CS 410 Text Information Systems 2021 Fall - Project Report
Causal Topic Modeling

Team Information

- Team Name: MP
- Team Members: Masami Peak (NetID: masamip2, Email: masamip2@illinois.edu) – Captain – **Requirement #4**

Topic Information

Description:
This project is building a causal topic model for identifying hidden key topics in the MLB (Major League Baseball) articles correlated to the annual MVP (Most Valuable Player) winner.

How to Use/Test the Software – Requirement #3

How to Access the Environment
Please watch the software usage tutorial: Project Presentation (https://mediaspace.illinois.edu/media/t/1_7h9807wh) or follow the steps below. All required modules and datasets for the project are provided in the online environment. For the case where you would like to set up the same development environment of this project, I additionally introduce The Software Development and Installation in the README.md of the Git repository of this project (https://github.com/masamip2/CourseProject#the-software-development-and-installation).

1. Go to the Online Jupyter Notebook (https://mybinder.org/v2/gh/masamip2/CourseProject/HEAD) of this project.
2. Double-click the 'mlb.ipynb' file in the left folder structure section to display the file.
3. Click >> icon ('Restart the kernel, then re-run the whole notebook' button) under the tab of the file 'mlb.ipynb'.

NOTE: If a popup message 'Server Connection Error' comes up, please access the above link to open the page again.

Additional Features
The main functions of the online version are very similar to the ones of the normal Jupyter Notebook. The differences can be:

- Adding custom data to the 'Data' folder by clicking the upload icon ('Upload Files') at the top of the folder structure.
- Downloading or exporting the files (e.g., .ipynb) on the environment per file through the 'File' menu at the top.

An Overview of Process – Requirement #1

- The software is for Causal Topic Modeling on MLB (Major League Baseball) articles.
- The goal is to find the name of MVP (Most Valuable Player) in any of the topics on the documents for the year.

1. **Module Installation & Data Loading**
   The datasets under the Data folder are loaded.

2. **Data Cleaning & Preprocessing**
   The main dataframe 'articles' is created after data cleaning and preprocessing.

3. **Document Processing**
   2 items: DICTIONARY and Corpus needed for LDA Model are created.
   - DICTIONARY is created based on Trigram-Terms. (e.g., {TermID: Term} = {0: 'apple', 1: 'banana', …})
   - (Trigram-Terms are created based on Vocabulary.)
   - (Vocabulary of Documents is created through Word Tokenization, Lemmatization, Stop-words Removal, Stemming.)
   - Corpus is created based on Dictionary and Trigram-Terms. (e.g., [(TermID, Frequency)] = [(0, 1), (1, 2), …])

4. **Topic Model & Coherence Score**
   - Each Topic Model for a specific year is built using DICTIONARY and Corpus.
   - Coherence Score, which indicates the interpretability between the topics and the relevant terms, is used to find the optimal number of topics.

5. **Topic Model Evaluation**

- An LDA Model with the number of topics which has the highest Coherence Score is chosen.
- The name of MVP is looked for in any of the topics.

6. **Topic Model Visualization**
   The Topic Model for each year is visualized to analyze the topics and the relevant terms. The following data is displayed.
   - Top 5 Most Relevant Terms for the Topic which has the name of MVP.
   - The rank of the MVP's name in the Top 300 Most Relevant Terms.
   - The bar chart indicates if a term frequency in the topic dominates the overall term frequency, the term represents the topic very well.
7. **LDA Model Diagram**
   The diagram illustrates k topic probability distribution on a document and m words probability distribution on a topic, using 2 sample documents.

How to Customize and Run the Software

The software can be customized for any award-related genre, such as NFL (National Football League), NBL (National Basketball League), Movie (e.g., Academy/Oscar Awards), or Music (e.g., Grammy Awards). You are required to obtain the datasets for the specific sport or art based on the particular column types and the constants are also configurable.

1. **Article Datasets**: CSV files with any name, except 'mvps.csv' and 'articles.csv', under the 'Data' folder will be converted to the Pandas' dataframe type and saved as 'articles.csv'. The custom datasets have to have the following columns:
   - id: String
   - headline: String – the headline of the article
   - summary: String – the summary of the article
   - created: String (date format: '%Y-%m-%d') – the published date of the article
   - source: String – the source of the article

2. **Supporting Dataset**: The 'mvps.csv' can support data analysis. The custom datasets have to have the following columns:
   - id: Int
   - name: String – the name of the MVP
   - team: String – the team of the MVP
   - league: String – the league of the MVP's team
   - year: Int – the year of the MVP award

3. **Applicable Months**: Minimum and maximum months for the genre's season have to be configured.
   - MIN_MAX_MONTHS = {'min': 4, 'max': 10}

4. **Applicable Years**: Minimum and maximum years of the genre's articles have to be configured.
   - MIN_MAX_YEARS = {'min': 2011, 'max': 2021}
5. **Evaluating Years**: The range of the years for evaluating the hidden topics has to be configured.
   - YEARS = [2018, 2019, 2020, 2021]

6. **Stop-wards**: Words to be ignored for the genre have to be configured.
   - STOP_WORDS = ['mlb', 'major', 'league', 'baseball', 'game', 'team', 'player']

7. **Topic Model**:
   - **Minimum Total Count**: Minimum total count of each n-gram in the collection should be configured.
     MIN_COUNT = 1 (**NOTE**: See the section 'MIN_COUNT Test' in mlb.ipynb for defining the number)

   - **Threshold for N-Grams**: Phrase of words \`a\` \`b\`: (cnt(a, b) - min_count) * TotalVocabSize / (cnt(a) * cnt(b)) > threshold should be configured. Smaller threshold allows longer phrases (e.g., Los Angeles Angeles).
     THRESHOLD = 1 (**NOTE**: Longer phrases seem preferred for team names, location names, etc)

   - **Numbers of Topics**: The range of the numbers of topics for a topic model may be configured.
     NUMS_TOPICS = {'min': 2, 'max': 10} (**NOTE**: number of topics more than 10 seems too many topics for MLB)

   - **Default Number of Topics**: Default number of topics in the corpus may be configured.
     NUM_TOPICS = 5

- **Number of Documents**: Number of documents in each chunk can be configured.
  CHUNKSIZE = 100 (**NOTE**: huge chunk size does not seem appropriate for less than 3000 documents.)

- **Number of Passes**: Number of passes in the corpus (going through the entire corpus) can be configured.
  PASSES = 10 (**NOTE**: small number of passes does not seem appropriate for less than 3000 documents.)

- **Seed Setting**: Seed for reproducibility can be configured.
  RANDOM_STATE = SEED (defined for general use (e.g., os.environ["PYTHONHASHSEED"] = str(SEED)))

- **Coherence Measure**: Consistency measurement for topic segmentation can be configured like below.
  COHERENCE = 'c_v'
  c_v: co-occurrence counts of top words of the topic
  u_mass: occurrence counts of common words
  c_uci: co-occurrence counts of words
  c_npmi: normalized c_uci

How the Software Implemented – Requirement #2

<u>Module Installation and Data Loading</u>

The required modules are actually preinstalled using requirements.txt. The datasets are loaded using the following functions.

1. **install_modules(modules= MODULES)**: The necessary modules are installed, if any of them are not installed yet. (MODULES = ['pandas', 'nltk', 'gensim', 'matplotlib', 'pyLDAvis'])
2. **load_dataframe(file_path, usecols, dtype)**: The csv files are loaded as Pandas' dataframe type with:

   <u>'articles' dataframe</u>
   - usecols=USECOLS (USECOLS = ['id', 'headline', 'summary', 'created', 'source'])
   - dtype=DTYPES (DTYPES = {'id': str, 'headline': str, 'summary': str, 'created': str, 'source': str})

   <u>'mvps' dataframe</u>
   - usecols=USECOLS_SUP (USECOLS_SUP = ['id', 'name', 'team', 'league', 'year'])
   - dtype=DTYPES_SUP (DTYPES_SUP = {'id': int, 'name': str, 'team': str, 'league': str, 'year': str})

3. **concat_csvs()**: all the article CSV files from different sites are combined to be a single dataset of articles.

<u>Data Cleaning and Preprocessing</u>

The text data in the combined dataset is cleaned and preprocessed using the following functions.

1. **filter_created(df1)**: the articles (df1) are filtered on the 'created' column by the appropriate range of the months and years.
2. **remove_duplicate(df1)**: duplicates on 'headline' and 'summary' columns are removed from the articles (df1).
3. **preprocess_text(text)**: the data cleaning tasks below are applied to the text on the 'headline' and 'summary' columns:
   - lowercasing the text
   - replacing newline, return, punctuation (any special character), and multiple whitespaces with a whitespace
   - replacing digit with a whitespace
   - replacing ' s ' with ' ' (e.g., 'Angeles s players' -> 'Angeles Players')
   - transforming ' (char) (char) ' and '(char) (char) ' to '(char)(char)' (e.g., ' n y ' and 'n y ' -> 'ny')
   - transforming '(characters) t ' to '(characters)t' (e.g. 'wasn t ' -> 'wasnt')
   - stripping the left and the right whitespaces
4. **concat_text(df1)**: the 'headline' and 'summary' are combined onto the 'text' and the year on the 'created' is on the 'year'.
5. **concat_name(df2)**: the MVP's name in each league per year are joined by ';' on the 'name' if the names are preferred to be merged into the articles (e.g., 'Abreu;Freeman' in 2020).
6. **save_articles(df1, df2=None, file_name=ARTICLES_FILE)**: the articles (optionally, the 'name' on mvps are merged into the articles on the 'year' column) are saved as ARTICLES_FILE (ARTICLES_FILE = 'articles.csv').
7. **preprocess_articles(df1, df2=None)**: the articles are preprocessed to create an 'articles' dataset.

<u>Document Processing</u>:

The **DICTIONARY**, the **trigram_terms,** and the **corpus** for an LDA model are created based on the cleaned and preprocessed text data using the following functions.

1. **create_vocabulary(docs)**: the following document processing tasks are applied to the combined text (documents):
   - Word Tokenization: making each word in the text a word token
   - Lemmatization: grouping together the inflected forms of a word (e.g., blocks -> block)
     (**NOTE**: See the section 'lemmatizer & Stemmer Test1' in mlb.ipynb for deciding to use Lemmatization.)
   - Stop-words Removal: excluding common words from the lemmatized words
   - Stemming: reducing the derived forms of a word (e.g., blocked -> block)
     (**NOTE**: See the section 'lemmatizer & Stemmer Test2' in mlb.ipynb for defining the stemming algorithm.)
2. **create_ngram_model(vocab)**: a model of phrases is trained on the vocabulary.
3. **create_trigram_terms(docs)**: trigram terms are created by the traceable steps below:
   vocabulary: created based on the documents
   bigram terms: created by a bigram model trained on the vocabulary
   trigram terms: created by a trigram model trained on the bigram terms
4. **create_dictionary(trigram_terms)**: a DICTIONARY is created based on the trigram terms on the vocabulary from all the documents, and referred for creating a corpus or an LDA model. (e.g., **{TermID: Term}** = {0: 'apple', 1: 'banana', …})
5. **create_corpus(trigram_terms, dictionary= DICTIONARY)**: a corpus is created based on the trigram terms and the DICTIONARY. (e.g., [**(TermID, Frequency)**] = [(0, 1), (1, 2), …])
6. **create_trigram_corpus(docs)**: trigram_terms and a corpus are created based on the documents with the DICTIONARY.

<u>Topic Model and Coherence Score</u>:

The coherence score of each topic model with a different number of topics is calculated using the following functions.

1. **build_lda_model(params)**: gensim.models.ldamodel.LdaModel(params) is used with the following params:
   - corpus=corpus: the corpus [(TermID, Frequency)] created based on the trigram terms and the DICTIONARY
   - id2word=DICTIONARY: the DICTIONARY {TermID: Term} created based on the trigram terms
   - num_topics – can be set in 2 different ways below:
     =NUM_TOPICS: the default number of topics in the corpus defined at the NUM_TOPICS
     =num_topics: the number in the range defined at the NUMS_TOPICS (e.g., {'min': 2, 'max': 10})
   - chunksize=CHUNKSIZE: the number of documents in each chunk defined at the CHUNKSIZE
   - passes=PASSES: the number of passes in the corpus defined at the PASSES
   - random_state=RANDOM_STATE: the seed for reproducibility defined at the RANDOM_STATE (= SEED)
2. **compute_coherence_score(params)**: gensim.models.CoherenceModel(params) is used with the following params:
   - model=lda_model: the LDA model created by gensim.models.ldamodel.LdaModel function
   - texts=trigram_terms: the trigram terms created by a trigram model
   - dictionary= DICTIONARY: the DICTIONARY {TermID: Term} created based on the trigram terms
   - coherence=COHERENCE: the consistency measurement for topic segmentation defined in COHERENCE
3. **optimize_num_topics(corpus, trigram_terms, dictionary=DICTIONARY)**: the coherence scores are found for each number of topics.

<u>Topic Model Evaluation</u>:

The optimal topic model with between 2 and 10 topics is chosen based on the highest coherence score. LDA model returns different results due to the methods applying randomness, if any seed is not set at the random_state parameter in the gensim.models.ldamodel.LdaModel() function. Evaluation is performed using the following functions.

1. **save_model(year)**: the LDA model for the year can optionally be saved as 'topic_model_{year}.pkl'.
2. **get_mvps(year)**: the subset of the mvps dataset and the names of the MVPs as vocabulary for the year are retrieved.
3. **print_mvps_topics(lda_model, names)**: the names of the MVPs for the year in the terms of the optimal number of the topic model are displayed. The top 5 terms are also shown and the rest of the terms are abbreviated.
4. **evaluate_model(year, to_save=False)**: the following tasks are performed for the specified year.
   - subset the documents for the year
   - create trigram_terms and a corpus based on the documents
   - build lda_models and calculate the coherence_scores based on the trigram_terms and the corpus
   - show a plot of 'Number of Topics v.s. Coherence Score'
   - choose the optimal lda_model with the specific number of topics based on the best coherence_score
   - print 'Optimal Number of Topics', 'Best Coherence Score', 'MVPs in any of the {k} Topics'
   - show the MVPs for the year as reference
   - print the names of the MVPs as terms in any Topic

Topic Model Visualization:
The plot 'Number of Topics v.s. Coherence Score' shows the optimal number of topics. Visualizing the optimized LDA topic model explains the term frequencies within the selected topic using the following function.

1. **pyLDAvis.gensim_models.prepare(lda_model, corpus, dictionary=DICTIONARY)**: The optimized LDA model on the corpus for the specific year with the DICTIONARY is visualized.

Once 'Intertopic Distance Map' shows up, either clicking each circle representing a topic or the 'Next Topic' button to select a topic enables to see the 'Top-30 Most Relevant Terms for Topic {k}' in a bar chart. After a topic is selected, you can hover over each term on the right bar chart to see the term distribution or frequency on the topic v.s. overall term frequency.

- **The Year 2018**

Name of MVPs (& League)	Topic Circle #	Rank in the 300 Terms	Top 5 Most Relevant Terms for the Topic
Mookie Betts (American)	1	74	red_sox, pitcher, ray, angel, ha
Christian Yelich (National)	4	11	best, sixth_inning, playoff, brewer, houston_astro

- Optimal Number of Topics: 6
- Mookie Betts: He was in the team Boston Red Sox in 2018, but it seems his name is low in the rank of the relevant terms for the topic. Due to the small number of sampled documents for 2018, it is more difficult to see any trend.
- Christian Yelich: He is in the team Milwaukee Brewers, and it seems his name is high in the rank of the relevant terms for the topic. Also, the top word for the topic is 'best' and the bar chart indicates that its term frequency in the topic dominates overall term frequency, which means the word 'best' represents the topic very well.

- **The Year 2019**

Name of MVPs (& League)	Topic Circle #	Rank in the 300 Terms	Top 5 Most Relevant Terms for the Topic
Mike Trout (American)	6	46	angel, late, won, doubl, employe
Cody Bellinger (National)	1	273	nation, astro, win, washington, houston_astro

- Optimal Number of Topics: 6
- Mike Trout: He is in the team Los Angeles Angeles, and it seems his name is not so high in the rank of the relevant terms for the topic. Although the 3rd most relevant term for the topic is 'won' and the bar chart indicates that its term frequency in the topic dominates overall term frequency, which means the word 'won' represents the topic very well.
- Cody Bellinger: He was in the team Los Angeles Dodgers, but it seems his name is very low in the rank of the relevant terms for the topic. However, all of the topic circles related to MVPs for 2018 and 2019 are located near the horizontal line on the 'Intertopic Distance Map', while they are segmented like, one side has one large topic circle and the other side has several smaller topic circles on the map. It is worth paying attention to how the topic circles are distributed for predicting MVP: any topic circles near the horizontal line that form one large circle or several smaller circles.

- **The Year 2020**

Name of MVPs (& League)	Topic Circle #	Rank in the 300 Terms	Top 5 Most Relevant Terms for the Topic
Jose Abreu (American)			
Freddie Freeman (National)			

- Optimal Number of Topics: 7
- Jose Abreu: His name is not found in any articles for 2020. The number of sampled documents for 2020 is very small because the number of games was reduced from 162 to 60 due to COVID-19. Thus, it is very hard to analyze the topic model for MVP for the year 2020 unfortunately.
- Freddie Freeman: Same as the above comment.

- **The Year 2021**

Name of MVPs (& League)	Topic Circle #	Rank in the 300 Terms	Top 5 Most Relevant Terms for the Topic
Shohei Ohtani (American)	5	10	seri, new, octob, singl, angel
Bryce Harper (National)	4	122	manag, end, histori, moment, walk

- Optimal Number of Topics: 10
- Shohei Ohtani: He is in the team Los Angeles Angeles, and it seems his name is high in the rank of the relevant terms for the topic. Also, the bar chart indicates that his name ('shohei_ohtani')'s term frequency in the topic dominates overall term frequency, which means the word 'shohei_ohtani' represents the topic very well.
- Bryce Harper: He is in the team Washington Nationals and it seems his name is low in the rank of the relevant terms for the topic. Although, the 10th top word for the topic is 'best'.

Further Analysis on the Name of 2nd MVPs for 2021

Name of 2nd MVPs (& League)	Topic Circle #	Rank in the 300 Terms	Top 5 Most Relevant Terms for the Topic
Vladimir Guerrero (American)	4	67	manag, end, histori, moment, walk
Juan Soto (National)	8	183	dodger, los_angel, chris_taylor, club, dodger_stadium

- Vladimir Guerrero: He is in the team Toronto Blue Jays in American League.
- Juan Soto: He is in the team Washington Nationals in National League.

Further Analysis on the Name of 3rd MVPs for 2021

Name of 3rd MVPs (& League)	Topic Circle #	Rank in the 300 Terms	Top 5 Most Relevant Terms for the Topic
Marcus Semien (American)	1	177	playoff, ha, year, wa, brave
Fernando Tatis (National)	6	155	taylor, Houston, astro, number, chang
	2	181	homer, san_francisco, giant, padr, need
	7	199	world_seri, wa, got, infield, star

- Marcus Semien: He is in the team Toronto Blue Jays in American League.
- Fernando Tatis: He is in the team San Diego Padres in National League.

LDA Model Diagram:
LDA model diagram illustrates k topic probability distribution on a document and m words probability distribution on a topic, using 2 sample documents.

1. **get_topic_distribution(lda_model, bow)**: lda_model.get_document_topics(bow, minimum_probability=1e-2) is used to get the topics with different colors for a document using the bag of words (corpus).
2. **create_term_topic_map(lda_model, bow)**: lda_model.get_term_topics(word_id, minimum_probability=1e-5) is used to get a map where a topic with the highest probability can be found for each word.
3. **display_terms_topic(doc, terms_topic, topic_ids_colors, doc_topics)**: the following tasks are prepared for display.
   - each topic probability (min=1e-2=0.01) for a document
   - each word probability (min=1e-5=0.00001) for the most likely topic
   - indicating a topic (min=1e-2=0.01) for each word (min=1e-5=0.00001) in a document
4. **display_diagram(lda_model, docs)**: LDA model diagram is displayed for each of the sample documents.

Other Requirements in the CS 410 Project Topics Google Doc

The Documented Source Code:

1. mlb.ipynb: the main source code to demonstrate and visualize LDA models for this project.
2. scraper.py: the supporting code to collect MLB-related article datasets.
3. Data (articles.csv, espn.csv, mlb.csv, mvps.csv, nyt.csv, reuters.csv, wsj.csv): the datasets of MLB-related articles.

Main Results:

1. Due to MVP being determined by the voters in the Baseball Writers' Association of America , intuitively, MLB players' names mentioned in MLB-related articles should have reflected the winner of MVP for the year. The number of MLB articles in 2021 is 942 which is more than twice the number in 2019. Thus, the analysis for 2021 shows a clear difference that the rank of MVP's name is higher compared to the 2nd and the 3rd MVP nominees in each league. This model definitely can be useful to reference for the next year.
2. Interestingly, most of the names of the finalist for the MVP award in the 2 leagues are relevant terms on their topic. In other words, the names are often not on the same topic. Thus, each of the topics can be analyzed more to see if it indicates the characteristics of the players in the finalist: not only their team names but also any key or positive term.

3. Lots of words in the articles seem neutral and that makes it harder to distinguish the hidden topics that are already in a known topic: Baseball. Although, each MLB player's name is the most important term and if the name is a highly relevant term for a topic, then that means the player has been talked about in MLB articles and has a higher chance of gaining the MVP award.

Self-evaluation:

I have completed this project successfully, especially I have got the expected outcome for the LDA model on the year 2021 MLB articles. For the previous years, it was difficult to analyze the outcome due to the insufficient amount of data. Thus, sample data (MLB-related articles) should be collected regularly to have a sufficient amount of data for building an LDA model with better performance. As a side note, the article sites tend to show newer articles on the page where people or web crawlers are easy to access and move older articles somewhere difficult to be found.

Demo (Project Presentation):

The software usage tutorial: Project Presentation (https://mediaspace.illinois.edu/media/t/1_7h9807wh)