What's This Really About?

Carly Larsson
MAT 259A

In the Beginning

The Seattle Public Library dataset is information dense in many dimensions, but it is missing odd bits of information that would seem necessary to a library. Neither the bibNumber nor itemNumber of a book links you to the genre of the book. One can access the subject of the book, but subject keywords are non standardize and vary between copies of the same novel. This lack of standardized genre or subject isn't an obstacle for Dewey classified books, but does pose a problem for nonDewey classified books.
Presumably fiction, also known as nonDewey books, are arranged in the Seattle Public Library near similar books, as is the case in libraries across the United States. Yet, there seems to be no information in the database on how this sorting occurs. In my mind it would happen by genera, just as Dewey books are shelved by there numeric category.
The Genre of a fictional book can be a hard to pin down. Perhaps it is not included in the database because it is not an absolute, and subject to change. For example, The Lord of the Rings could be shelved alongside classic pieces of literature or with other High Fantasy books. Though far from trivial, to demystify the situation an analysis on the keywords used in the subject of every book entry can be done to cluster books into genres and perhaps a definitive answer can be reached.

MySQL

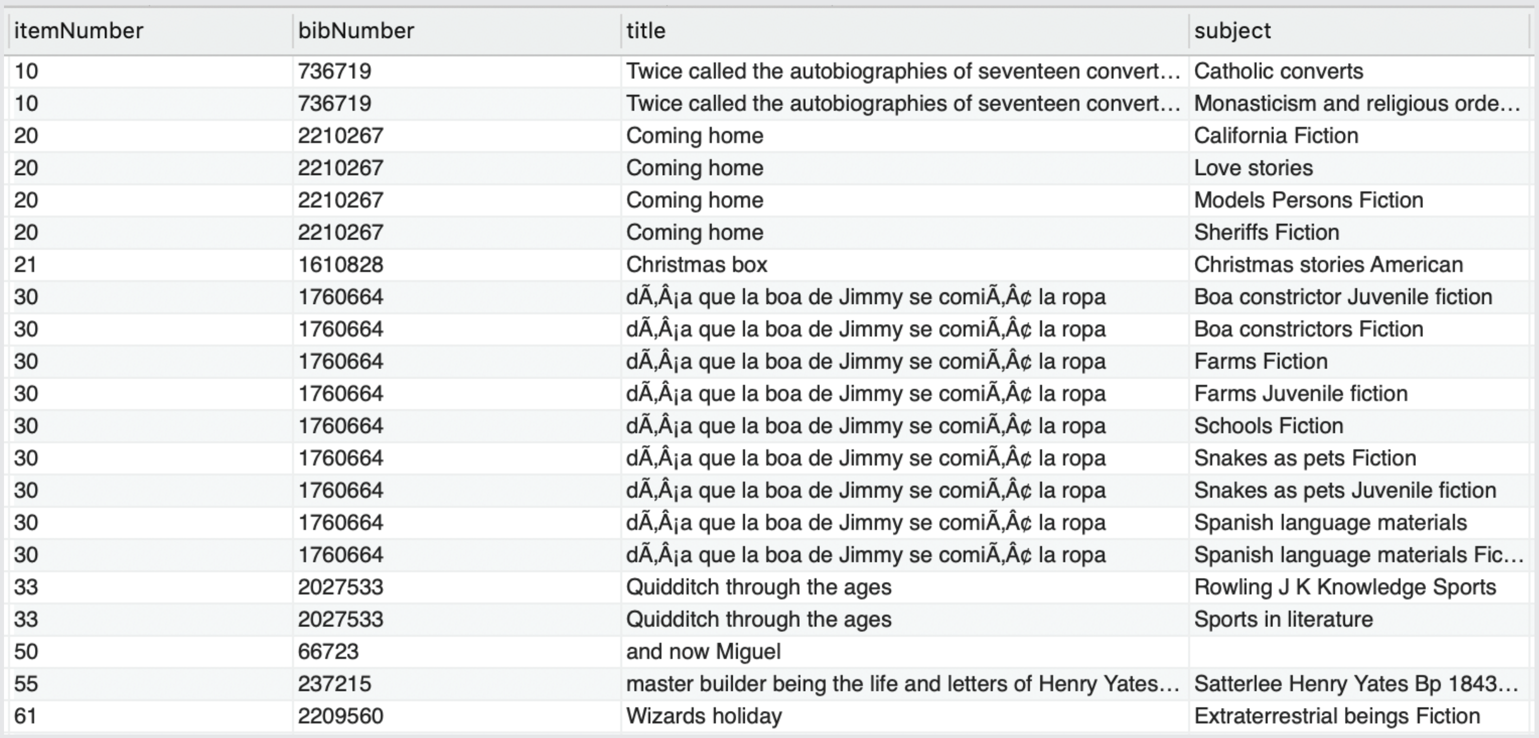
In order to do any analysis on the books and their subjects in the dataset, a subset of the books of interest had to be retrieved. The MySQL query needed to retrieve a dataset that included only books, so only those items in the dataset with itemType equal to acbk, arbk, bcbk, drbk, jcbk, or jrbk. Further, only nonDewey items would be retrieved for the dataset. I initially thought that there should be no duplicate copies of books represented in the dataset, but since we are interested in the subject and the keywords in the subject are not standardized across bibNumber you cannot aggregate. However, this change in plan ends up being useful later as a sanity check. If the algorithm works to cluster books based on genre, than the same book with slightly different keywords should end up very close together (the idea of distance in this space will be described later).
With all of these considerations the query below was developed:

```
SELECT DISTINCT
        typeNumBib.itemNumber,
        typeNumBib.bibNumber,
  title.title,
  subject.subject
FROM
  (
    SELECT itemType.itemType, itemToBib.itemNumber, itemToBib.bibNumber
    FROM itemType JOIN
    itemToBib ON itemType.itemNumber = itemToBib.itemNumber
    WHERE itemType.itemType IN ("acbk", "arbk", "bcbk", "drbk", "jcbk", "jrbk")
  ) AS typeNumBib
        JOIN deweyClass ON deweyClass.bibNumber = typeNumBib.bibNumber AND
                            deweyClass.deweyClass = " "
  JOIN title ON typeNumBib.bibNumber = title.bibNumber
  JOIN subject ON typeNumBib.bibNumber = subject.bibNumber;
```

*To spare future students pain, the documentation for the deweyClass needs to be changed. Nondewey Class Objects, deweyClass field are not set to NULL but to the empty string "".

Analysis

Above is the query to gather the nonDewey books, but answers can only be found with further analysis. I decided to use Python to do the analysis because I'm very familiar with the language and it offers the most options in the way of Machine Learning libraries (it also has its own advanced visualization libraries, but can easily be ported to Processing).
I began by figuring out how I was going to represent data in a usable way. This means taking qualitative data, descriptions about each



itemNumber	bibNumber	title	subject
10	736719	Twice called the autobiographies of seventeen convert…	Catholic converts
10	736719	Twice called the autobiographies of seventeen convert…	Monasticism and religious orde…
20	2210267	Coming home	California Fiction
20	2210267	Coming home	Love stories
20	2210267	Coming home	Models Persons Fiction
20	2210267	Coming home	Sheriffs Fiction
21	1610828	Christmas box	Christmas stories American
30	1760664	dÃ,Â¡a que la boa de Jimmy se comiÃ,Â¢ la ropa	Boa constrictor Juvenile fiction
30	1760664	dÃ,Â¡a que la boa de Jimmy se comiÃ,Â¢ la ropa	Boa constrictors Fiction
30	1760664	dÃ,Â¡a que la boa de Jimmy se comiÃ,Â¢ la ropa	Farms Fiction
30	1760664	dÃ,Â¡a que la boa de Jimmy se comiÃ,Â¢ la ropa	Farms Juvenile fiction
30	1760664	dÃ,Â¡a que la boa de Jimmy se comiÃ,Â¢ la ropa	Schools Fiction
30	1760664	dÃ,Â¡a que la boa de Jimmy se comiÃ,Â¢ la ropa	Snakes as pets Fiction
30	1760664	dÃ,Â¡a que la boa de Jimmy se comiÃ,Â¢ la ropa	Snakes as pets Juvenile fiction
30	1760664	dÃ,Â¡a que la boa de Jimmy se comiÃ,Â¢ la ropa	Spanish language materials
30	1760664	dÃ,Â¡a que la boa de Jimmy se comiÃ,Â¢ la ropa	Spanish language materials Fic…
33	2027533	Quidditch through the ages	Rowling J K Knowledge Sports
33	2027533	Quidditch through the ages	Sports in literature
50	66723	and now Miguel	
55	237215	master builder being the life and letters of Henry Yates…	Satterlee Henry Yates Bp 1843…
61	2209560	Wizards holiday	Extraterrestrial beings Fiction

Above is a sample of the data retrieved with the previous SQL query. It had to be dealt with in post processing. Mainly, that itemNumbers (i.e. the same physical items in the library) more than occasionally have the same bibNumber and title, but a different entered subject.

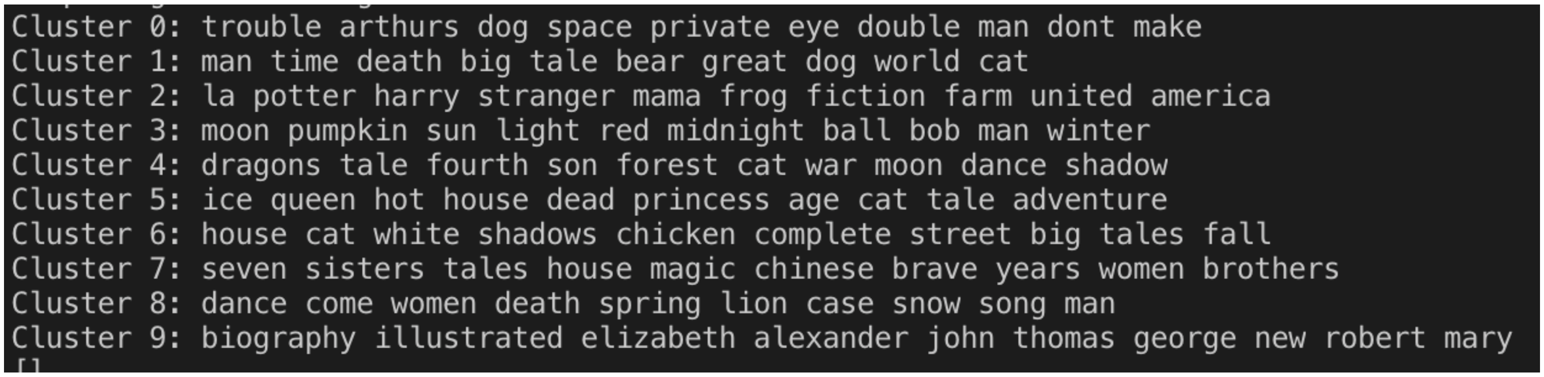
book, and figuring out how to represent it in a quantitative way. There are many Python libraries that help with this process, I ended up using Numpy, CSV, and Sklearn.
Numpy: A python library for doing complicated math and statistics problems
CSV: Allows you to read and write CSV files very easily
Sklearn: Python Machine Learning Library

I decided to use the KMeans Algorithm, which sorts data into clusters. This was in the hopes that similar subjects would result in genre clustering.

```
import numpy as np
import csv
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import MiniBatchKMeans

subject_item = {}
cluster_dictionary = {}

def addToDictionary(dictionary_name, itemID, tokenVector):
  dictionary_name.update({itemID : tokenVector})

with open('NonDeweyData.csv') as csv_file:
  csv_reader = csv.reader(csv_file, delimiter=',')
  line_count = 0
  for row in csv_reader:
      addToDictionary(subject_item, row[0], row[2])
      line_count += 1
      if line_count > 500000: break

vectorizer = TfidfVectorizer(max_df=.01, max_features=1000,
                     min_df= .001, stop_words='english',
                     use_idf=True)
mymodel = vectorizer.fit_transform(subject_item.values())

true_k = 10
km = MiniBatchKMeans(n_clusters=true_k, init='k-means++', n_init=1,
                   init_size=1000, batch_size=1000, verbose= True)
km.fit(mymodel)



order_centroids = km.cluster_centers_.argsort()[:, ::-1]
terms = vectorizer.get_feature_names()
for i in range(true_k):
   print("Cluster %d:" % i, end='')
   for ind in order_centroids[i, :10]:
      print(' %s' % terms[ind], end='')
   print()
```

Results

Based off the subject, I was able to create ten clusters that seems to be grouped based on genre. There are many parameters, that I plan on continuing changing for better results, but below are the top ten keywords in each cluster.



```
Cluster 0: trouble arthurs dog space private eye double man dont make
Cluster 1: man time death big tale bear great dog world cat
Cluster 2: la potter harry stranger mama frog fiction farm united america
Cluster 3: moon pumpkin sun light red midnight ball bob man winter
Cluster 4: dragons tale fourth son forest cat war moon dance shadow
Cluster 5: ice queen hot house dead princess age cat tale adventure
Cluster 6: house cat white shadows chicken complete street big tales fall
Cluster 7: seven sisters tales house magic chinese brave years women brothers
Cluster 8: dance come women death spring lion case snow song man
Cluster 9: biography illustrated elizabeth alexander john thomas george new robert mary
```

The next step is to pair the subject data points back with the titles, and come up with an accuracy heuristic. Due to time restrictions this will have to be part of future exploration.