Data+ 2021 Rubenstein Library Card Catalog File Overview

* **Helpful Terms**
	+ *Accessioned*: record the addition of to the library
	+ *Catalog*: refers to the Main Entry Catalog of the Rubenstein Library, the set of files which we are working with
	+ *Collection*: group of items (papers, letters, etc.) written by the same author and accessioned at the same time.
	+ *Drawer*: group of cards nearby in alphabetical order, main entry file contains 69 drawers, corresponding to their former physical drawers
	+ *Internet Archive*: site hosting search tool for exploring images of the cards
	+ *Main entry*: first card of a collection, typically providing metadata
	+ *Narrative*: cards following main entry, typically containing a summary of the contents of the collection
* **Notable Technologies Used**
	+ *Google Tesseract*: for running OCR on the cards
	+ *Jupyter Notebook*: IDE for running code
		- Jupyter Notebook version 6.3.0, Python 3 kernel
		- Anaconda version 4.10.3
	+ *Python*: language in which all files are coded in (version 3.9.5)
		- *Notable packages used*: pandas, numpy, matplotlib, plotly, geopandas, seaborn, wordcloud, gender\_guesser, regex, nltk, neattext, scipy, SpaCy, streamlit
	+ *OpenRefine*: data cleaning tool
* **General Files**
	+ README
		- Gives a general overview of what the project accomplished and provides notes for future researchers.
	+ *main\_file\_dataset.csv*
		- The final, structured dataset created from the full text generated from running OCR on jpegs of the card catalog and various methods of natural language processing and some manual editing
			* *Collection\_Head*: for main entries, is author name; for narrative cards, is blank
			* *Name*:author name for collection
			* *Text*: full text of card, OCR generated
			* *Year*: date items in collection were written, either a single year or a range of years
			* *Start*: beginning year of date range for date written
			* *End*: end year of date range for date written
			* *Location*: Where collection was written, if identified by SpaCy package
			* *Page\_drawer*: page number within drawer
			* *Drawer\_No*: drawer number of card
			* *Link*: link to corresponding card in Internet Archive
			* *Coll\_Head*: 1 if card is a main entry card, 0 if card is a narrative card (collections determined by algorithm described below)
			* *Collection*: collection number
			* *Author\_Identity*: classifies collection author as male, female, organization, or unknown
			* *Country*: country of origin for cards with valid location
			* *Continent*: continent of origin for cards with valid location
* **Catalog**
	+ *new\_text*
		- OCR-generated text files for each drawer of cards
	+ *file\_name\_checked*
		- Csvs with corrected author names for each drawer
	+ *Main\_File\_All*
		- Full text file of all OCRed cards, no alterations
	+ *csv\_iterations*
		- Old csv files, with various combinations of columns found in final dataset
			* *main\_file\_all\_text*: contains just text column, unedited OCR results
			* *all\_text\_chunked\_name*: contains pulled out names, prior to manual name editing
			* *all\_sorted\_before\_year\_column*: contains fixed names, is sorted by collections
			* *all\_sorted\_collection*: has all above, unedited regex collected years, links to Internet Archive cards, drawer number and page in drawer, and collection number
			* *All\_sorted\_keyword, all\_collection\_combined\_keywords*: contains keywords pulled from the text of each card (not included in final dataset)
			* *year\_fixed*: dataset after manually fixing some years
			* *Year\_Loc\_fixed*: dataset after making some fixes to year and location columns.
* **Initial Data Wrangling**
	+ *split\_to\_drawers*
		- Within each drawer, we attempted to glean the authors' names from each card. We used natural language processing from the nltk package and regular expressions to find proper nouns at the beginning of the cards that started with the start letter for that drawer.
	+ *text\_cleanup*
		- This code combined the whole OCRed text into one dataframe, with a row for each card
	+ *text\_name\_chunking*
		- This code cleaned common name errors using natural language processing and created a csv of all text with names
	+ *upload\_internet\_archive*
		- Working with the Internet Archive API, this code uploaded each drawer’s pdf to a labeled Internet Archive item
* **Spatial Analysis**
	+ *nc\_counties*
		- Shapefiles for NC county map
	+ *tl\_2020\_us\_state*
		- Shapefiles for USA state map
	+ *spatial\_frequencies*
		- Notebook containing all spatial related code, separated into 3 main sections
			* *USA*: finds counts of cards originating from each state and maps them using geopandas and matplotlib
			* *NC Counties*: does the same as the USA map, but for North Carolina counties instead of states
			* *International*: finds the counts of cards originating from foreign countries and visualizes the frequencies
* **Other Code:** Jupyter Notebooks for various data analysis
	+ *sort\_by\_collection*
		- In this notebook, we created an algorithm to sort the dataset into collections. This proved difficult, as there was no one way to tell a card from a narrative using the text alone. Thus, we studied the cards to find patterns that suggest the type of card a row belonged to. A new author's name signified a new collection automatically, but within an author's associated cards, we had to get more creative on how to differentiate between main entries and narrative cards.
		- Next, we continued data cleaning to iteratively build our dataset
			* Added a boolean column to determine if a row is a collection header
			* Extracted years using regular expressions for common patterns
			* Added start and end date columns for a collection, if applicable
			* Used SpaCy to glean the locations from the main entry card for each collection
			* Created a link for each row to the corresponding card in Internet Archive
			* Found keywords using TF-IDF scores
	+ *duke\_history*
		- Explores the presence of Duke history in the catalog through 3 avenues, through both quantitative and qualitative analysis
			* Duke Presidents
			* Duke building names
			* Duke’s nomenclature
	+ *gender\_demographics*
		- Using the gender\_guesser Python package, we classified collection authors by their likely gender, and analyzed these demographics
	+ *summary\_characteristics*
		- This notebook computes the number of collections, unique authors, author identity counts, and top five largest collections
	+ *word\_cloud*
		- We visualized the most occurring words in the collections by using Word Clouds. These word-clouds are also used as a comparison tool for female & male authors as well as the writings from different centuries.