Seforim Sale Data Analytics

Thesis Submitted in Partial Fulfillment
of the Requirements
of the Jay and Jeanie Schottenstein Honors Program

Yeshiva College

Yeshiva University

May 2022

Zechariah Rosenthal

Mentor: Professor Patricia Medina, Computer Science

Acknowledgements: The data collection, modeling, and research was done in partnership with Eli Perl. The work on the demographics, visualizations, and this written thesis are entirely my own work.

Motivation

Annually, YU students run the largest sale of Jewish books ("*seforim*") in North America. Endeavors such as this involve a lot of logistics and a lot of collected data, so I want to leverage that data both to increase efficiency of the business as well as to learn interesting information about the consumer body.

Outline

The goal of this Capstone is to complete an entire Data Science "lifecycle" project on the YU Seforim Sale and to produce a written report on the results. This required six steps:

1. Background: gaining a complete understanding of the business challenge. This involved learning how the Seforim Sale operates, what kind of data might be available, and interviewing Sale employees. I focused on two problems that can be solved with (somewhat related) data: inventory prediction, and demographic trends. These are described in greater detail below.

2. Data: acquiring and understanding the data. I designed and administered a survey of Seforim Sale customers (n=335), as well as gained access to their Shopify cumulative sales data from the past several years.

3. Cleaning: carefully preprocessing the data so it is in a usable form for automated modeling and visualizations. This is extremely detail oriented and depends on how the data I gather was organized.

4. Modeling: generating baseline, simple, and complex statistical models to predict desired values. I evaluated the model using validation data in the development environment. I primarily used modeling for the inventory prediction problem and attempted to predict 2022 sales.

5. Deployment: package the model and results into a reusable, transferable, application. I made an easy-to-use inference application that predicts future years' sales, given past years of data. I also built a Tableau data dashboard for the demographic data. Additionally, I will also be delivering the organized and cleaned spreadsheets of the demographics and inventory so that the Seforim Sale employees can use them.

6. Visualizations: present the data in an understandable, clear way that "tells a story." This very report will serve as the report of all my findings between the inventory prediction

and demographic trends. This will include all relevant graphs, diagrams, and other visualizations of the data. It will all be based on a Tableau dashboard which will be linked to in the project files.

Files

1. **Deliverable**: inventory prediction web app
2. Inventory cleaning and modeling notebook
3. Flask Inventory Prediction code and files
4. **Deliverable**: Tableau dashboard file
5. Demographic cleaning and modeling notebook
6. **Deliverable**: Final report (this doc)

Background

Here are some basic facts about the Sale's business:

- The Yeshiva University Seforim Sale is run entirely by YU students.
- It sells Hebrew and English Judaic books, as well as other related merchandise.
- The annual Sale runs for about 3 weeks in February.
- The Sale's staff consists of ~100 students, including ~65 salespeople.
- They claim to sell 6,500 unique titles (I found about 5 times more)
- They brought in $740,000 in revenue in 2020
- Over the duration of the sale, total shoppers reach approximately 15,000 people

Available books span genres including but not limited to:

- Bible
- Talmud
- Jewish Law and Practice
- Textual Study and Commentaries
- Jewish Philosophy and Thought
- Jewish History
- Cookbooks
- Children’s Books and Educational Materials

They use Shopify for their sales and have access to the past several years of sales' data.

Task 1: Inventory Prediction

This is the standard "inventory problem." It is hard for any business to know exactly what consumers want. They want to make sure to stock inventory sufficiently without overstocking. I used historical sales data to predict how many copies of each title, and of each genre as a whole, will sell in 2022 (or any future year). These predictions (if accurate) can greatly help the Seforim Sale improve their business and know how much to order in future sales.

Based on interviews with Seforim Sale executives, most ordering is done on an extremely case-by-case, ad-hoc basis. Often, good *sale* data from multiple years back isn't handy or aggregated, so they often just rely on the *ordering* data from the immediately prior year. Often, this leads them to simply order whatever quantity they sold last year.

I used simple baseline and complex statistical Machine Learning models, using *all* prior data I could access. This allowed us to pick up on broader trends across titles and genres.

Task 2: Demographic Trends

This is an interesting sociological and business research opportunity. It is interesting to identify what groups of people frequent the Sale and what their purchasing patterns are. I was interested in categories such as age, gender, place of residence, occupation, university attended (if any), undergraduate major (if any), how many books they bought, and which categories of book they bought.

Beyond just curiosity, there is also a business interest in knowing who your customers (primarily) are. This can perhaps lead to better catering to their main demographic, or recognizing unreached demographics who are not well served yet by the Seforim Sale.

Data

This stage took a lot of legwork to find the people who could help us find the data. Thankfully, the 2022 CEO of the Seforim Sale, Eli Seidman, was extremely helpful and cooperative. He gave us access to whatever records Shopify kept (excluding sensitive user data), as well as allowed us to administer the survey. They had data for 2015-2020 and 2022. There was no sale in 2021 due to COVID-19.

The parts of the Sale historical transaction records that we're interested in are:

- Book title
- Genre
- Vendor
- Quantity sold in year X

For the demographics, I had to make our own data from scratch. I designed a Google Form survey, found generous funding for a $100 Seforim Sale gift card raffle, and placed a link and a prompt to fill out the survey on all email receipts that the Sale sent. I got 335 responses. Part of the cleaning below investigates how *representative* that sample really is. I found it to be mostly very representative, except in a few genres of book-buyers.

Even though I designed the survey to be as clear as possible, it still required major cleaning and transformation work to get it visualizable.

The most important features I gathered are:

- Gender
- Age
- City
- University attended (if any)
- Major concentration in university (e.g. "Psychology")
- Occupation
- Number of books bought at the sale

Unfortunately, only about 219 people filled out the per-genre quantity of books bought, so I was unable to get clear data on that or how that correlates with other features. Nevertheless, I still have the per-genre distribution of sales overall from the Shopify data.

Cleaning

Let's walk through the cleaning and preprocessing for each Task. I will link directly to the relevant Colab cells under discussion.

Inventory Cleaning

The Sales data is separated by year. Recall that I had the years 2015 - 2020 and 2022. For our initial modeling, I used 2015-2019 as the training data, 2020 as the validation data, and 2022 as the test data. [Colab]

Concatenating all years together, I found our first cleaning job: genre. (In the DataFrame, the column is called "product_type"). There seem to be 116 unique values for genre, yet the current sale website lists only ~20. What happened? Let's look at the first 10 unique values:

> ['Acharonim', 'Achronim', 'Biogpraphy', 'Biographies', 'Biography', "CHildren's", 'Chassidus', 'Children', "Children's", 'Childrens' ,...

The subtle discrepancies between different spellings and formulations (not to mention the presence of numerous typos) gives the impression that there are way more unique categories than there actually are. To rectify this, I hard-coded a list of category names as they appeared in the 2022 Sale, and defined a function to transform each genre into the category which was syntactically "closest" (as defined by Levenshtein edit distance). [Colab]

I fixed some negative values for quantity and set them to zero.

I also added a "bucketized" quantity column with categories of {0-9, 10-19, 20-29, 30-39, ..., 490-499}.

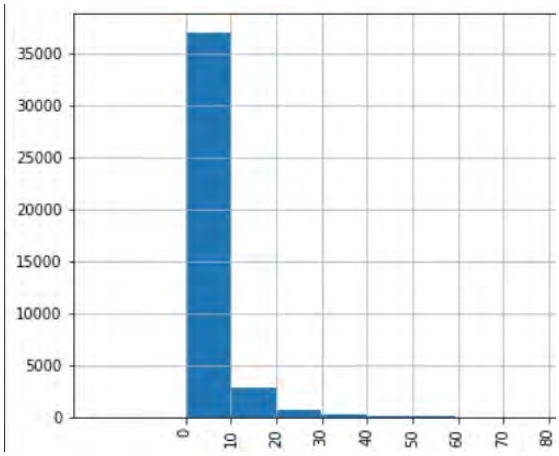
After a few other small fixes, here was what our data looked like: [Colab]

	product_title	product_vendor	product_type	net_quantity	year	quant_bucket
0	Imrei Baruch Minhagim	Rabbi Baruch Simon	Yeshiva University	218	2015	210-219
1	Headlines	Menucha	English Halacha	188	2015	180-189
2	CHUMASH MESORAS HARAV SHEMOS	Levitz	English Tanach	160	2015	160-169
3	SECRET RESTAURANT RECIPES	Artscroll	Cookbooks	153	2015	150-159
4	Sacks Haggada- HE/EN (28)	Koren	Haggada	142	2015	140-149

Inventory EDA

I used the handy Pandas Profiler library to get a quick profile of the current data frame. This gives histograms of each of the columns, alerts us to duplicate and extreme values, and other helpful features.

One key observation was that from the 41,267 samples (remember: I put each book per year in its own row), about 90% of them fell in the 0-9 bucket. This will make classifying them more difficult later on, given the category imbalance. [Colab]



Demographics Cleaning

I got 335 responses, and our survey had 40 questions (half of them were "How many of genre X did you buy?"). I split the data into the demographic focused features on one hand, and the book buying columns on the other. I re-merged the data afterwards.

I decided to run the profiler *before* cleaning this time to see what kind of work it needed. Age, Gender, How many books did you buy and other multiple choice or numerical columns were in good shape. It was the free answer text box questions of City, State, Occupation, University, and Major which were extremely messy. [Colab]

- City: initially there were 135 distinct cities (note: in a sample of 335). After putting them in lowercase and stripping whitespace I was down to 99. At that point I figured it would

be fastest to simply correct the typos by hand. This got us down to 84 actually unique cities [Colab]

- State: Similar to city, putting things all to uppercase resolved some duplicates. Thankfully, I had regex-ed the survey box to only accept exactly two letters as input, so there weren't any spelled out (or likely: mis-spelled out) states. However, curiously, there were some like "IS" and "ON" that are not US state codes. Looking at these individual cases they turned out to be Israel and Ontario. Although not conventional, I decided to leave the non-US state codes in, since they were consistent descriptors of their location.
- Occupation: Similar to city. Started with 113 distinct and ended with a whopping 20. I chose to consolidate jobs in a single field (e.g. "medicine", "business") so that there would be larger groups. [Colab]
- University: Similar to Occupation. Started with 94, ended with 45 distinct values. I chose to consolidate where I could (e.g. Stern College for Women, Syms, and Yeshiva College were all combined into "Yeshiva University"). Interestingly, some people put in the years they *attended* college here. With some sleuthing, I was able to deduce their universities. [Colab]
- Major: This was the trickiest. I started with 134 distinct and ended with 37. I had to make some subjective decisions for consolidating double majors and more unique majors into more general categories. For double majors, the more common of the two majors was selected. For specialized majors, the most-similar common major was selected. A few examples:
    - computer science / english lit. -> `computer science`
    - psychology and jewish studies -> `psychology`
    - biochemistry -> `biology`

After this cleaning, I was left with 13 duplicate rows, which I wasn't sure how to treat. I decided to leave them in since they represent someone making multiple trips to the Sale. [Colab]

Demographics/Buying Cleaning

Now I cleaned the buying data. Here's what the DataFrame looked like initially:

	Username	How many books did you buy at the sale?	How many of those books did you buy for yourself?	How many of the books did you buy for others?	Which of these types of books did you buy at least 1 of?	How many of each type of book did you buy? [Achronim]	How many of each type of book did you buy? [Biography]	How many of each type of book did you buy? [Chassidus]	How many of each type of book did you buy? [Children's]
0	hmc196@yahoo.com	3	3	0	Haggada;Yeshiva University	NaN	NaN	NaN	NaN

1 rows × 31 columns

There are more columns of the "How many of each type" questions.

I wanted to know what kinds of books different groups of people were buying. One potential pitfall was people who were buying books *not* for themselves or their household. Those kinds of

purchases may not have anything to do with their demographics. So in the survey, I asked three questions:

1. How many books did you buy at the sale?
2. How many of those books did you buy for yourself? Include all books you bought for you/your household.
3. How many of the books did you buy for others? Include all books NOT bought for you/your household.

The latter two should sum to the first one. In 14 samples, they did not sum to the total bought. I flagged these columns with a "these numbers are suspicious" flag and moved on. [Colab]

It is worth noting that 111 (33%) of people bought some books for others, but only 57 (17%) bought *more* books for others than themselves. This tells us that our per-genre data should be fairly reflective of demographic purchasing decisions. [Colab]

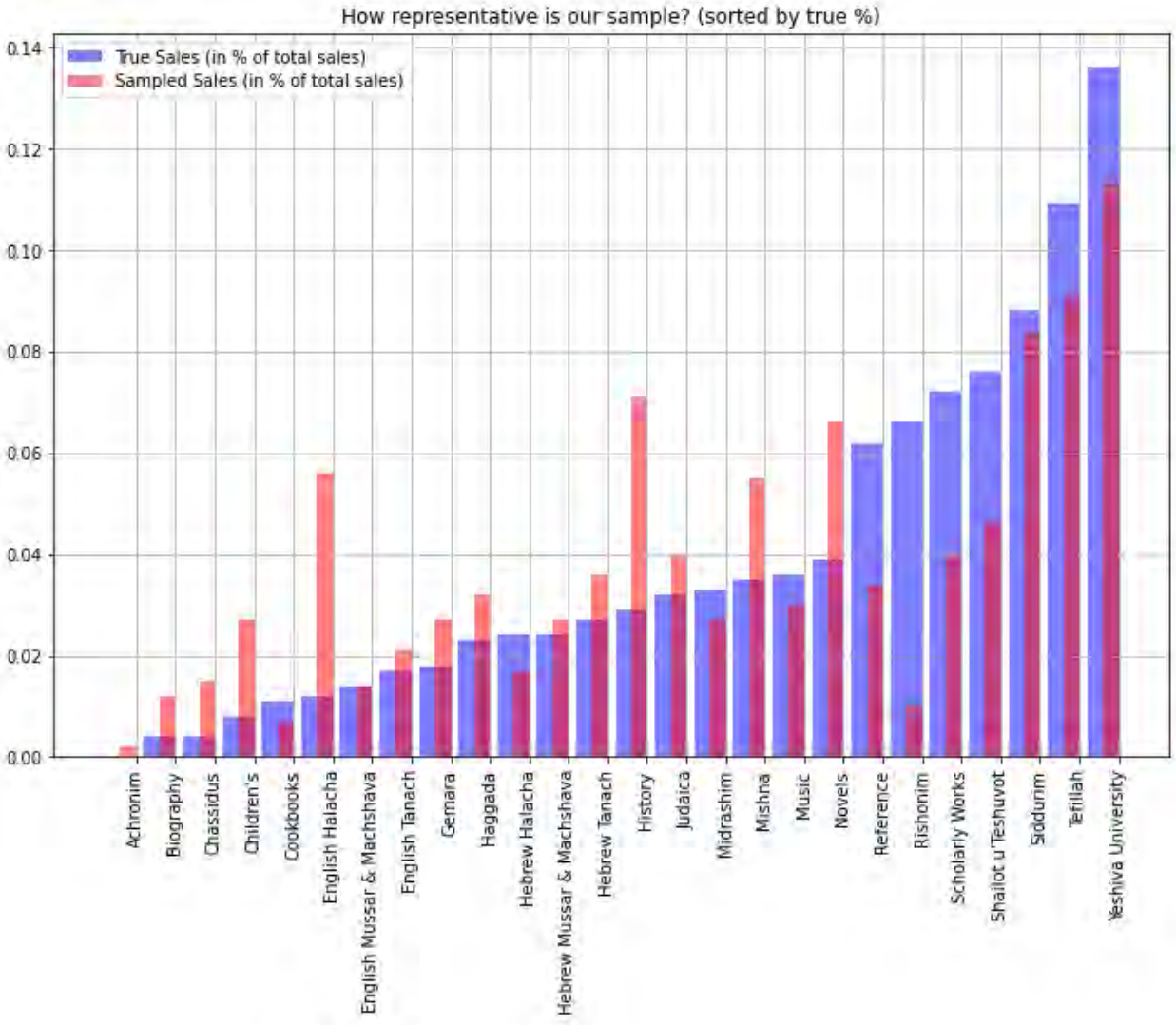
Was our survey representative?

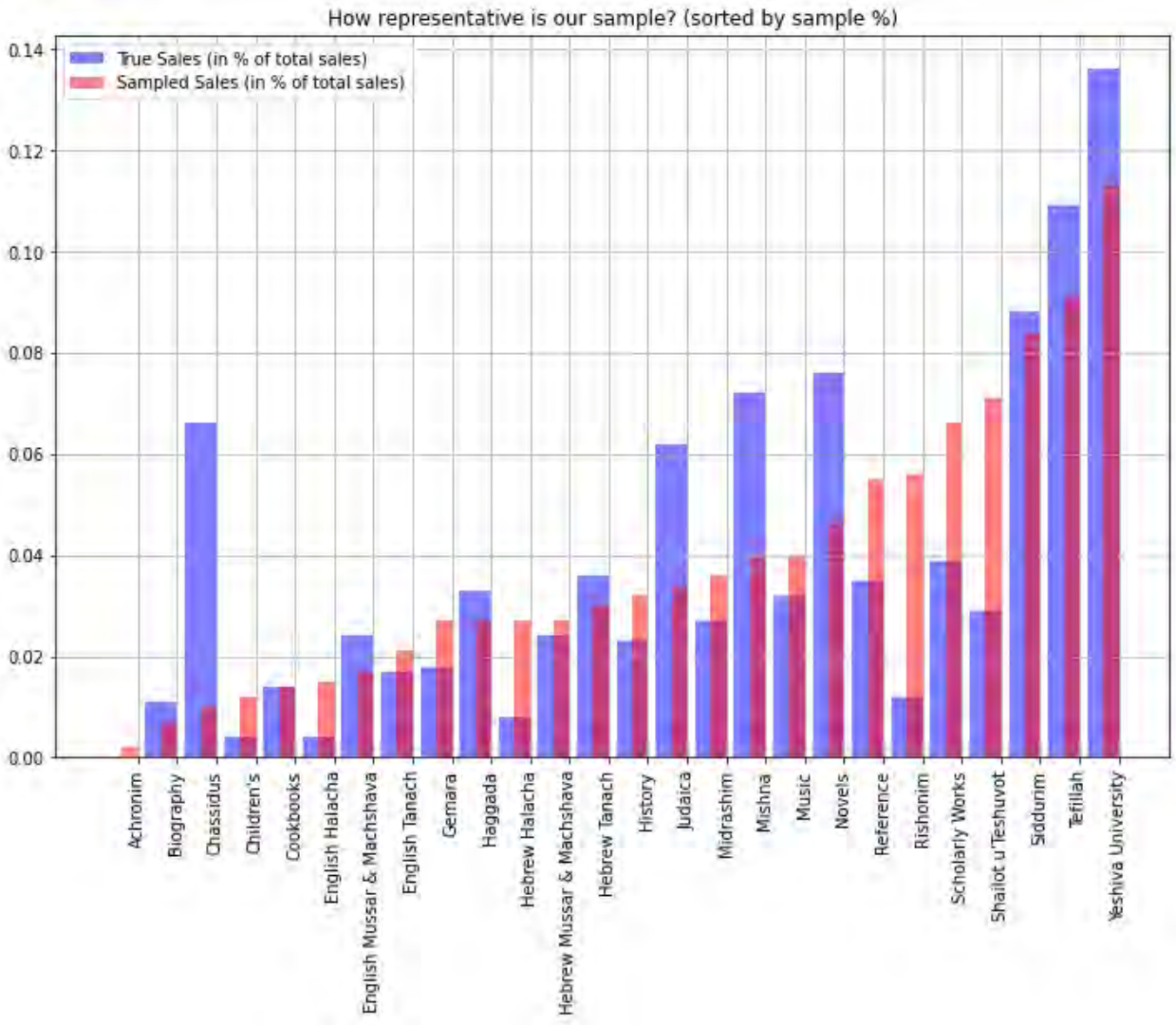
There was some other cleaning with the remaining columns, but the point is moot. As said above, only 219 out of 335 people bothered to fill out the per-genre questions correctly, which wasn't enough to give us robust data. When I designed the survey, I decided to make that section optional, to encourage people to finish the survey even if they didn't feel like itemizing their purchase. If I had made it a required field, those same people would probably have just clicked random options to get through the survey faster and I would have the *illusion* of good data.

However, 219 samples is better than nothing, and I was able to use that information to see how representative our sample was.

I didn't have access to any per-customer data from the Seforim Sale's Shopify, which would have made checking our sample simpler. Instead I decided to compare the percentage share of each genre's sales in both the real 2022 Sale and our sample.

I first got the per-genre Sale data for 2022, and normalized the quantities to add to 1. [Colab] I aggregated the quantity sold from the 219 samples per genre and normalized that too. I then put together a pair of overlapping bar graphs. The first sorts by true amount sold, and the second sorts by the amounts sold in our sample. The discrepancy between the overlap shows how biased our sample is. The graphs are reproduced below. [Colab]


How representative is our sample? (sorted by true %)
True Sales (in % of total sales)
Sampled Sales (in % of total sales)
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
Achronim
Biography
Chassidus
Children's
Cookbooks
English Halacha
English Mussar & Machshava
English Tanach
Gemara
Haggada
Hebrew Halacha
Hebrew Mussar & Machshava
Hebrew Tanach
History
Judaica
Midrashim
Mishna
Music
Novels
Reference
Rishonim
Scholarly Works
Shailot u'Teshuvot
Siddurim
Tefillah
Yeshiva University



Again, we're working with an n=219 subset of our sample responsible for 1645 of the sales (out of 2598, 63%), so it's not really that robust. Nevertheless, there's pretty even overlap between the true and sample values in a majority of the categories. More work could be done to more precisely measure the representativeness of the sample.

Modeling

As I explained in the Outline, I primarily used modeling for the inventory prediction problem and attempted to predict 2022 sales. I generated baseline, simple, and complex statistical models to predict future sales. I then evaluated the model using validation data from 2020 in the development environment, and iterated from there.

Baseline models [Colab]

For our first pass at a predictive model, I assessed how different statistical features of the data perform as predictors for future inventory demand. After I cleaned the data, I separated our "training" data (2015-2019) and used it to generate a mean, median, and linear regression values. I did this on a per-title and per-genre basis. This yielded the following DataFrame heads:

	2015	2016	2017	2018	2019	mean	median	lin_reg
Hebrew Writers on Writing	0	0	0	0	1	0.2	0.0	0.8
HAGGADAH TREASURY (Hard cover)	0	0	0	2	0	0.4	0.0	1.0
Peshat Isn't So Simple	15	15	16	10	12	13.6	15.0	10.3
משניות זכר חנוך, גדול - יכין ובועז מסודר מחדש	3	1	0	0	0	0.8	0.0	-1.3
Reasonable Doubts	1	2	0	0	0	0.6	0.0	-0.6

	2015	2016	2017	2018	2019	mean	median	lin_reg
Achronim	480	390	514	324	325	406.6	390.0	293.8
Biography	560	529	750	610	636	617.0	610.0	686.9
Chassidus	0	0	0	0	1062	212.4	0.0	849.6
Children's	2138	2157	1869	2173	2722	2211.8	2157.0	2567.0
Cookbooks	1052	1094	792	927	700	913.0	927.0	651.7

Next, I assessed how well these statistics approximate the actual quantities sold in the validation year (2020) using Mean Squared Error.

Per-Title Projections:

	mean	median	lin_reg
MSE:	30.02	28.68	53.09

Metrics for Category Projections:

	mean	median	lin_reg
MSE:	97532.45	98191.17	155980.33

Hyperparameter Tuning

I next optimized the above process by seeing if performance varied depending on how many previous years were considered for a prediction, and which statistic performs best in each case. An interesting observation to note is that when a product has 0 sales in a given year, that doesn't necessarily mean that there was *no* demand for such a product - it likely only means that said product was not ordered for that year's Sale.

For this reason, in addition to the same zero-conscious analysis I did so far, I also produced the same statistics but disregarding years when 0 products were sold. This entailed ignoring any 0-values training years when calculating the mean, median, and regression line. Furthermore, in order to ensure our evaluation of these statistics weren't affected by years in which the quantity sold was influenced by availability rather than demand, I drop any product which is 0 valued in 2020 (our validation year).

First, I included all zeros. The index represents how many years back of data were considered. The values in the DataFrame are the MSE. Here are per-title and per-genre, respectively:

```
title_score_df.head()
```

	mean	median	lin_reg
2	36.419077	36.419077	165.923355
3	33.589498	27.532305	91.988738
4	30.868245	27.249314	63.706015
5	30.026996	28.689347	53.094757

```
cat_score_df.head()
```

	mean	median	lin_reg
2	115940.517241	115940.517241	361058.931034
3	93621.670498	128978.241379	273824.509579
4	92584.017241	105073.422414	204554.905172
5	97532.459310	98191.172414	155980.336897

Next, I disregarded all zeros, as explained above. Here are per-title and per-genre results, respectively:

```
title_score_df.head()
```

	mean	median	lin_reg
2	135.741236	135.741236	188.347590
3	136.926979	134.715644	173.920187
4	137.464265	134.658107	179.360248
5	137.727390	134.246494	198.953668

```
cat_score_df.head()
```

	mean	median	lin_reg
2	114017.931034	114017.931034	306567.344828
3	86869.471264	105682.896552	263617.747126
4	82518.971983	81778.077586	204550.318966
5	85216.365517	74895.827586	157638.698276

It appeared that for title projections, the optimal inference range is four years, and the ideal statistic is the median (MSE: 23.74). For categories, four years and mean (MSE: 92584.01). Furthermore, the results were demonstrably worse when ignoring zeros, surprisingly.

Decision Tree Regression Model [Colab]

I used the same data as the baseline, but I also included the genre labels in addition to the year-to-year sales quantity data. I label-encoded the different genres. Since I used a decision-tree model, I didn't have to be so concerned with the model incorrectly inferring an ordinal relationship between the genres.

Since I was training a time-series model to predict future outcomes, I needed to transform the data into training data and regression targets such that it could learn to predict future data from past inputs.

To do that, I determined that each "sample" will consist of 3 feature groups:

1. Data from two years prior to the predicted year, including:
    a. A year label (e.g., 2015, if the regression target is for 2017)
    b. Quantities of all of the products sold in that year (0 if not sold)
2. Data from one year prior to the predicted year (i.e. same thing for 2016 if the regression target is for 2017).
3. The encoded category label for each product

The regression target consisted of the quantities of all the products sold during that year (e.g., 2017). The model was then trained on all of the available samples paired to their regression

targets, such that the model predicted the quantity sold from the quantities of the two years prior. This produced a 4-rowed DataFrame that looked like this:

time_series_df

	0	1	2	3	4	5	6	7	8	9	...	87408	87409	87410	87411	87412	87413	87414	87415	87416	87417
0	2015	0	0	15	3	1	0	0	0	0	...	0	0	0	1	0	0	0	2	0	0
1	2016	0	0	15	1	2	0	0	0	2	...	1	0	0	0	0	0	1	3	0	2
2	2017	0	0	16	0	0	0	5	0	0	...	0	2	0	0	0	0	0	0	13	0
3	2018	0	2	10	0	0	1	0	11	0	...	2	0	0	0	0	0	0	0	0	0

4 rows × 87418 columns

After training, the per-title MSE was 43.17 and the per-genre MSE was 142,298.93.

Hyperparameter Tuning

Since I couldn't perform folded cross-validation (which would corrupt the necessary ordering of time-series validation data), I defined our own function for grid-search hyperparameter tuning. [Colab] For per-title, the best MSE score was 29.60 and for per-genre was 86,285.07. A great improvement!

Here's a few summary statistics of the per-title prediction. The median book (across over 40000 titles) was predicted to be <1, implying it should not be ordered.

mean	1.30
std	4.21
min	0.00
25%	0.00
50%	0.33
75%	1.33
max	165.66

Classification Model

Next I tried a totally different approach. Since I really only needed a rough estimate of the quantity rather than an exact prediction, instead of treating this as a regression problem I tried viewing it as a classification problem using XGBoost as our classification model.

Using the quantity buckets I defined earlier, I trained a model which learns which bucket a product should be assigned to based on its title, genre, vendor, and year.

This produced an accuracy for 2020 of 85.76%. After hyperparameter tuning, the accuracy was up to 88.97%. (Note: before I was using the Mean Squared Error as the evaluation metric since it was a regression problem. Here, I use accuracy since it is a classification problem.)

The model seemed to be remarkably accurate, but it's actually terribly misleading. As I saw in our EDA, 88.9% of all products sold 0-9 units (and thus would be labeled as being in the 0-9 quantity bucket). If our classifier blindly labels every single product as belonging in the 0-9 bucket (which it is doing), it would be 88.9% accurate.

To account for this possibility, I tuned the model again, but this time optimizing for weighted F1 score to account for the class imbalance and optimize precision and recall. This led to a F1 score of 0.83. However, the decent F1 scores belied the model's total underfit of the classification problem. Here's the summary statistics of its title predictions:

mean	1.0
std	0.0
min	1.0
25%	1.0
50%	1.0
75%	1.0
max	1.0

Once again, the model blindly assumed all input belongs in the ```0-9``` bucket. The data is so imbalanced that even taking a weighted F1 score as the optimization metric didn't result in quality learning. A possible solution would be to resort the data into buckets of varying sizes which would distribute the data more evenly; however, doing that effectively would likely result in a terribly overfitted model that would perform terribly on new input.

Hybrid Model

So far, our most reliable predictors for title and category sales respectively have been:
Title Sales:

- Median with four years of inference data
- Time-series decision tree regressor with two years of inference data

Category Sales:

- Mean with four years of inference data
- Time-series decision tree regressor with two years of inference data

Since the business problem is best served by a range of possible values rather than a specific prediction, I designed a hybrid model to use the two best models for each use case to predict a range of possible sales quantities.

Title Hybrid Model Performance:

Accuracy = 72.57%
Range STD = 3.26

Category Hybrid Model Performance:

Accuracy = 10.34%
Range STD = 96.94

While it's far from perfect, our hybrid model seemed to do well enough in terms of predicting a reasonably precise range of product sales (even if it's pretty terrible for predicting quantity per category). I believe that I have reached the performative upper bound with this model considering the scarcity of data, and for this reason we're opting to incorporate this hybrid model architecture into our productionized deployment.

If I had more years of data, perhaps our decision tree regression model could be better trained, or I would be able to look into more sophisticated models for time-series prediction like LSTMs. Furthermore, if there was a more even distribution of quantities, perhaps I could get better performance from both our regression and our classification models. Lastly, if there was more information actually describing the products themselves, perhaps I could leverage embeddings or other NLP techniques to learn patterns in the data based on semantic meaning as well.

Deployment

In order to hand off our results to the Seforim Sale, I needed to take it out of the specialized Colab format and package it in a robust application that anyone could use. So I made an easy-to-use inference application that predicts future years' sales, given past years of data. It's a simple Flask web app, with the past years of data hard coded in. Future work could enable the ability to upload more (future) years of data. Here's the link to the deployed web app. It looks like this:

I also built a Tableau data dashboard for the demographic data. This can be found here.

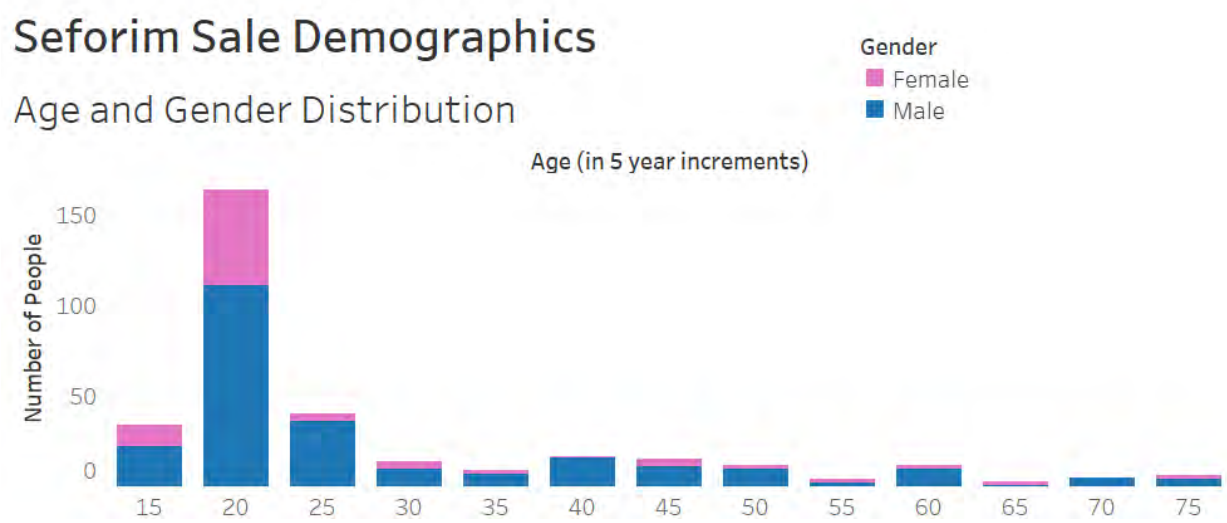
Additionally, I also delivered the organized and cleaned spreadsheets of the demographics and inventory so that the Seforim Sale employees can use them for their own purposes.

Visualizations

I needed to present the data in an understandable, clear, way that "tells a story." This very report will serve as the report of all our findings between the inventory prediction and demographic trends. This includes all relevant graphs, diagrams, and other visualizations of the data.

Here are some the graphs I put together from the demographic data, as well as some of the design choices in making these visuals:

Visual 1



If I had to boil down the question this visual answers, it would be: "Who is coming to the sale?" The most basic information about people is their age and gender. How should I show this data? I chose to display Age in a binned bar-chart histogram, with gender represented as stacked bars, with a color difference, in each bin.

On a graphic design note, I chose to use the easy and common convention of coloring pink for Female and blue for Male. Pink and Blue are also extremely contrasting, so even in the bins where there is a small percentage of Female samples, the thin strip of hot pink is visible. Even though I want it to be immediately visually obvious what it represents (through clear titling of the sheet), I put a small legend unobtrusively in the corner to detail the precise meaning of the colors.

Additionally, I chose to bin the histogram in 5-year increments. That is, 15-19 is the first bin, 20-24 the next, and so on. Binning makes the histogram much less cluttered, and can allow for a clearer view of the skewed distribution. Choosing a number like 5 for the bin size, which has extremely simple multiples (end with 0, end with 5, end with 0) eases the digestion of the large amount of numerical data.

Visual 2

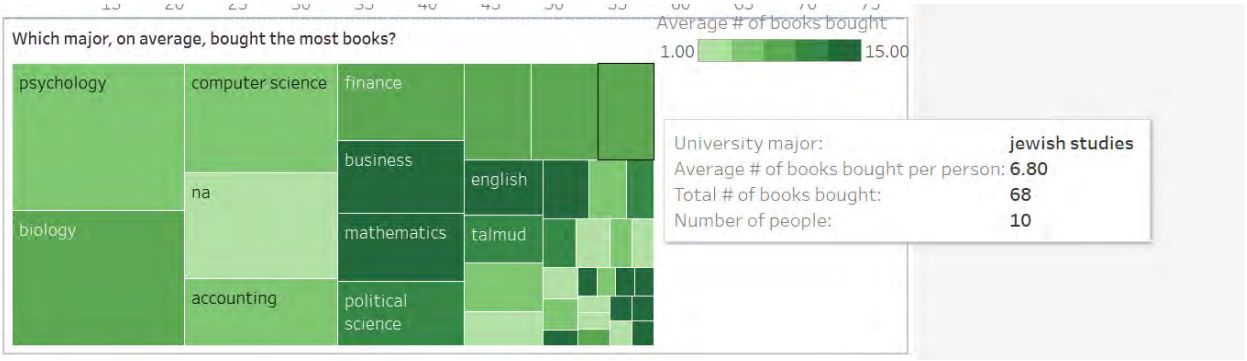
The central question of this next visual is: "Which kinds of students/people buy the most books?" This question deliberately had two connotations: 1) "Which kind of student buys the most books *on average*?" and 2) "Which kind of students buy the most books *in total*?"

So I need to consider:

1. Grouping by Major
2. Total number of samples in each major
3. Total number of books bought by people in that major
4. Average number of books per person, for people in that major

That's a lot of information! However, I get the sense that they're related, so perhaps I can combine them in innovative ways.

I decided to represent this visual with a colored tree map, with more information available interactively as hover-text. Here's what it looks like:



Let's see how all 4 types of information are provided in this one visual, without sacrificing clarity.

The Major title (1) is printed as a title in a contrasting font color on the larger boxes. For smaller boxes (as in the image above), the major is available in the hover-text.

The size of the box represents the relative number of people in that major (2). The precise number of people is available in the hover text.

The color/shade of the box represents what the average number of books the people in that major bought. (4) The darker, the higher the average. I included a legend to clarify what the color represents. Instead of a smooth color spectrum, I chose a discretized 5 colors so that you can actually identify a box with, e.g., "the 4th darkest."

I also bound the scale at an upper limit of 15. There was a single sample in the "electrical engineering" major, and that person bought 85 books! This outlier made sure his category bought an average of…85 books. If I would have used the default spectrum with the upper limit of 85, all the diversity of the 1 to 15 range would've been squashed into a single color. I decided to tradeoff having the largest majors be differentiated and "misrepresent" the outlier as merely an average of 15. Either way, the precise averages are always available on hover-text.

I deliberately chose green for two reasons. One, I didn't yet use it in any of the visualizations. If I had used blue, some may have confused that from the previous Sheet as representing Male. Second, this visualization shows the size and books-per-person of each group. Those two numbers multiplied represent the total number of books, which is directly correlated with the monetary revenue. The bigger and darker the green, the more money is to be found in that group! This emphasizes the more financial side of this visual subtly.

This combined visual metric of darkness X area is actually just the "total number of books bought by people in this major." (3) Of course, the specific number is available in hover-text if the user wants it.

I successfully encoded all 4 pieces of information in a non-cluttered way. Sometimes these more "novel" visualization techniques of tree map can be very useful and effective! It also encourages interaction since not all the precise data can be seen without hovering over the various boxes. This is a great example of the "Overview + Detail" paradigm of data visualizations. The static big picture gives the total gist, and specific, user-driven actions reveal all the details they may want.

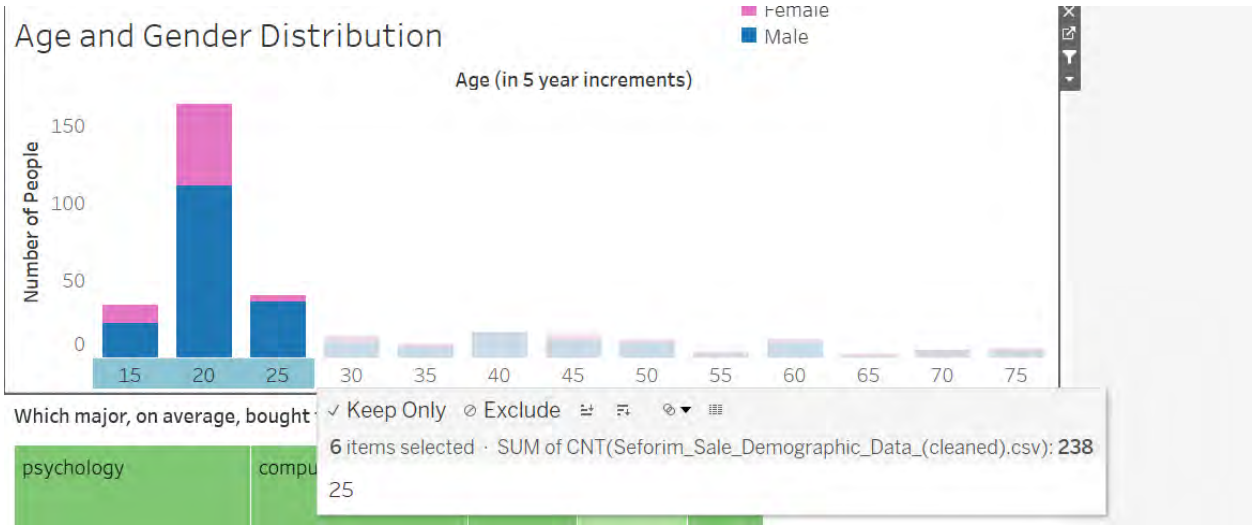
Interactivity

The strength of using Tableau for this dashboard is the interactivity. Tableau has an extremely clear interface design, with different actions for hovering, clicking, dragging. This empowers viewers to engage deeply with the data, ask their own questions of it, and see the dashboard reorganize itself for them.

The key tool here is cross-filtering. By selecting any subset of the sample in one graph, this automatically causes the other graph to rearrange to only represent that sample too. Let's walk through a few questions a hypothetical user could ask, and how they would find the answer.

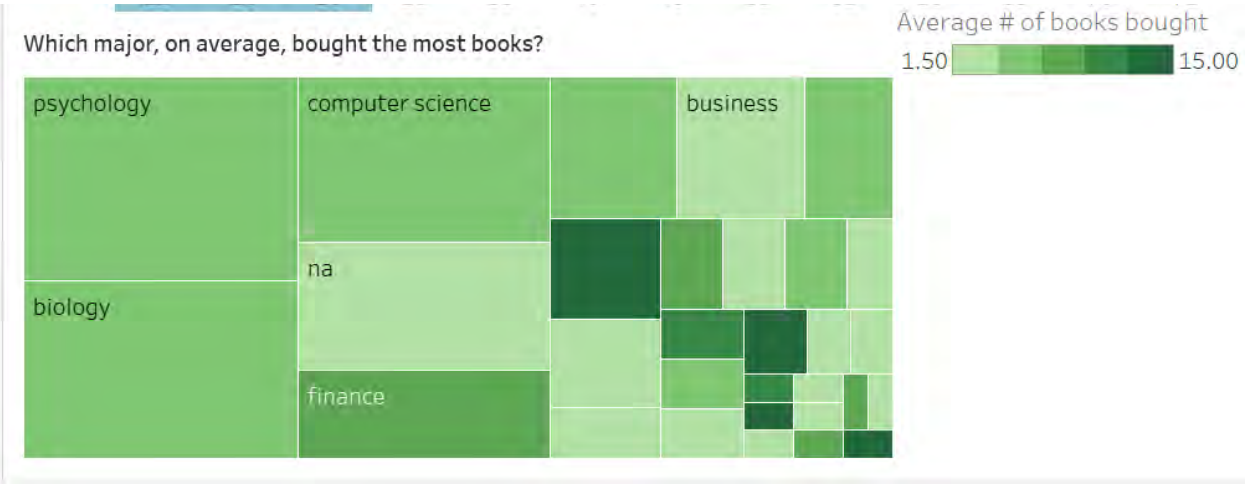
How many people under 30 came to the sale?

Simply drag to select "15" "20" and "25", and the hover text will sum them and give the answer:



What are the spending patterns of under 30s?

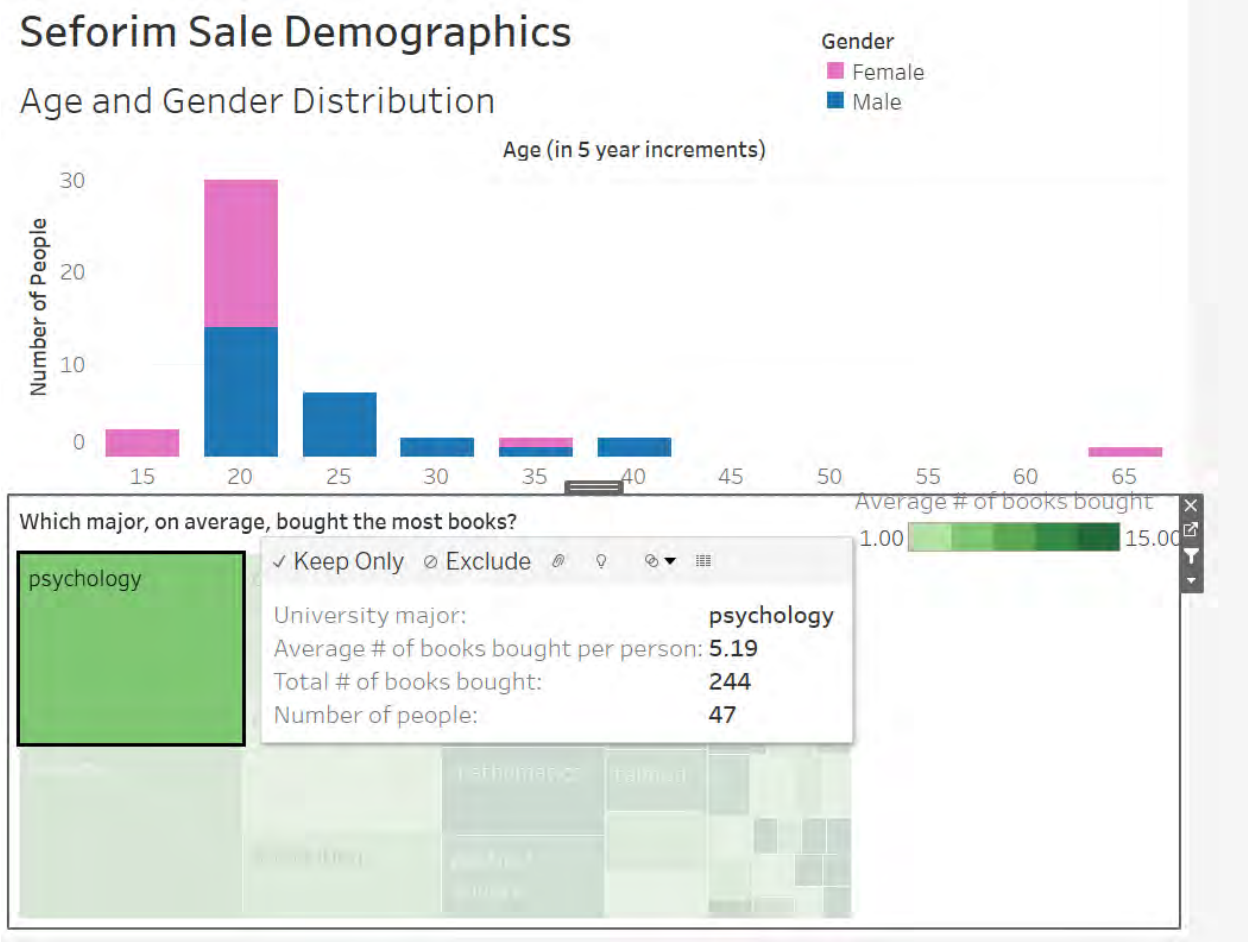
Now that I have this subset of the sample selected, let's see how it has filtered the other graph:



Compared with above, there are many fewer large, dark boxes. This means our subset (under 30s) bought fewer books on average than the older crowd. This makes a lot of sense! Of course older people, further along in their career, will have more disposable income.

What are the demographics of Psychology majors?

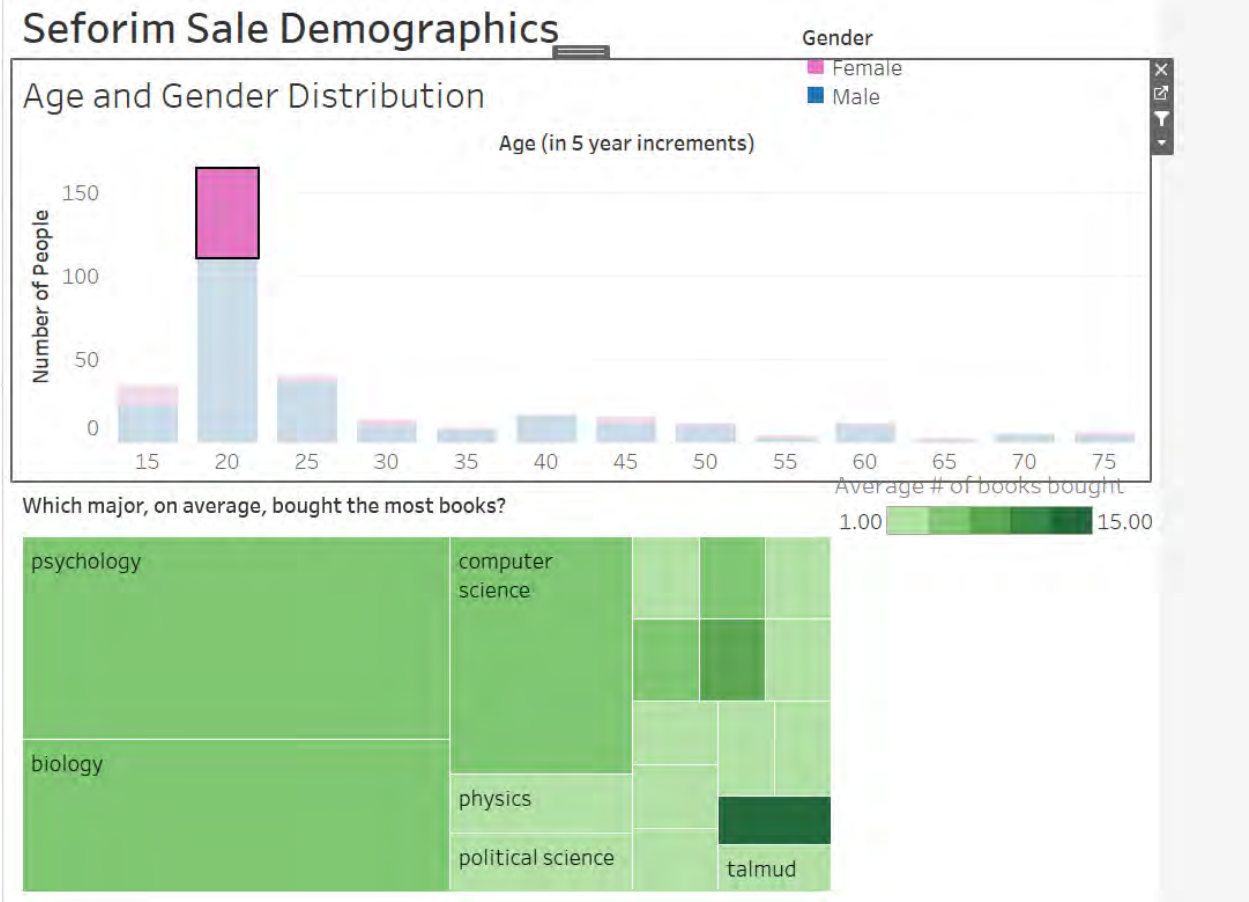
Let's clear our filter by simply pressing the "Esc" key and make a new filter. Let's click on the psychology major box, and see how the Age and Gender graph changes. We'll also include the detailed hover text of the psychology major:



As I can see, whereas the original sample was ~30% Female, this sample is closer to ~50% Female (at least in the 20-24 bin). This makes sense, as the national gender split of Psychology majors heavily leans towards Female.

What's the Major breakdown of the young, Female demographic?

I can select just part of the stacked bar graph to filter by just Female samples of a specific age. This will give us an entire fresh Major/average books bought graph below:



Looks like the biggest samples of young women are all in STEM fields! YU's marketing department may be interested in this finding.

Other

Here are some other noteworthy results from our work on the demographic dataset.

- 35% of our sample attended graduate school. The base rate for Americans is 12%.

Here are the "total number of books bought" summary stats:

- Total number of books bought: 2598
- Total number of people in our sample: 335
- Mean: 7.75 books/person
- 25% of people bought between 1-2 books
- 25% of people bought between 2-4 books
- The median person bought 4 books
- 25% of people bought between 4-8 books
- 25% of people bought between 8-145 books

A full **50%** of the Seforim Sale's sales come from the top **25%** of customers.

Here is the University distribution:

Value	Count	Frequency (%)
yeshiva university	226	67.5%
na	28	8.4%
touro	9	2.7%
brandeis university	6	1.8%
queens college	6	1.8%
city college of new york	4	1.2%
brooklyn college	3	0.9%
barnard college	3	0.9%
lander college for men	3	0.9%
new york university	2	0.6%
Other values (35)	45	13.4%

Here is the Major distribution:

Value	Count	Frequency (%)
psychology	47	14.0%
biology	43	12.8%
computer science	31	9.3%
na	30	9.0%
accounting	19	5.7%
finance	18	5.4%
business	17	5.1%
mathematics	16	4.8%
political science	15	4.5%
history	12	3.6%
Other values (27)	87	26.0%

Conclusions

I successfully shepherded this project through the six stages of a data science lifecycle. I produced three key deliverables: a web app for inventory prediction, a tableau data dashboard, and this written report.

While it's far from perfect, our hybrid model seemed to do well enough in terms of predicting a reasonably precise range of product sales (even if it's pretty terrible for predicting quantity per

category). I believe that I have reached the performative upper bound with this model considering the scarcity of data, and for this reason we're opting to incorporate this hybrid model architecture into our productionized deployment.

If I had more years of data, perhaps our decision tree regression model could be better trained, or I would be able to look into more sophisticated models for time-series prediction like LSTMs. Furthermore, if there was a more even distribution of quantities, perhaps I could get better performance from both our regression and our classification models. Lastly, if there was more information actually describing the products themselves, perhaps I could leverage embeddings or other NLP techniques to learn patterns in the data based on semantic meaning as well.

While I did find out some insightful results from the demographics, I did not receive robust per-genre purchasing data as I hoped. Future work could go into collecting that data and seeing if any interesting buying patterns fall out of it.

Bibliography

1. The Team Data Science Process lifecycle, Microsoft. https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/lifecycle
2. YU Commentator. https://yucommentator.org/2022/02/behind-the-scenes-of-the-yu-seforim-sale/. February 2022.
3. Python Data Science Handbook, J. VanderPlas, O'Reilly (2016)
4. Hands-On Machine Learning with Scikit-Learn, Keras & Tensorflow (2nd ed.), A. Géron, O'Reilly (2019)
5. The Elements of Statistical Learning (2nd ed.), T. Hastie, R. Tibshirani & J. Friedman, Springer (2011)
6. The Data Science Design Manual, S. Skiena, Springer (2016)
7. Data Visualization: A Practical Introduction, K. Healy, Princeton University Press (2018)