった。
- 宮沢賢治がひらがな多め、芥川竜之介が名詞が多め、夏目漱石は1文が長い傾向がある、という風に作家の文体の特徴をある程度読み取ることができた。
- 全体として
  - 宮沢賢治の多いクラスタが出現しやすい -> 文章が特徴的
  - 芥川竜之介の多いクラスタが出現しにくい -> 今回の方法では特徴が上手く捉えられなかった。

今後やりたいこと

- 他の作家だとどのような特徴が出るのか
- 同じ作家の別の作品だとどのようにクラスタが出てくるか
- 他のクラスタリング手法も試してみる
- ニューラルネットベースのベクトル化手法を試したい

まとめ

- 夏目漱石、芥川竜之介、宮沢賢治の小説についてクラスタリングを行った。
- 特徴量として、品詞情報を用いた方が、TF-IDFを用いた場合よりもEntropy, Purityの観点で良くなった。
- クラスタリングの結果から、各作家の文章の特徴が読み取れた。
- 今回の方法では、クラスタとして、まとまりやすい作家とまとまりにくい作家がいた。
- 使用したデータとスクリプトはGoogle Driveにあげておきます。
  - URL: https://drive.google.com/file/d/1bQX27VB2V8-qqKKwVBwx2mgoc5aHI6a3/view?usp=sharing

以降のスライドは捕捉

捕捉: データの収集について

- データは青空文庫APIで取得
  - 参考ブログ: https://qiita.com/ksato9700/items/626cc82c007ba8337034

1. 宮沢賢治、芥川竜之介、夏目漱石の小説を全てダウンロード
2. ダウンロードした小説からタイトルや書式記号等を削除し、本文のみを抽出
3. 文字数が2000以下 or 句読点が10以下のものは短すぎるとして排除
4. データ1つ当たりの文字数を揃えるために、長い小説は5000文字毎に別のデータとして分割。(分割は句読点の位置で行った。小説として意味のある区切りではない)
5. 1クラスあたりのデータ数を揃えるために、1クラスあたり150個ランダムに抽出

捕捉: 形態素解析機について

使用した形態素解析機: **GiNZA**

- spacyというフレームワークの元に作成されている形態素解析機
- MeCab等に比べて重いが、精度が高い (はず)
- 使用したバージョンは v5.1、モデルは `ja_ginza`
- GiNZAは `ja_ginza` の他に `ja_ginza_electra` (Transformerモデル)も使用できる
- 公式サイト: https://megagonlabs.github.io/ginza/

捕捉: 品詞情報について

- 使用した特徴量
  - ひらがな率 - ひらがな文字数 / 文書の文字数
  - カタカナ率 - カタカナ文字数 / 文書の文字数
  - 名詞率 - 名詞の数 / 文書の形態素数 (単語の数のようなもの)
  - 動詞率 - 動詞の数 / 文書の形態素数
  - 助詞率 - 助詞の数 / 文書の形態素数
  - 助動詞率 - 助動詞の数 / 文書の形態素数
  - 補助記号率 - 補助記号の数 / 文書の形態素数 (補助記号は、。「」など 1文の長さの特徴量でもある)
- 参考: 品詞情報による作家判定の前例に、文体診断ロゴーン(http://logoon.org/)がある。

捕捉: TF-IDF について

TF-IDFは文書をベクトル化するための基礎的な手法である。
各文書データを 全文書の単語の種類数次元(今回は34000次元)のベクトルに変換する。
各要素はその文書に含まれる単語の重要度で、その文書に多く含まれていて、かつ、
他の文書にあまり含まれていないような単語の数値が高くなる。
計算式は以下の通り。

$$tfidf_{i,k} = \frac{\text{文書}_k\text{中の単語}_i\text{の出現回数}}{\text{全ての文書中の単語}_i\text{の出現回収}} \cdot \log \frac{\text{総文書数}}{\text{単語}_i\text{を含む文書の数}}$$

今回は、scikit-learnの機能を使ってTF-IDFを計算した。

- 参考: https://qiita.com/ground0state/items/155b77f4c07e1a509a14

また、以下の単語は固有名詞であったり、IDF値が0に近くなるので除外した。

- 1文字の単語, 15文字以上の単語, 補助記号, 助詞, 助動詞, 英数字を含む単語