UNIVERSITY OF TENNESSEE

Mini-Project 2: Dunnhummby Shoppers' Missions

Joshua Fagan, Alex Klibisz

October 27, 2016

The Dunnhumby dataset contains product-level transactions over two years from customers in 2,500 households; the specific retail context is not disclosed (i.e. grocery store, department store, convenience store). All transactions can be linked to a specific household and household demographics such as income or number of children are available. We explore this dataset using the Latent Dirichlet Allocation (LDA) and Author-Topic (AT) graphical models with the goal of understanding the latent missions of shoppers. In our analysis, both models yield distinct, yet reasonable categorizations of products into topics. We find the model performance increases after employing a low-entropy naming scheme. The model results are useful for answering several questions about the customers. Through further statistical analysis, we are able to determine which demographic classifications are *most different* from the entire population.

1 INTRODUCTION

Our goal is to use graphical topic modeling techniques to identify latent missions for shoppers in the Dunhumby dataset. Further, we strive to link missions to demographics and/or product categories. Some concrete questions we seek to answer are as follows:

1. Can we group products into distinguishable categories (topics)?
2. Can we identify demographic attributes that distinguish specific shoppers from the rest of the population?
3. Can we identify a temporal pattern in shoppers' topics?
4. Can we show how the distinguishing attributes can be leveraged for more effective marketing campaigns?

Our work is presented as follows. In section 2 we introduce the data and highlight some important observations. In section 3 we introduce the *low-entropy naming* technique which improves the quality of our product-topic grouping. In section 4 we describe our LDA implementation, the perplexities for various topic sizes, interpret topics manually, and compare demographic-specific topic distributions against the population distribution. TODO Josh talk about your stuff here briefly.

2 Data Review

The Dunhumby dataset[1] contains product-level transactions over two years from customers in 2,500 households; the specific retail context is not disclosed (i.e. grocery store, department store, convenience store). All transactions can be linked to a specific household and household demographics such as income or number of children are available. The entire Dataset consists of the tables: CAMPAIGN_TABLE, CAMPAIGN_DESC, COUPON_REDEMPT, COUPON, HH_DEMOGRAPHIC, CAUSAL_DATA, PRODUCT, TRANSACTION_DATA.

Our modeling and analysis draws only from product information, transaction records, and demographic classifications in the PRODUCT, TRANSACTION_DATA, and HH_DEMOGRAPHIC tables, respectively. This includes 2,595,732 unique transactions, 276,484 unique shopping baskets, 92,339 unique products, and 2,500 unique households. The first five rows of each table are provided in tables 2.1, 2.2, and 2.3. The remainder of this section will summarize relevant relationships in this data and highlight two interesting characteristics.

PRODUCT_ID	COMMODITY_DESC	SUB_COMMODITY_DESC
25671	FRZN ICE	ICE - CRUSHED/CUBED
26081	NO COMMODITY DESCRIPTION	NO SUBCOMMODITY DESCRIPTION
26093	BREAD	BREAD:ITALIAN/FRENCH
26190	FRUIT - SHELF STABLE	APPLE SAUCE
26355	COOKIES/CONES	SPECIALTY COOKIES

Table 2.1: Selected columns from the PRODUCT table.

household_key	BASKET_ID	PRODUCT_ID	QUANTITY
2375	26984851472	1004906	1
2375	26984851472	1033142	1
2375	26984851472	1036325	1
2375	26984851472	1082185	1
2375	26984851472	8160430	1

Table 2.2: Selected columns from the TRANSACTION_DATA table.

household_key	AGE_DESC	MARITAL_STATUS_CODE	INCOME_DESC
1	65+	A	35-49K
7	45-54	A	50-74K
8	25-34	U	25-34K
13	25-34	U	75-99K
16	45-54	B	50-74K

Table 2.3: Selected columns from the HH_DEMOGRAPHIC table.

[1] https://www.dunnhumby.com/sourcefiles

2.1 Understanding Relationships

The relevant relationships in the dataset are as follows: Every product in PRODUCTS has a unique identifier, PRODUCT_ID. The HH_DEMOGRAPHIC table contains a row for each household, identified by the unique identifier, household_key. Every transaction in TRANSACTION_DATA comprises a basket and a product tied to a specific household, identified by BASKET_ID, PRODUCT_ID, household_key, respectively. Intuitively, there exist many baskets across all transactions, many products in a basket, and a single household attributed to each basket. Based on this, we can determine the specific products a household purchases at a per-basket level.

2.2 Products, Baskets, Transactions in the Context of Topic Modeling

We map the Dunhumby data to the context of graphical topic modeling as follows. The corpus is all baskets across all transactions. The documents are the baskets. The words are products contained in each basket. Words can be represented by various identifiers, discussed in this section. Authors can be identified by using the household_key or grouping into demographic classifications (e.g. all households with 2 children are a single author).

2.3 Identifying Unique Products

The PRODUCT table contains unique identifiers, PRODUCT_ID for products. Intuitively, this should be the means by which we identify a unique product. However, after evaluating model output, we found that there are actually many otherwise indistinguishable products that have different PRODUCT_ID properties.

For example, products 1029743 and 1106523 both have COMMODITY_DESC = "FLUID MILK PRODUCTS", SUB_COMMODITY_DESC = "FLUID MILK WHITE ONLY", MANUFACTURER = 69, and CURR_SIZE_OF_PRODUCT = "1 GA". In other words, these products are indistinguishable (at least for our purposes), but are distinct when PRODUCT_ID is used as the identifier. This becomes an obvious issue when looking at the categorizations of products into topics - one sees the "same" product scattered across several topics. With this in mind, we identified two pragmatic alternatives for identifying unique products.

2.3.1 Identify by COMMODITY_DESC and SUB_COMMODITY_DESC

Instead of using the PRODUCT_ID, we can identify unique products using the COMMODITY_DESC and SUB_COMMODITY_DESC properties either separately or as a concatenated string. Using this tecnique, the prior example would combine both of the milk products as a single product.

2.3.2 Low Entropy naming

We will propose the concept of *low-entropy names* in a later section. For now, it suffices to say that each product's low entropy name is a short sequence of the tokens in the product's concatenated COMMODITY_DESC and SUB_COMMODITY_DESC that appear most frequently across the entire vocabulary of products. Thus, the LOW_ENTROPY_NAME property for the two milk products above would be *white fluid*. These names are not necessarily easily interpretable, but they decrease the product vocabulary size in a meaningful way.

The result of re-mapping similar products to identical identifiers is demonstrated in the box plot in 2.3.2. When we distinguish transactions by the PRODUCT_ID property, the purchase frequencies are extremely low. This indicates that a very large number of non-overlapping products are purchased. At the level of granularity provided by the data, they should overlap more than this. When we distinguish transactions by the combined COMMODITY_DESC + SUB_COMMODITY_DESC properties, purchase frequencies increase drastically. This indicates that there are many products that indeed have the same descriptions and different PRODUCT_IDs. When you distinguish transactions by the LOW_ENTROPY_NAME property, the purchase frequencies increase further.

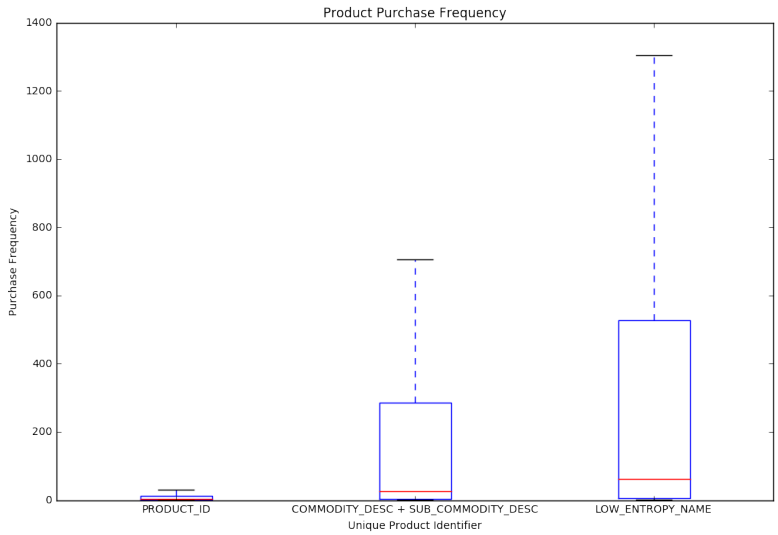

Figure 2.1: Purchase frequency for three types of unique product identifiers.

2.4 Inflated Transaction Quantities for Gasoline Products

To model the quantity of products in a basket, one may increment the number of times a product's identifier is included in the document. For example, if the product *BAG SNACKS* has QUANTITY = 3, the document should contain { BAG SNACKS, BAG SNACKS, BAG SNACKS }.

We found that products with SUB_COMMODITY_DESC = "GASOLINE-REG UNLEADED" had extraordinarily high QUANTITY values. Specifically, there were 24,692 gasoline transactions with a mean quantity of 10,308. The remaining 2,570,770 products had a mean quantity of 1.305. The discrepancy between gasoline product quantity and other products' quantities is demonstrated in figure 2.4. To overcome this issue, we imputed all of the gasoline quantities with QUANTITY = 1.

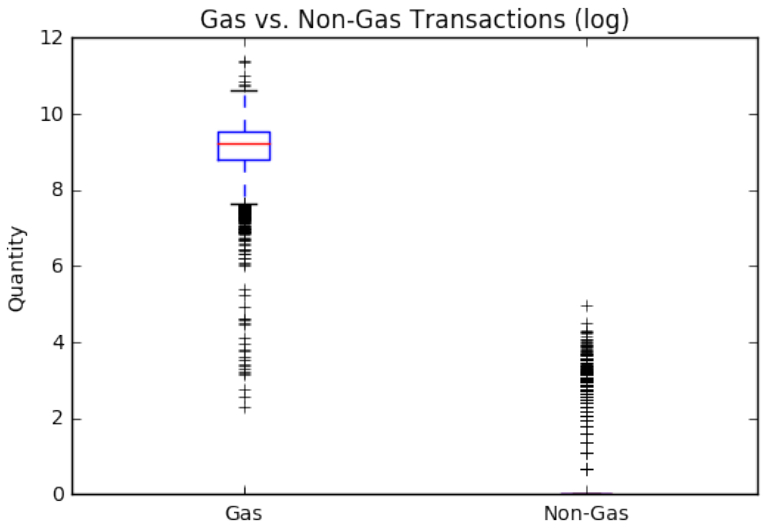

Figure 2.2: Transaction quantities for gasoline and non-gasoline products (log-scale) before imputing.

3 Data Preparation

After running our models several times using COMMODITY_DESC + SUB_COMMODITY_DESC as unique product identifiers, we found products with slightly-differing names were scattered into different topics. For example, "FUEL GASOLINE-REG UNLEADED" and "COUPON/MISC ITEMS GASOLINE-REG UNLEADED" were often grouped into different categories. We concluded that, though concatenating available descriptions is better than using the PRODUCT_ID alone, it was not a sufficient way to group intuitively similar products. To address this, we introduced the concept of low-entropy names.

3.1 Low-Entropy Names

The intuition for low-entropy names is to represent a product by tokens that have the highest likelihood to match the product with other semantically similar products. It helps to look at

an example of the low-entropy names and how they map to the original names, shown in table 3.2, and a comparison of product vocabulary options in table 3.1.

Low-entropy names are created as follows. We tokenize a product by combining its COMMODITY_DESC and SUB_COMMODITY_DESC values into a single string, splitting the string on spaces into individual words, stemming the words using a regular expression to remove punctuation and the NLTK *PorterStemmer*[2], removing any tokens that occur in a manually-defined set of stop words, and replacing tokens that match a manually-defined set of synonyms. We count the token frequencies by looping over every product, tokenizing it, and incrementing a counter corresponding to each token. After this loop, we have a complete frequency count for all tokens in the corpus. We loop over the products again, this time finding the two most frequent tokens for every product. These two tokens become the product's low-entropy name.

The complete process for creating low-entropy names is defined informally in pseudo code:

```
# A product is tokenized by splitting its descriptions into single tokens,
# stemming the tokens, removing stop words, and replacing synonyms.
function tokenize(product):
    tokens = split(product['COMMODITY_DESC'] + product['SUB_COMMODITY_DESC'])
    tokens = stem(tokens)
    tokens = remove_stop_words(tokens)
    tokens = replace_synonyms(tokens)
    return tokens

# Counter-dictionary used to count the frequency.
corpus_tokens_frequency = Counter()

# Tokenize each product to count the frequencies.
for product in corpus:
    tokens = tokenize(product)
    for token in tokens:
        corpus_tokens_frequency[token] += 1

# Tokenize each product, the top two most frequent tokens become the
# LOW_ENTROPY_NAME.
for product in corpus:
    tokens = tokenize(product)
    sorted = sort_by_frequency(tokens)
    top = sorted[0:2]
    product['LOW_ENTROPY_NAME'] = join_string(top)
```

[2] http://www.nltk.org/howto/stem.html

Vocabulary Type	|V|	Highest Frequency for a Token
PRODUCT_ID	92353	1
SUB_COMMODITY_DESC	2383	1005
COMMODITY_DESC+SUB_COMMODITY_DESC	3873	1005
LOW_ENTROPY_NAME	2827	1323

Table 3.1: A comparison of vocabulary characteristics.

Low Entropy Name	Matching	Original Names
everyday spli	1323	GREETING CARDS/WRAP/PARTY SPLY PARTY EVERYDAY, GREETING CARDS/WRAP/PARTY SPLY CARDS EVERYDAY
chocol candi	1016	CANDY - PACKAGED CANDY BAGS-CHOCOCLATE, CANDY - PACKAGED CANDY BOXED CHOCOLATES, NO COMMODITY DESCRIPTION CANDY BAGS-CHOCOCLATE, CANDY - PACKAGED SEASONAL CANDY BAGS-CHOCOLATE, CANDY - PACKAGED SEASONAL CANDY BOX-CHOCOLATE W, CANDY - PACKAGED SEASONAL CANDY BOX-CHOCOLATE
beer ale	772	BEERS/ALES BEERALEMALT LIQUORS
entre premium	665	FRZN MEAT/MEAT DINNERS FRZN SS PREMIUM ENTREES/DNRS/T, FRZN MEAT/MEAT DINNERS FRZN SS PREMIUM ENTREES/DNRS/N, FROZEN MEAT FRZN SS PREMIUM ENTREES/DNRS/T
extract spice	623	SPICES & EXTRACTS HISPANIC SPICES AND SEASONINGS, SPICES & EXTRACTS SPICES, SPICES & EXTRACTS SPICES/SEASONING, SPICES & EXTRACTS SEAFOOD-MISC-SPICES, SPICES & EXTRACTS SPICES & SEASONINGS
sherbt ice	589	ICE CREAM/MILK/SHERBTS PREMIUM PINTS, ICE CREAM/MILK/SHERBTS SUPER PREMIUM PINTS, ICE CREAM/MILK/SHERBTS PREMIUM
spli greet	576	GREETING CARDS/WRAP/PARTY SPLY CARDS SEASONAL, GREETING CARDS/WRAP/PARTY SPLY PARTY SEASONAL
gift everyday	547	GREETING CARDS/WRAP/PARTY SPLY GIFT-WRAP EVERYDAY
chip potato	531	CHIPS&SNACKS POTATO CHIPS, BAG SNACKS POTATO CHIPS
shampoo hair	512	HAIR CARE PRODUCTS SHAMPOO

Table 3.2: Low entropy names mapped to their original names with the number of matching products for each low entropy name.

4 Method 1: LDA using Low-Entropy Naming

4.1 LDA Implementation

We use an LDA model implementation from the popular python library Gensim[3]. The library provides convenient means for parsing the documents into a dictionary and a corpus that can be used to train an LDA. Once the LDA is trained, a document can be passed through it to determine its topic distribution. The complete pipeline for preparing, training, and using

[3]https://radimrehurek.com/gensim/models/ldamodel.html

the LDA is diagrammed in figure 4.1.

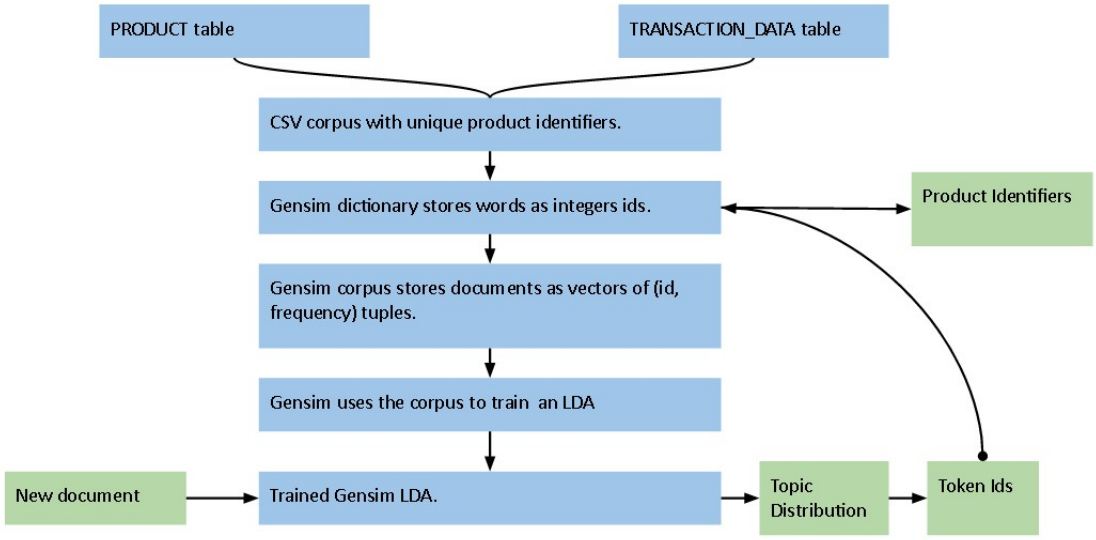

Figure 4.1: Pipeline for preparing documents, fitting the LDA, and evaluating new documents.

4.1.1 Hyperparameters

For the value of T (number of topics), we tried 5 to 75 in intervals of five. The model presented uses 10 topics. We retained the default distribution hyperparameters from the Gensim implementation. They are described in the documentation as follows.

alpha and eta are hyperparameters that affect sparsity of the document-topic (theta) and topic-word (lambda) distributions. Both default to a symmetric 1.0/num_topics prior.

alpha can be set to an explicit array = prior of your choice. It also support special values of 'asymmetric' and 'auto': the former uses a fixed normalized asymmetric 1.0/topic_no prior, the latter learns an asymmetric prior directly from your data.

4.2 Evaluating with Perplexity

We used the common perplexity evaluation to determine which value of T best fits the dataset. Again, T was sampled from 5 to 75 at intervals of 5. We randomly sampled 70% of the corpus for training the LDA, the remainder was used to evaluate perplexity.

Gensim offers a `bound()` method for evaluating perplexity. However, this method returns a less familiar representation of perplexity. Specifically, the `bound()` method takes the testing corpus and returns the variational bound against the trained LDA. According to the library's author, this is a lower bound on perplexity,[4] based on a technique from *Online Learning for Latent Dirichlet Allocation*[5]. Further, a lower bound implies deterioration in the model, **and a higher bound indicates improvement.**[6] The perplexity evaluations over all values of T are presented in figure 4.2.

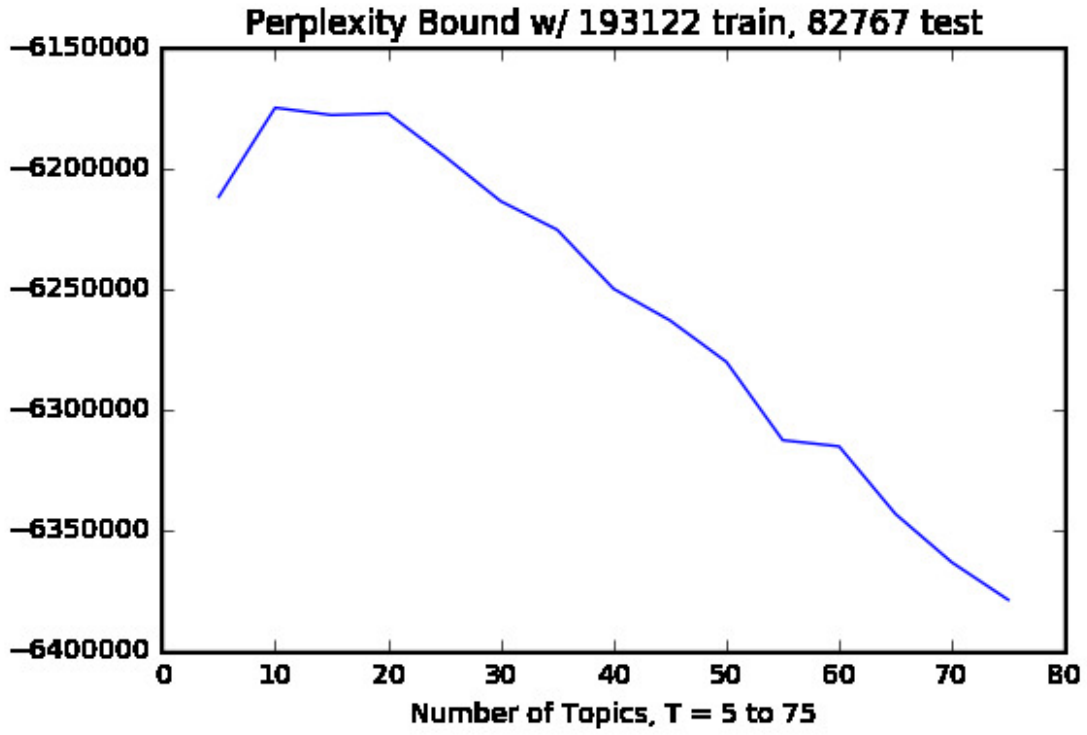

Figure 4.2: LDA perplexity lower bound; note that higher is better.

4.3 Manual Topic Observations

To address the question of grouping products into distinguishable categories, the simplest way to evaluate our model is to manually inspect the topic-word associations. To this end, we include topics 0, 5, and 9 in tables 4.1, 4.2, 4.3, respectively.

We subjectively analyze the selected topics as follows. Topic 0 includes several junk-food

[4] https://groups.google.com/forum/!topic/gensim/LM619SB57zM
[5] https://www.cs.princeton.edu/ blei/papers/HoffmanBleiBach2010b.pdf
[6] https://groups.google.com/forum/!topic/gensim/iK692kdShi4

items, alcoholic beveraages, and gasoline at a high probability. One may call this the "convenience store" topic. Topic 5 includes generic grocery items, but does not stand out as particularly distinguishable. Topic 9 includes several baby and cleaning items. One may call this the "parent category."

Topic 0 Word	Low-Entropy Word	p(w|z)
SOFT DRINKS 12/18&15PK CAN CAR	15pk car	0.29
GASOLINE-REG UNLEADED	unlead fuel	0.14
BEERALEMALT LIQUORS	beer ale	0.09
SW GDS:DONUTS	donut breakfast	0.06
CARDS SEASONAL	spli greet	0.02
PREP FD: SIDE DISHES (HOT)	prep side	0.02
COLD AND FLU - DXM	dxm flu	0.02
TORTILLA/NACHO CHIPS	nacho tortilla	0.01
ROLLS: DINNER	roll dinner	0.01
BREAD:ITALIAN/FRENCH	italian french	0.01

Table 4.1: LDA top ten words in topic 0 (T = 10)

Topic 5 Word	Low-Entropy Word	p(w|z)
FLUID MILK WHITE ONLY	white fluid	0.09
BANANAS	banana tropic	0.03
ORANGES NAVELS ALL	navel orang	0.02
CIGARETTES	cigarett	0.02
DAIRY CASE 100% PURE JUICE - O	pure 100	0.02
EGGS - X-LARGE	egg	0.01
ALL FAMILY CEREAL	famili cereal	0.01
MAINSTREAM WHEAT/MULTIGRAIN BR	wheat multigrain	0.01
CREAM CHEESE	cream chees	0.01
MAINSTREAM WHITE BREAD	white bun	0.01

Table 4.2: LDA top ten words in topic 5 (T = 10)

Topic 9 Word	Low-Entropy Word	p(w|z)
BABY FOOD - BEGINNER	beginn babi	0.06
PAPER TOWELS & HOLDERS	holder towel	0.04
BABY FOOD JUNIOR ALL BRANDS	junior babi	0.03
CANDY BARS (MULTI PACK)	multi pack	0.03
TOILET TISSUE	toilet tissu	0.03
FACIAL TISSUE & PAPER HANDKE	handk tiss	0.03
CHEWING GUM	gum chew	0.03
LIQUID LAUNDRY DETERGENTS	deterg laundri	0.02
BABY DIAPERS	diaper dispos	0.02
BAR SOAP	soap liquid	0.02

Table 4.3: LDA top ten words in topic 9 (T = 10)

4.4 Demographic Distribution over Topics and Entropy Comparison

Next, we address the task of identifying demographic attributes that strongly distinguish shoppers and their latent missions.

To this end, we first compute the population distribution over topics. This is computed by collapsing all documents in the corpus into a single document and running it through the LDA to get a topic distribution, shown in figure 4.3.

Next we use the household demographic information from the HH_DEMOGRAPHIC table to compute distributions for every demographic pair (e.g. (AGE_DESC, 19-24)) over topics. The demographic distribution allows us to say, "given a household falls into demographic X, the probability of purchasing a product from topic Y is Z." For each column in the HH_DEMOGRAPHIC table, we find all of its unique values. For each of these values, we find all household_keys matching this value. We use the household_keys to look up all documents for the matching households. Finally, we collapse the matching documents into a single large document and run it through the LDA to get the demographic topic distribution.

Finally, we compute the KL-Divergence for each demographic distribution relative the population distribution using the python `scipy.stats.entropy()` method[7]. This measure allows us to determine which demographics are most distinct from the population. The results are presented in table 4.4

Interestingly, the top five most distinct demographics all have high earnings ($100K +) and each prefers category five. The first, second, and sixth-ranked demographic distributions are presented in figure 4.3, respectively. A simple extension of this technique would compare all possible pairs of demographics to determine which demographics have the most polar shopping preferences.

	Demographic	Category	Entropy relative Population	Top Topic
0	INCOME_DESC	175-199K	0.102886	5
1	INCOME_DESC	200-249K	0.093301	5
2	INCOME_DESC	250K+	0.060177	5
3	INCOME_DESC	150-174K	0.052650	5
4	INCOME_DESC	100-124K	0.046601	5
5	HOMEOWNER_DESC	Probable Owner	0.040425	8
6	INCOME_DESC	125-149K	0.033437	5
7	AGE_DESC	19-24	0.027506	1
8	AGE_DESC	65+	0.024285	5
9	HH_COMP_DESC	1 Adult Kids	0.020416	8
10	INCOME_DESC	Under 15K	0.019510	8

Table 4.4: Top 10 entropy values for demographic distributions over topics relative the population distribution

[7] https://docs.scipy.org/doc/scipy-0.18.0/reference/generated/scipy.stats.entropy.html

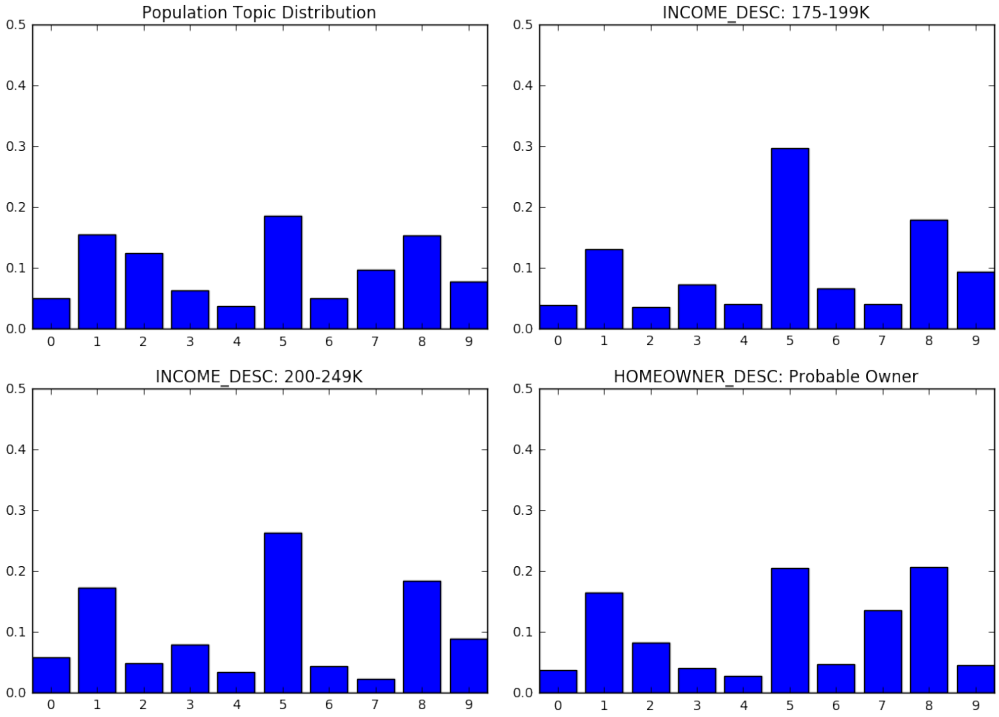

Figure 4.3: LDA distribution of all documents over topics and the 1st, 2nd, and 5th most different demographic distributions.

5 Method 2: AT Model

5.1 AT Implementation

As can be expected, finding an implementation of the Author-Topic (AT) model was more difficult than finding an implementation of LDA. We were successful in finding one implementation of AT that is part of a python package called python-topic-model[8]. This package has an entirely different input than the LDA package, and thus more data processing was necessary to implement the model.

A simple pipeline of the implementation process is given by 5.1. The product data file is used to create a list of unique words that is used as the vocabulary. The copus is a list of documents. Each document in the corpus is a list of words. Each word, is in fact the index of the word in the vocabulary. Some of the documents in the transaction data file have authors that do not have any demographic information in the demographic data file. We removed all of these documents from the corpus prior to training the models. The demographic data is also used to create a unique list of authors, essentially an author vocabulary. This author vocab-

[8]https://github.com/arongdari/python-topic-model

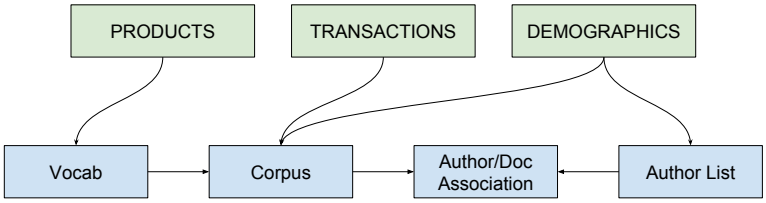

Figure 5.1: Caption

ulary coupled with the author-document association information gleaned while creating the corpus is used to create essentially a corpus of authors. The corpus of authors is a list collaborating author groups. Each collaborating author group is a list of authors. Each author in a group is actually an index into the author vocabulary. The key point is that the group of authors at index i in the author corpus are the authors for the document at index i in the document corpus.

5.2 Experiments

For the number of topics, T, values of 10, 15, 20, 25, 30, 40, 50 were all tested. Gibbs sampling iterations of 10 and 50 were tested. Two author paradigms were tested, demographic information as authors, and household IDs as authors. In all, 28 methods were trained.

5.3 Evaluating the Models

The package used to implement AT does not have a perplexity implementation. Because of this evaluation of the AT models relies on visual inspection and entropy evaluation.

5.3.1 Manual Topic-Word Observations

A number of different models are manually evaluated. First, the model using 10 topics, 50 Gibbs Sampling iterations, and demographic authors will be evaluated. Second, the model using 20 topics, 50 Gibbs Sampling iterations, and demographic authors will be evaluated. Finally, the model using 10 topics, 50 Gibbs Sampling iterations, and household ID authors will be evaluated.

In 5.1 we can subjectively evaluate this topic as pertaining to "mid-weekday easy meal shopping" demonstrated by the existence of: white buns/bread, lean beef, ramen, pasta, single serving means.

In 5.2 we can subjectively evaluate this topic as pertaining to "fresh produce" demonstrated by the existence of: strawberries, lemons, bananas, limes, cauliflower.

In 5.3 it is harder to determine high level ideas in the topics of the model with 20 topics, but we can subjectively evaluate this topic as pertaining to "dairy/refrigerated items" demonstrated by the existence of: shredded cheese, yogurt, deli ham, milk, eggs, cheese singles, whipped topping.

5.4 seems to have picked up on similar features as the model with the demographic authors did as again we can subjectively evaluate this topic as pertaining to "fresh produce" demonstrated by the existence of: bell peppers, bananas, cucumbers, cauliflower, strawberries.

Low-Entropy Word	p(w|z)
incl liter: 2 LTR	0.0435400775017
white bun: 20 OZ	0.0194244916306
ramen cup: 3 OZ	0.0168338102399
srv sngl: 20 OZ	0.0165095036903
pasta stabl: 15 OZ	0.0142959229387
15pk car: 12 OZ	0.013945219344s
lean beef:	0.0110867964999
cigarett: CTN	0.010113876851
economi entre: 8 OZ	0.0100610827615
white fluid:	0.00906176606788

Table 5.1: Top 10 words of topic 2 from model trained with 10 topics, 50 Gibbs Sampling iterations, and demographic authors

Low-Entropy Word	p(w|z)
not yogurt: 6 OZ	0.0556582883036
strawberri berri: 16 OZ	0.0160711494707
lemon citru:	0.0138042548419
banana tropic: 40 LB	0.0128796004538
bun roll: 24 OZ	0.0128080143077
lime citru: 36CT	0.011400153433
italian french:	0.0112092570432
crown cauliflow:	0.0103740853378
boneless breast:	0.00962243080304
homestyl chunki: 18.8 OZ	0.00896622446313

Table 5.2: Top 10 words of topic 9 from model trained with 10 topics, 50 Gibbs Sampling iterations, and demographic authors

Low-Entropy Word	p(w|z)
incl liter: 2 LTR	0.0483855072865
shred chees: 8 OZ	0.0242816741747
not yogurt: 6 OZ	0.0201244788694
15pk car: 12 OZ	0.0152563142958
ham deli:	0.0130049116127
white fluid:	0.0117903391126
everyday spli:	0.0111287427101
egg: 1 DZ	0.00941056697825
singl chees: 12 OZ	0.00886746545381
whip top: 8 OZ	0.00868972313672

Table 5.3: Top 10 words of topic 17 from model trained with 20 topics, 50 Gibbs Sampling iterations, and demographic authors

Low-Entropy Word	p(w|z)
boneless breast:	0.0128208118678
shred chees: 8 OZ	0.0124654921888
unlead fuel:	0.0117255912101
bell pepper: 48-54 CT	0.0112574052801
banana tropic: 40 LB	0.0110734750933
white fluid: 1 GA	0.0110024111575
cucumb veget: 36 CT	0.0101120218442
italian french:	0.00981104517492
crown cauliflow:	0.00952260920018
strawberri berri: 16 OZ	0.00943064410679

Table 5.4: Top 10 words of topic 4 from model trained with 10 topics, 50 Gibbs Sampling iterations, and household key authors

5.3.2 Manual Author-Topic Observations

In addition to the topic-word distributions we can also examine the author-topic distributions. This visual evaluation is only done on the models that used demographic information as authors. This is because the sheer number of authors in the household key data makes this kind of evaluation difficult. Results are shown below for the model trained with 10 topics, 50 iterations of Gibbs sampling, and demographic authors. Other topic sizes were evaluated showing similar results. The main interesting discovery is the clustering of author-topic distributions into three main clusters: relatively uniform low topic probabilities represented in 5.2, relatively uniform low topic probabilities with a few high spikes represented in 5.3, one very high spike, open at 100% represented in 5.4. Of the three clusters, I think the authors in the mid cluster are the most interesting.

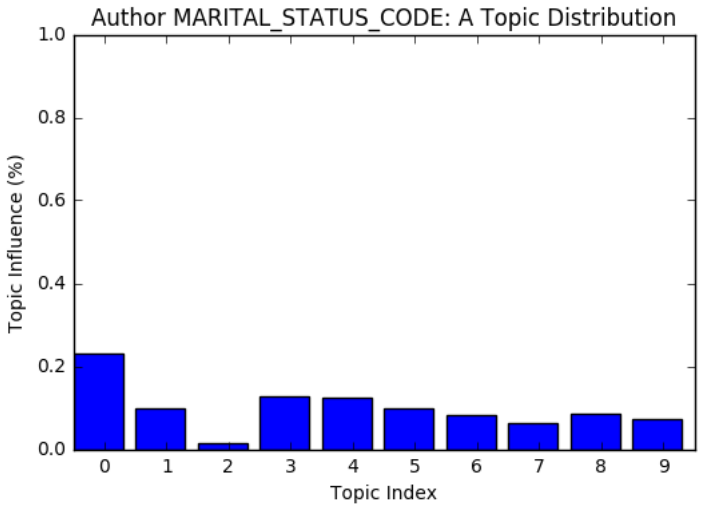

Figure 5.2: Representation of relatively uniform low topic probabilities

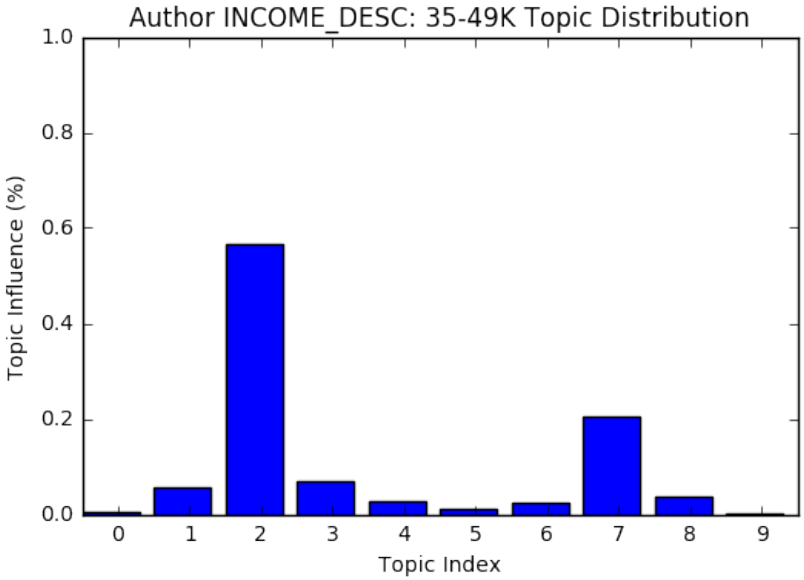

Figure 5.3: Representation of relatively uniform low topic probabilities with a few spikes

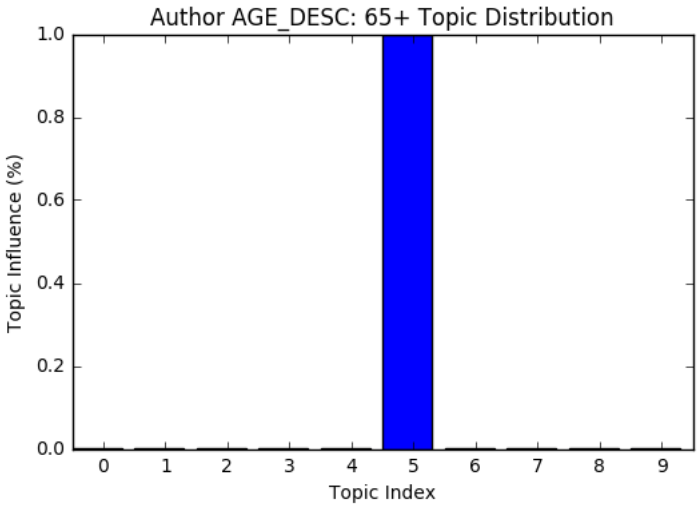

Figure 5.4: Representation of one very high spike

5.3.3 Per Author Entropy

One more way to glean interesting information from the models is to calculate an aggregate author-topic distribution. This will allow us to see how on average the authors rely on the individual topics. With the aggregate we can also calculate the entropy, the difference between two distributions, for that each author has from the aggregate.

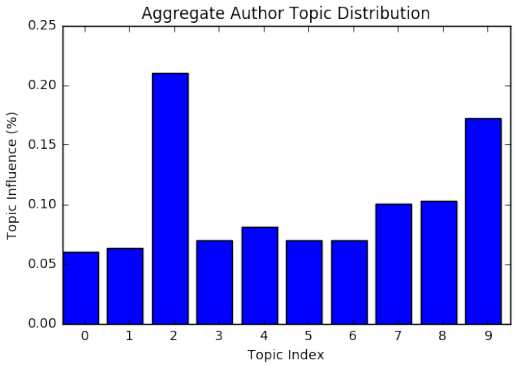

Figure 5.5: Aggregate author-topic distribution

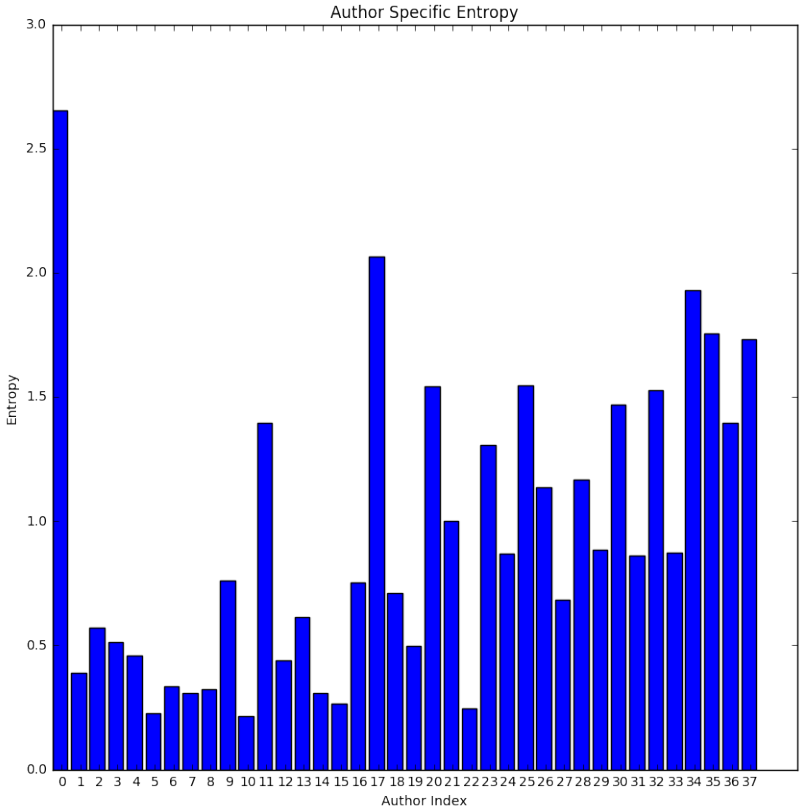

Figure 5.6: Per author entropy with respect to the aggregate author-topic distribution

We can see in 5.5 that topics 2 and 9 are heavily weighted. This is evidenced by the fact that there are a small number of authors who have the "single spike" author-topic distribution at 2 and another small number of authors with their spike at 9.

We can see in 5.6 that topic 1 has a very large entropy. This makes sense as viewing the topic distribution for the first author, 5.4 the distribution has a single peak at 5. 5 is also one of the smallest represented topics in the aggregate distribution. Taking the two pieces of information the first author would naturally have a very high entropy.

6 Future Work

Future work might include a more sophisticated model and more extensive incorporation of available data. In its current state, our work fails to capture any temporal aspect of the model, though the purchase day is available in the TRANSACTION_DATA table. A hidden topic Markov model or long-short-term-memory recurrent neural network implementation might capture the temporal aspect of shopping. This might help answer questions like "How

does a demographic's topical preference change over time?" or "Do the coupons and advertising campaigns create a change in topical interest?"