2021年1月20日提出

卒業論文

LDAトピックモデルと履修成績を用いた履修レコメンドシステム

学籍番号：1686592c

氏名：宮崎仁弥

専攻分野：情報コミュニケーション論講座

ITコミュニケーション論コース

指導教員：村尾　元教授

副指導教員：康　敏教授

目次

1. はじめに

1-1.　研究の背景と目的

現在大学生が受講できる授業の数は大変多くなっている。例えば、神戸大学国際人間科学部の2020年に開講された授業数は約1500件である。そのため、学生は自分の趣味・嗜好に合わせて履修することが可能になっており、授業選択の自由度が高くなっている。しかし授業が多様化した反面、履修計画を建てることは煩雑化した。数ある科目の中からシラバスを確認し、自分が興味を持てる授業なのかなどの判断をしながら履修する科目を探し出すことはなかなか時間がかかる。神戸大学国際人間科学部グローバル文化学科の学生の履修科目とその成績のデータをもとに、科目選択の効率化や自分の知らなかった得意・興味のある科目の発見を促すことが本研究の目的である。

1-2.　本論文の構成

本論文の構成は次のようになっている。第2章では、先行研究を紹介する。第3章では、本研究の提案システムの全体の手順について記す。第4章では、検証とその考察について述べる。第5章では、まとめと今後の課題・展望について述べる。

2. 関連研究

本研究と同様に大学生を対象に科目を推薦するシステムに関する既存研究が存在する。竹森ら[1]は、学部新入生を対象に教養科目を推薦するシステムの設計を行った。各科目のシラバス内の「科目名」、「授業目標」、「授業計画」から名詞のみを抽出したものを科目の特徴語と呼び、doc2vec を用いて科目ベクトルを取得する。その科目ベクトルに対してウォード法を用いたクラスタリングを行った。クラスタごとに科目特徴語を TF-IDF で重み付けし、重みの大きい単語をワードクラウドで表示する。その次に高校主要科目に該当する大学科目のシラバスから doc2vec を用いて、5 科目ごとに科目ベクトルを作成する。各クラスタに属する各科目に対し、高校科目ベクトルとの類似度を計算し、クラスタごとに平均化、0~5 の値で正規化し、レーダーチャートに表し可視化する。学生はワードクラウドとレーダーチャートを見てクラスタを選択し、そのクラスタの科目を学生にレコメンドする。科目をクラスタリングしているという点で共通しているが、本研究では成績を用いているという点で異なる。

西森ら[2]は TF-IDF とコサイン類似度を用いて科目間の類似度を求めた。履修する科目と履修済みの科目の類似度に直接 GPA をかけることで、科目の成績を推定する。履修済みの科目に対して、推定を行ったところ、無作為に推定した場合より絶対平均誤差が低いことを明らかにした。本研究では類似度に直接 GPA を掛け合わせるのではなく、トピックベクトルに GPA を掛け合わせている

3. 全体の手順

全体の手順を図 1 に示す。シラバスのデータから授業のトピックベクトルを求める。学生の既履修の授業の成績と授業のトピックベクトルから嗜好性ベクトルを算出する。未履修の授業と嗜好性ベクトルのコサイン類似度を計算し、コサイン類似度が大きい科目を学生にレコメンドする。3-1 で「トピックベクトルの算出」、3-2 で「嗜好性ベクトルの算出」、3-3 で「レコメンド」についてそれぞれ説明する。

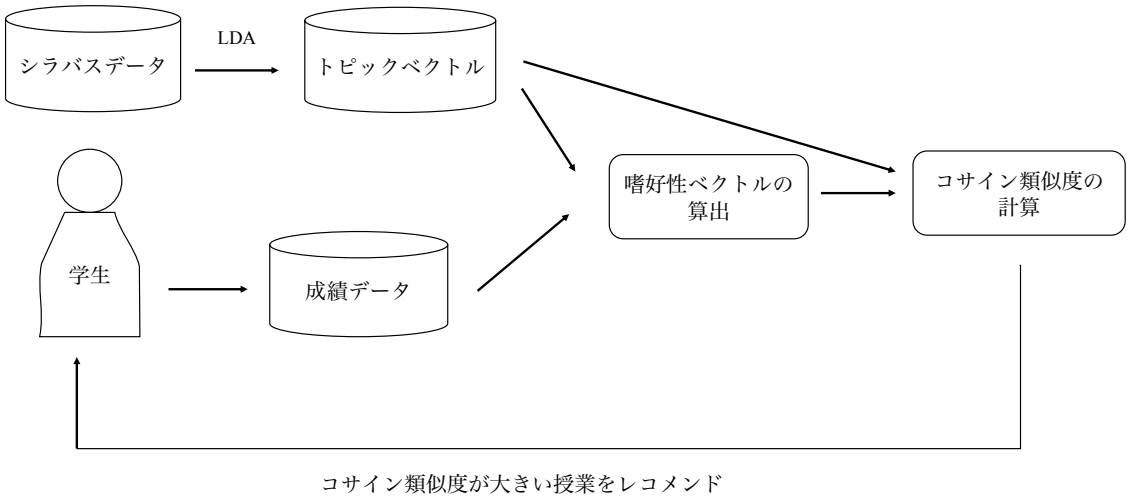

図 1.　全体の手順

3-1.　トピックベクトルの算出

3-1-1.　トピックベクトルの算出

分析対象のシラバスの中から「科目名」、「時間割コード」、「開講年度」、「授業のテーマ」をスクレイピングする。シラバスのテキストデータに TF-IDF を用いてベクトル化し、LDA トピックモデルを生成、科目のトピックベクトルを算出する。表 1 は科目とそのトピック分布の一例である。ここではトピック数は 6 としている。トピック分布の例の数値は、各科目が 6 つのトピックにどれくらい属しているかを表している。現代 IT 入門 A はトピック E の

値が一番大きいため、トピック E に最も属していると考えられ、グローバル社会動態発展演習 A はトピック D 以外の値が 0 であるため、トピック D にのみ属していると考えられる。

表 1.　科目とトピック分布の一例（小数点以下 4 位以下は切り捨て）

科目名	トピック					
	A	B	C	D	E	F
現代 IT 入門 A	0.041	0	0.179	0.101	0.672	0
グローバル社会動態発展演習 A	0	0	0	0.977	0	0
国際関係論 A	0	0.032	0.448	0	0.509	0

3-1-2.　TF-IDF

TF-IDF は、文書内に出現する単語について TF（出現頻度）と IDF（逆文書頻度）からその単語の重要度を求める手法である。TF とは Term Frequency の略である。これは各文書での単語の出現頻度を意味する。関数 f を出現頻度を求める関数とし、文書d_jにおける単語t_iの TF を表したものが(1)式である。

$$\mathrm{tf}(\mathrm{t_i},\ \mathrm{d_j}) = \frac{\text{文書}\mathrm{d_j}\text{内の単語}\mathrm{t_i}\text{の出現回数}}{\text{文書}\mathrm{d_j}\text{内の全ての単語の出現回数の和}} = \frac{\mathrm{f}(\mathrm{t_i},\ \mathrm{d_j})}{\sum_{\mathrm{t_k}\in\mathrm{d_j}} \mathrm{f}(\mathrm{t_k},\ \mathrm{d_j})} \tag{1}$$

しかし、TF のみではどの文書にも現れる単語の値も大きくなってしまう。そのような単語は文書の特徴を表しているとは考えにくい。そこで用いられるのが IDF である。IDF は Inverse Document Frequency の略である。これはある単語が含まれる文書の割合の逆数を表す。その単語の出現する文書の数が少ないほどこの値は大きくなる。ある文書集合における単語について考える場合、$\mathrm{df}(\mathrm{t_i})$を単語が出現する文書数とすると、IDF は(2)式から求めら

れる。

$$\mathrm{idf}(t_i) = \log\left(\frac{\text{総文書数}}{\text{単語}t_i\text{が出現する文書数}}\right) = \log\left(\frac{N}{\mathrm{df}(t_i)}\right) \quad (2)$$

TF-IDF は TF 値と IDF 値を掛け合わせる以下の(3)式で求められる。

$$\mathrm{tfidf}(t_i, d_j) = \mathrm{tf}(t_i, d_j) \cdot \mathrm{idf}(t_i) \quad (3)$$

それにより、ある文書での出現回数は多いが、他の文書にはあまり出現しない単語の TF-IDF 値は大きくなる。TF-IDF 値が大きい単語ほどその文書の特徴を表していると言える。

3-1-3. LDA トピックモデル

LDA トピックモデルは、文書の確率的生成モデルとして提案された。トピックとは文書における主題のことである。LDA では一つの文書に複数のトピックが存在すると仮定し、そのトピックの分布を離散分布としてモデル化する[3]。本研究ではシラバスの各授業のトピックの分布を LDA を使って求めている。本研究ではこのトピック分布をトピックベクトルとみなし計算を行う。

なお、LDA トピックモデルでは自動的にトピック数を決定できないので手動でトピック数を決定する必要がある。その際の指標となるのが Perplexity と Coherence である。

3-1-4. Perplexity

Perplexity は平均分岐数とも訳され、トピックモデルによる予測精度を表している。Perplexity は小さいほどそのモデルの性能が良いことを示す。

3-1-5. Coherence

Coherence は、トピックの質を表す。意味の近い単語が集まっているトピックをより多く抽出できる手法が良いモデルであるという観点から Coherence の研究は行われている。トピックの質はトピック中の単語感類似度の平均値から求められる、トピック全体の Coherence が高ければ良いモデルである。

3-2. 嗜好性ベクトルの算出

ここでは学生の成績データを用いる。学生の成績データ内にある授業の年度と時間割コードによりシラバスファイルから検索し、該当する授業を探しだし、その授業のトピックベクトルを取得する。授業の嗜好性ベクトルの計算の例を表2に示す。トピックベクトルそれぞれの値に成績データの値をかけ、重みづけを行う。各授業のトピックベクトルに対してこの計算を行い、トピックごとに値を合計する。このようにして合計されたベクトルは学生の各トピックに対する嗜好性を表す嗜好性ベクトルである。例えば、表4の学生はトピックBの値が6.8と一番高いためトピックBが得意であると考えられ、反対に値が一番小さいトピックCは不得意であると考えられる。

表2. 嗜好性ベクトルの計算の例

科目	トピックベクトル			成績	トピックベクトル×成績		
	A	B	C		A	B	C
English	0.1	0.2	0.9	2	0.2	0.8	1.8
情報学	0.8	0.6	0	4	3.2	2.4	0
社会学	0.5	0.9	0.1	4	2	3.6	0.4
嗜好性ベクトル（トピックごとに合計）					5.4	6.8	2.2

3-3. レコメンド

3-3-1. レコメンド方法

授業のトピックベクトルと学生の嗜好性ベクトルの類似度をコサイン類似度を用いて求める。コサイン類似度が大きい授業ほど嗜好性に合っているため、その学生におすすめである。コサイン類似度が大きい授業から降順に並べ、上位の授業を学生にレコメンドする。

3-3-2. コサイン類似度

コサイン類似度はベクトル空間において、2 本のベクトルがなす角度を表す指標である。以下の(4)式で求められる。1 に近ければ類似しており、0 に近ければ類似していないことを表す。

$$\cos(\vec{q},\ \vec{d}) = \frac{\vec{q}\cdot\vec{d}}{|\vec{q}||\vec{d}|} = \frac{\vec{q}}{|\vec{q}|}\cdot\frac{\vec{d}}{|\vec{d}|} \tag{4}$$

4. 検証と考察

4-1. トピックの検証

4-1-1. シラバスデータの取得・整形

本研究では、神戸大学外部公開用シラバスの国際人間科学部の 2016 年から 2020 年の 3,501 授業のうち「科目名」、「時間割コード」、「開講年度」、「授業のテーマ」を Python パッケージの Beautiful Soup と Selenium を用いてスクレイピングした。授業のテーマのテキストデータには Janome を用いて形態素解析を行い、わかち書きをした。名詞が授業の特徴を表すと仮定し、わかち書きされたシラバスの単語群の中から名詞のみを抽出した。表 2 は授業名とそのシラバスの授業テーマから抽出された名詞の一例である。

表3. シラバスの一例

授業名	名詞
音楽文化史 1	エポックメーキング,音楽,作品,作曲,家,音楽,芸術,表現,様式,変遷,社会,文化,史,意味,考察
現代社会理論 A	貧困,共有,事態,人類,歴史,共同,性,基礎,近代,後,個人,化,過程,貧困,忘却,進展,現代,私,自己,認識,社会,帰結,私,キーワード,現代,時,空間,認識
情報リテラシー演習 1	オンライン,コミュニケーション,文書,処理,計算,基本,操作,方法,身,情報,機器,具体,活用,技能,習得

4-1-2.　トピック数の検討

トピックモデルの分析には、Pythonライブラリのgensimを用いた。4-1のシラバスデータに対して、トピック数を2~50に変化させながら、PerplexityとCoherenceを求めプロットしたものを図2に示す。Perplexityは大きいほど良く、Coherenceは小さいほど良いとされるのでトピック数を6に設定した。

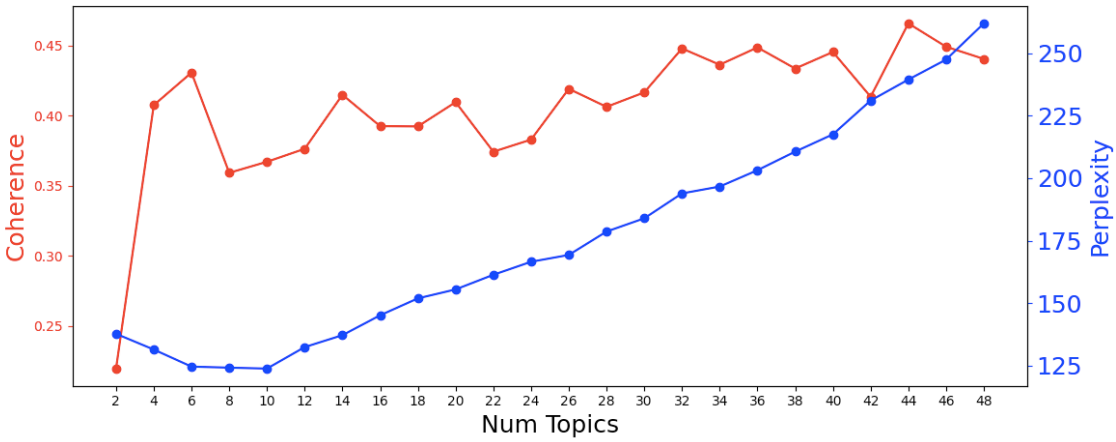

図 2.　PerplexityとCoherenceのプロット結果

4-1-2.　LDA モデルの生成

トピック数6 にしてLDAを実行し、トピックに属する単語をワードクラウドで表示したものを図3に示す。ワードクラウドは各トピックにおける発生頻度が上位30位の単語を元に生成されており、発生頻度が大きいほど文字が大きくなっている。

図3.　LDAによって生成された各トピックに属する単語をワードクラウドで表示したもの

4-2.　レコメンドの検証

4-2-1.　検証方法

レコメンド方法についての評価にあたっては、成績評価のある既履修の科目の中からレコメンドし、レコメンドされた順番と成績の相関係数を見る方法を行った。成績とレコメンド順に正の相関関係があれば、学生の科目に対する得意・不得意に即したレコメンドができていると考えられる。実際のデータは、神戸大学国際人間科学部グローバル文化学部に所属する学生4名の履修成績データを用いた。成績は、秀を4.2、優を4、良を3、可を2、不可を0に変換した。成績が「合格」となっている授業は基本的に必修授業のため、今回はおすすめに関与しないように0とした。履修取り消しされた授業は成績データから除いた。

4-2-2.　検証結果

結果を表6に示す。最も相関係数が大きい学生の値でも約0.12であるためほとんど相関はない。横軸をレコメンドの順位、縦軸を成績としたプロットを図4に示す。レコメンドが適切であるならば、レコメンド順位が高い横軸の左部分に好成績の授業が、おすすめ度が低い右部分に成績の低い授業がプロットされ、左上から右下がりの図になる。しかし図を見てわかるように、好成績の授業は左右に散らばっており、反対に成績の低い授業も点在している。このことから、今回の手法が、学生の成績の良い科目をうまくレコメンドできているとはいえない。

表4.　成績とレコメンド順の相関係数

学生	相関係数
学生A	-0.0994319
学生B	0.08027
学生C	-0.0578176
学生D	0.116057

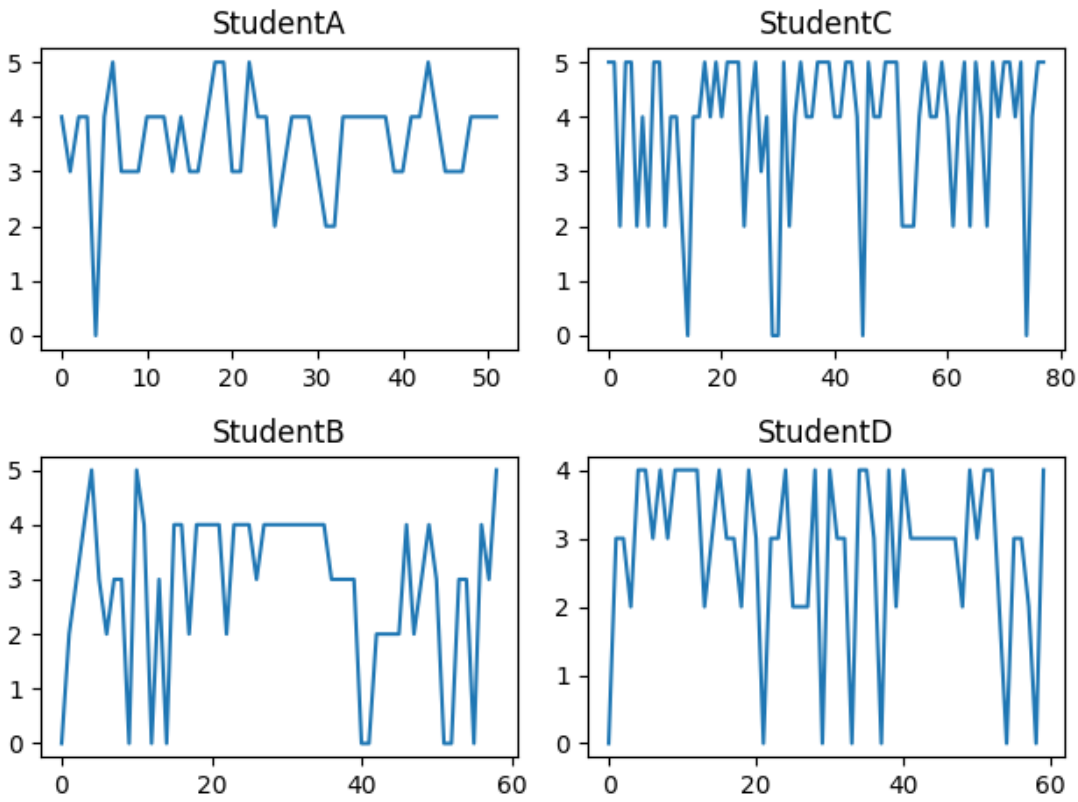

図4. プロット結果（横軸：レコメンド順, 縦軸：成績）

4-3. 考察

4-3-1. シラバスデータ

成績とレコメンド順に相関がない理由として大きく分けて3つの原因が考えられる。シラバスのテキストデータ、レコメンド方法、トピック以外の成績に影響する要因である。それぞれに対して考察していく。

今回使用したシラバスの授業テーマのテキストデータは授業によってその書き方や量にばらつきがある。例えば近現代社会思想論Aの授業テーマは「近代社会をめぐる諸理論」のみであり、LDAモデル生成の際に用いられている単語は「近代、社会、理論」の3単語のみである。この3単語から授業の特徴を捉えることは難しい。このようにシラバスの文章量

が少なく、特徴を捉えられていない授業があるためおすすめの精度が低下していることが考えられる。また、シラバスのテキストデータの問題により、トピックの生成もうまくいっていない。図3のトピックのワードクラウドを見てみると、トピック3は「臨床」や「心理」などの単語があることから心理系であると考えられるが、その他のトピックは一見してなんのトピックを表しているか推測できない。トピックをより正確にするにはより多くのデータが必要であることが考えられる。

4-3-2. レコメンド方法

今回のレコメンド方法は授業のトピックベクトルに直接成績をかけて、トピックごとに合計して得られた嗜好性ベクトルと授業のトピックベクトルのコサイン距離が小さいものをレコメンドするという方法である。しかし、この方法以外にも学生が一番得意なトピックの値が大きい授業をおすすめする方法や他の学生の嗜好性ベクトルを用いて協調フィルタリング的におすすめする方法もなども考えられる。

4-3-3. トピック以外の成績に影響する要因

今回はトピックベクトルと成績のみを用いてレコメンドを行った。図5は得られた嗜好性ベクトルとレコメンドされた授業のトピックベクトルをターミナルに出力したものである。上に表示されてるほどレコメンド順が高い。これを見てみるとこの学生はトピック4の値が約46.6と最も大きくおすすめされている授業も全てトピック4の値が最も大きい。図6は授業のトピックベクトルと成績を表示したものである。認知コミュニケーション論1, 2、近現代社会思想論A, Bはそれぞれトピックベクトルが非常に近い。しかし学生の成績はそれぞれ2と5、4と3など異なっており、学生は授業のトピックベクトルが近い授業でも成績にばらつきがあることがわかる。この要因としては、学生の履修したときの状況や成績評価基準の相違などが考えられる。例えば、体調の悪化や学外の用事などがあれば、授業への参加率は下

がり成績が下がる。その他には、同じテーマの授業でも成績の評価がテスト形式からレポート形式に変われば成績が異なる場合が考えられる。このように学生の成績には授業のテーマに現れない様々もあり、テーマのみから推定されるトピックを用いてレコメンドした場合、必ずしも成績が良いもののレコメンド順位が高くならないということが考えられる。

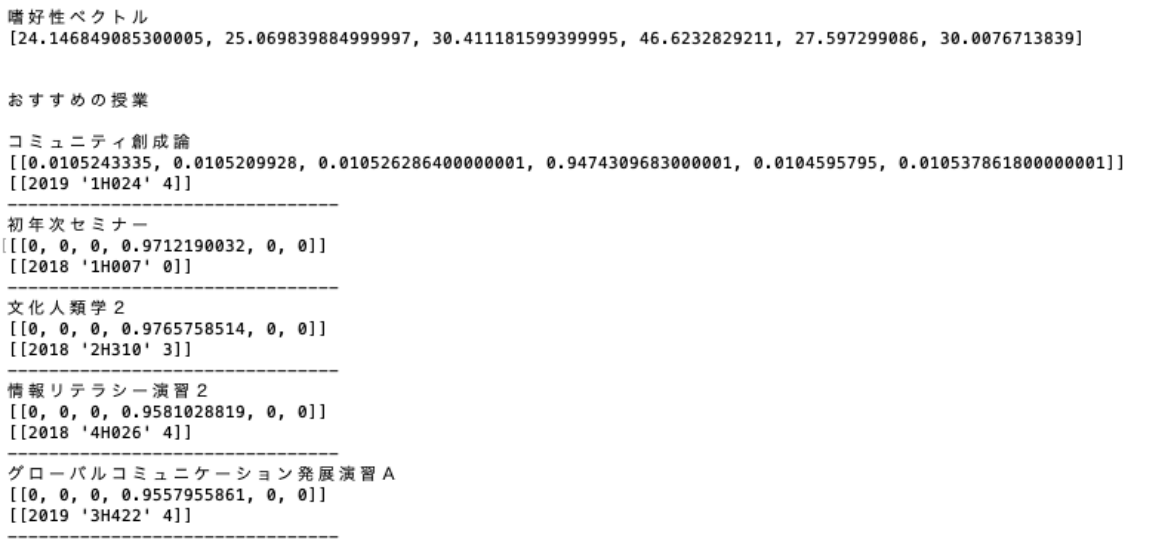

図5.　嗜好性ベクトルとおすすめされた授業のトピックベクトルの一例

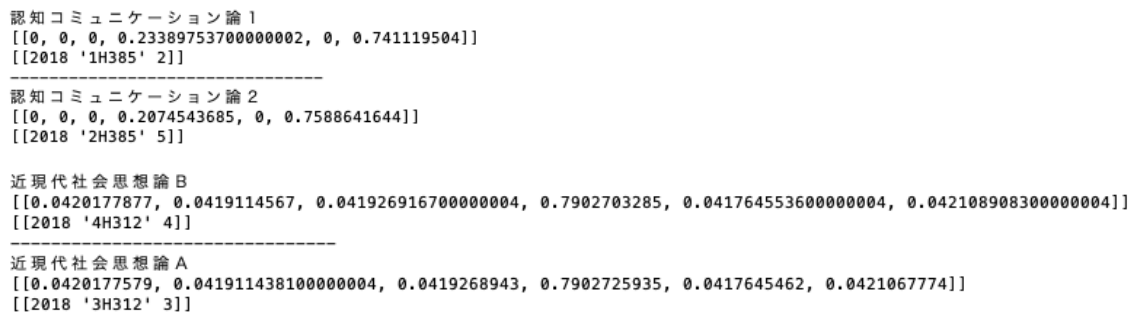

図6. 授業のトピックベクトルと成績

5. おわりに

5-1. まとめ

本研究では、LDA トピックモデルと成績を用いた履修レコメンドシステムを開発し、成績とレコメンド順の相関を求めることでその評価を行った。その結果、成績とレコメンド順には相関関係が見られず、今回のレコメンド方法では適切なレコメンドができていないことが確認された。

5-2. 今後の課題と展望

レコメンドがうまくいかなかった原因として、データ、レコメンド方法、トピック以外の要因の成績に対する影響を挙げた。データの問題は、一部の授業のデータの量が少なかったことに起因すると考えられる。そこで今回用いた授業テーマだけでなく、授業の概要と計画や成績の評価方法などを用いることによって、データの量と多様性を増やすことで解決できる可能性がある。レコメンド方法は、コサイン類似度以外にも、一番得意なトピックのみを考慮しておすすめする方法が考えられる。トピック以外の成績に影響する要因は、成績評価方法や学生の環境の変化などが考えられる。前者はデータの問題でも述べたようなシラバスの成績評価基準を用いることで考慮に入れることができる。後者は要因を特定することが難しい。履修成績にある授業に対して持っていたモチベーションをアンケートによって集め、モチベーションと成績の観点から分析することができるかもしれない。

トピックと成績の相関は今回の研究では見られなかったが、トピックの精度をあげることや多様なデータと組み合せることで、LDA トピックモデルと履修成績を用いた授業レコメンドができる可能性がある。また、今後は LDA トピックモデルに加えて、協調フィルタリングの技術を用いて、複数の学生の履修成績を考慮に入れた履修レコメンドシステムの研究を行っていきたい。

参考文献

[1] 竹森汰智, 亀井清華. “科目推薦のための doc2vec の応用方法の検討”. 情報処理学会. 2018

[2] 西森友省, 堀幸雄, 今井慈郎. “履修履歴を用いた科目推薦システム”. 情報処理学会. 2013.

[3] 奥村学, 佐藤一誠. “トピックモデルによる統計的潜在意味解析”. コロナ社. 2015.

[4] “トピックモデルをザックリと理解してサクッと試した”. （最終閲覧日：2020 年 12 月 19 日）https://qiita.com/d-ogawa/items/c423cd4b01c6ed84a5e7

[5] “LDA とそれでニュース記事レコメンドを作った。”.（最終閲覧日：2020 年 12 月 19 日）http://tdual.hatenablog.com/entry/2018/04/09/133000#ニュース記事レコメンドの作成

[6] “LDA によるトピック解析 with Gensim”. （最終閲覧日：2020 年 12 月 19 日）https://qiita.com/Spooky_Maskman/items/0d03ea499b88abf56819

LDAトピックモデルと履修成績を用いた履修レコメンドシステム

所属：情報コミュニケーション論講座 IT コミュニケーションコース

学籍番号：1686592C

氏名：宮崎仁弥

本論文では「LDA トピックモデルと履修成績を用いた履修レコメンドシステム」について記す。

大学生が受講できる授業の数は大変多くなっている。そのため、学生は自分の趣味・嗜好に合わせて履修することが可能になっており、授業選択の自由度が高くなっている。しかし授業が多様化した反面、履修計画を建てることは煩雑化した。数ある科目の中からシラバスを確認し、自分が興味を持てる授業なのかなどの判断をしながら履修する科目を探し出すことはなかなか時間がかかる。そこで、履修スケジュールを考える時間の短縮と履修成績の分析による学生の得意・不得意なトピックの分析を目的として、本研究では LDA トピックモデルと履修成績を用いた科目レコメンドシステムを構築、その評価を行った。

科目のシラバスデータは、神戸大学学外公開用シラバスの国際人間科学部の 2016~2020 年度のシラバスデータを用いた。

レコメンド方法は以下の通りである。学生の履修済みの授業のトピックベクトルに成績を数値化したものをかけ、トピックごとに合計したものを、その学生のトピックに対する嗜好性ベクトルとする。これから履修する科目の中からその科目のトピックベクトルと嗜好性ベクトルのコサイン類似度が高い科目をおすすめする。

このレコメンドシステムの評価を行った。評価には、履修成績の中から授業をレコメンドし、そのレコメンドされた順番と成績の相関関係を求める方法を用いた。その結果、レコメ

ンド順と成績の間には相関がないことが明らかになった。その原因として、シラバスのテキストデータ、レコメンド方法、トピック以外の成績に影響する要因の 3 つが考えられる。今回用いたデータはシラバスの内の授業のテーマの部分のみだったので、科目によっては文量が非常に少ない科目もあり、トピックを適切に抽出できなかった可能性がある。レコメンド方法に関しては、得意なトピックのみに基づいてレコメンドする方法や協調フィルタリングを用いることで他の学生のデータを考慮に入れてレコメンドする方法が考えられる。また、学生の履修履歴のみではなく、履修した際の学生の心理的状況や環境などのトピック以外の成績に影響している要因を考慮することで、レコメンドの精度が上がることが考えられる。

付録

ソースコード

プログラム 1　make_lda.py

```
import pickle
from gensim import corpora, models, similarities
import gensim
import math
import csv
import numpy as np

from gensim.corpora.dictionary import Dictionary
from gensim.models import LdaModel
from collections import defaultdict

import pandas as pd
import itertools

from tqdm import tqdm
import matplotlib
import matplotlib.pylab as plt
import json
from wordcloud import WordCloud

import logging



df = pd.read_csv('syllabus_globun.csv', usecols=[0])
class_names = df.values.tolist()
class_names = list(itertools.chain.from_iterable(class_names))

f = open("theme_words.csv", "r")
reader = csv.reader(f)
texts = [e for e in reader]
f.close()
```

```
    dictionary = corpora.Dictionary(texts)
    print(dictionary)
    # make corpus
    corpus = [dictionary.doc2bow(t) for t in texts]


    # tfidf
    tfidf = gensim.models.TfidfModel(corpus)


    # make corpus_tfidf
    corpus_tfidf = tfidf[corpus]

    NUM_TOPICS = 6

    # LDA Model
    # logging.basicConfig(format='%(asctime)s : %(levelname)s : %(message)s', level=logging.INFO)
    lda_model = gensim.models.ldamodel.LdaModel(corpus=corpus_tfidf, id2word=dictionary,
num_topics=NUM_TOPICS, alpha='symmetric', random_state=0)

    # test
    N = sum(count for doc in corpus for id, count in doc)
    print("N: ",N)

    perplexity = np.exp2(-lda_model.log_perplexity(corpus))
    print("perplexity:", perplexity)

    # テストデータをモデルに掛ける
    test_corpus = [dictionary.doc2bow(text) for text in texts]

    topic_results = []
    # クラスタリング結果を出力
    for unseen_doc in test_corpus:
        score_by_topic = [0] * NUM_TOPICS
        for topic, score in lda_model[unseen_doc]:
```

```
            score_by_topic[topic] = score
        topic_results.append(score_by_topic)
from pprint import pprint

df = pd.read_csv('syllabus_globun.csv')
df['トピックの確率'] = topic_results

np.random.seed(0)
FONT = "/Library/Fonts/Arial Unicode.ttf"

ncols = math.ceil(NUM_TOPICS/2)
nrows = math.ceil(lda_model.num_topics/ncols)
fig, axs = plt.subplots(ncols=ncols, nrows=nrows, figsize=(15,7))
axs = axs.flatten()

def color_func(word, font_size, position, orientation, random_state, font_path):
        return 'black'

for i, t in enumerate(range(lda_model.num_topics)):

        x = dict(lda_model.show_topic(t, 30))
        im = WordCloud(
                font_path=FONT,
                background_color='white',
                color_func=color_func,
                random_state=0
        ).generate_from_frequencies(x)
        axs[i].imshow(im)
        axs[i].axis('off')
        axs[i].set_title('Topic '+str(t))

plt.tight_layout()
plt.savefig(f'visualize_{NUM_TOPICS}.png')
plt.show()
```

プログラム 2　test_reccomendation.py

```
import pandas as pd
import pickle
import csv
import json
import numpy as np
import scipy.stats

topic_num = 6
sum_topic_odds = [0] * topic_num

def get_topic_value(nendo,code):
    topic_grades = None
    taken_class = df[(df['年度'] == nendo) & (df['時間割コード'] == code)]
    topic_value = taken_class['トピックの確率'].values.tolist()
    return topic_value



def search_goodat_topic(nendo, code, grade):
    topic_grades = None
    taken_class = df[(df['年度'] == nendo) & (df['時間割コード'] == code)]
    topic_value = taken_class['トピックの確率'].values.tolist()
    if len(topic_value) == 1:
        topic_grades = [n * grade for n in topic_value[0]]
    return topic_grades

def cos_sim(v1, v2):
    v1_array = np.array(v1)
    v2_array = np.array(v2)
    return np.dot(v1, v2) / (np.linalg.norm(v1) * np.linalg.norm(v2))

def reccomend(nendo,code):
    _class_names_cos_sim = []
    taken_class = df[(df['年度'] == nendo) & (df['時間割コード'] == code)]
    class_names_topics = taken_class[['科目名','年度','時間割コード','トピックの確率
']].values.tolist()
```

```
        for class_name_topic in class_names_topics:

            similarity = cos_sim((scipy.stats.zscore(class_name_topic[3])),sum_topic_odds)
            class_name_topic[3] = similarity
            _class_names_cos_sim.append(class_name_topic)
        return _class_names_cos_sim

df = pd.read_json('syllabus_tfidf.json')
with open('data/grades/StudentA.csv') as f:
        h = next(csv.reader(f))
        reader = csv.reader(f)
        grades = [e for e in reader]
        f.close()

df2 = pd.read_csv('data/grades/StudentA.csv')

count = 0
for row in grades:
        topic_grades = search_goodat_topic(int(row[0]), row[1], float(row[2]))
        if topic_grades is not None:
            sum_topic_odds = [topic_grades[i] + sum_topic_odds[i] for i in range(len(topic_grades))]
            count += 1

class_names_cos_sim = []
for row in grades:
        class_names_cos_sim.extend(reccomend(int(row[0]), row[1]))
reccomend_class = sorted(class_names_cos_sim, reverse=True, key=lambda x: x[3])



print("¥n 嗜好性ベクトル")
print(sum_topic_odds)
print('¥n')
print("おすすめの授業¥n")
for i in reccomend_class[0:10]:
        print(i[0])
        topic_value = get_topic_value(i[1],i[2])
```

```
        print(topic_value)
        class_grade = df2[(df2['年度'] == i[1]) & (df2['時間割コード'] == i[2])]
        print(class_grade.values)
        print('---------------------------------')
```

謝辞

mulabo の方々には大変お世話になりました。村尾元教授には IT の面白みや奥深さを教えていただき、私の人生の幅を広げていただきました。私の疑問に対し、的確な答えと他愛のない会話で楽しく導いてくださりありがとうございます。

院生の三嶋哲也氏、谷口哲郎氏、前川絵吏氏、川田恵氏は、非常に優しくいつも進捗を気にかけてくださいました。院生の方々のおかげで、落ち着きながらも和気藹々とした環境で研究することができました。特に三嶋哲也氏は 64 を提供してくださり感謝しております。氏を超えることを目標に今後も研究に励みます。

最後に、同ゼミ生の甲斐みち栞氏、武内萌氏、森愛子氏、コニマオ氏、ぱ氏とは、菓子を交換したり雑談をしたりと、皆さんのおかげで楽しみながら研究を進めることができました。特にコニマオ氏とぱ氏とはいつも遅くまで研究室に残り、お菓子パーティやマリオパーティなどを挟みながらした最後の追い上げは、忘れ難き思い出です。

これからも mulabo の一員として、先輩の後を継ぎ、後輩を支えられるような存在になれるよう精進いたします。