Open Storefront Directory: Data Collection Methodology

Jeremy Ben-Meir (jsb459@cornell.edu)

July 2022

Contents

1 Overview

We use publicly available data from multiple New York City and State departments in order to understand what businesses have existed in New York City from 2010 to 2021. After data cleansing, the dataset contains information on over 800,000 business observations from 2010 to 2021. On average, we record approximately 67,500 observations each year. For each observation we include: business name, address, building identifier, start and end dates (inferred). We also record lot information, zoning information, and much more. Supplemental data from the Census Bureau, or the Department of Transportation, for example, may be appended to enrich the data.

We took gaps in the business history as implied vacancy (if four businesses were once present in a certain building, but three only exist at a point in time, we can infer there is one vacant unit at that time).

The accompanying program can be run on top of the previously collected data. It will take into account just the most recent information and output a new dataset with updated relevance.

Read more about the data here.

2 Data Collection Methodology: A Closer Look

2.1 Sources

We began by analyzing public sources of rich data that might help pinpoint the historical lifespan of New York City Businesses. We found eight departments, each with valuable data. From these data sources, we primarily aim to collect the following information: Dates (Expiration dates, Start dates, Inspection dates), Address(es), Name(s), Phone number(s), License Identifier(s), Inspection Result, Business Type.

1. The Department of Consumer Affairs (DCA)

   The DCA offers five sources of data: applications, charges, inspections, licenses, revocations. This data may be found on the NYC Open Data website.

2. The Department of Agriculture (DOA)

   The DOA offers four sources of data: an overarching dataset, two inspection datasets, and one regarding deficiencies. This data is accessible through a FOIL request.

3. The Department of Education (DOE)

   The DOE has pharmacy data, which we collected using a webscraper on their website.

4. The Department of Finance (DOF)

   The DOF collects certificate data on every legal business entity in New York City. This data is accessible through a FOIL request.

5. The Department of Heath (DOH)

   DOH inspection data may be found on the NYC Open Data website. DOH license data, however, is accessible through a FOIL request.

6. The Department of State (DOS)

   BetaNYC, a government tech nonprofit, wrote of using barbershop and appearance enhancement licenses to track businesses over time. The DOE lists data on every barber shop and appearance enhancement service, which we collected using a webscraper on their website.

7. The Department of Transportation (DOT)

   DOT inspection and application data may be found on the NYC Open Data website.

8. The Liquor Authority

   The liquor authority lists every business with a liquor license, which we collected using a webscraper on their website.

2.2 Data cleansing

Once we collect data from each source, we need to translate the data to a Pandas dataframe. This transformation allows the data to be easily manipulated, utilizing Python to organize the data. Using the aforementioned data sources, we are able to access fine-grained details on businesses across all of New York City.

We needed to create a method to ingest addresses and ensure that we could understand multiple representations of the same address. New York City's borough-block-lot (BBL) system assigns each building to a unique borough, block, and tax lot. The Geosupport library for Python takes address strings and returns an abundance of information about

each building, including zip code, census tract, year built, building type, longitude, latitude, and–of course–BBL.

The source data, however, is often unclean, unclear, and fraught with errors. Often zip codes, addresses, or business names miss a letter or are entirely incorrect. So, in order to add a BBL to each record, we needed to clean each address to make it interpretable by the library. We eliminated non-numeric characters from the zip code and building number entries, and we ensured that the entries for borough had one of the five borough titles.

Once the addresses were cleaned, we were able to generate BBLs for over 1.9 million of the nearly 2.1 NYC business records. In addition to adding BBL codes to each business record, we standardized the phone numbers to a ten-digit string and renamed the columns such that the columns that represented the address in one dataframe lined up with those in the others.

2.3 Record linkage

Using the information under each column header, we need to determine which rows pertain to the same business. For example, both the DOF and the DOA may have collected information on a Dunkin franchise on 95th Street, and Columbus Avenue. The DOA may have entered the name as "Dunkin," at an address of "95th Street and Columbus Ave," the DOF may have written an entry using the name "Dunkin Donuts" and the address "730 Columbus Ave." Although the name and the address are represented differently, they reference the same business. We need to write a script to determine the similarity between two address entries, as well as two separate names. We may then assign a unique identification string to each business, to keep track of these matchings.

To start, within a tax lot, for each possible pair of records, we compared the name, phone number, agency-issued ID number, and business type. For the name, we calculated the cosine distance between the name values in each record (decimal value between 0 and 1). For the phone number and the agency-issued ID, we looked for an exact match (categorical value of either 0 or 1, although the phone number comparison feature may take the value of .5 if a phone number is not listed). For each business type, we created sets of like business types. If two records had business type values in the same set, we returned a value of 1 (otherwise 0). Within each BBL-block, there are $\binom{\#\text{ records per BBL}}{2}$ comparisons, and for each comparison, we extract these four features (the comparisons of business name, phone number, agency-issued ID number, and business type, which we will call the comparison features).

We tried using both a deterministic algorithm and an unsupervised clustering algorithm in order to determine if two business records pertained to the same business. We used a Gaussian Mixture Model as our unsupervised probabilistic model: this allowed us to fit multi-dimensional Gaussian distributions to our clusters of comparison features. We fit these distributions across the comparison features of the entire dataset. The Gaussian Mixture Model begins by fitting K (2 clusters in our case: match and no match) distributions to the data with random parameters. We find the optimal values for these parameters, and the solutions will correspond to the Maximum Likelihood Estimates (MLE) for this setting. These distribution parameters are then adjusted in an iterative process called expectation maximization. This iteration occurs until the convergence in the MLE value.

The primary issue with these models is that they expect to fit features in the space of $\mathbb{R}_n$ to the Gaussian distributions. While our name comparison feature (representing cosine similarity) is in $\mathbb{R}$-space, our other comparison features are categorical. We are thus unable to fit Gaussian distributions to these categorical features using the standard methodology, but found a paper and application of Gaussian Mixture Models on mixed datasets.[1] In short, this algorithm translates our $\mathbb{Z}$-space categorical features to the $\mathbb{R}$-space in a fashion similar to the hidden layers of a neural network. On our new set of $\mathbb{R}_n$ features we fit two Gaussian distributions, one that represents matches, and one that represents mismatches. We used the matches cluster to determine if two rows pertained to the same business.

[1] https://arxiv.org/pdf/2010.06661.pdf

2.4 Business type determination

After matching like-businesses, we must understand the type for each business. A small coffee shop may have a different vacancy and turnover performance than a large supermarket. We must use the name and business type labels (if applicable) to sort each business into particular, interpretable types.

While licenses usually contain some information about the business' industry, the information is not standardized. The North American Industry Classification Scheme (NAICS) is commonly used for industry categorization. Our challenge was to translate the unorganized industry/business type descriptions found in our data into NAICS codes. We used an adjusted version of the Bidirectional Encoder Representations from Transformers or BERT language model, called Sentence-BERT or SBERT to accomplish this. The model looks at words in a bidirectional contest–this suggests that it looks at words both to the right and left to understand their context. SBERT uses this analysis on common Google searches to create semantically meaningful sentence embeddings. This allows us to input two distinct sentences to the model, and quickly derive a value similar to the cosine distance. Rather than using characters to determine this distance, it is based on the semantic embeddings.[2]

We used these embeddings to compare the semantic distance between a business' type, and the list of business types associated with NAICS codes. We assigned the NAICS definition with the highest semantic distance and its code to the business type of each business. An NAICS code has six digits, and, as the place of each digit decreases, the specificity increases. This is to say that we may use the first two digits of the NAICS codes to classify businesses into 100 broad business types, rather than classifying them based on the error-prone string definitions of business type.

2.5 Business start/end date determination

We then use data from each data source to determine how long each business has existed in a particular location. This allows us to not only understand business trends, but patterns pertaining to building vacancy: if we know where each business is over time, we can identify vacancies. We may use the dates associated with each record to understand when a business begins and ends.

We took the earliest date that shows up for each business to represent the start date of the business. To understand when the business closed, there may be dates associated with an "out of business" date. Otherwise, we determine how recent the latest recorded dates are. If we determine that the latest recorded dates are recent enough, we decide the business still exists.

2.6 Observation generation

We create a new dataset of observations of each business each year, with a variable indicating whether or not the business survives in the following year.

From the two dates we create for each business, "start date" and "end date," we create "business observations." We generate these observations at the arbitrary date: January 1 of each year. We record each business that exists at that time, and if they survive into the following year (this is represented by a 1 or a 0).

At each January 1 date, we add certain useful features: how long the business has survived, the economic trends of the time period, the zoning conditions of the buildings. At this stage, we may also add subway station data, crime data, Yelp data, or any locational data to enrich the final dataset.

3 How accurate is the data?

We have not yet performed a comprehensive analysis of the accuracy of our data against some ground truth. Sources of this ground truth may be Google's historical street views, or a continuously updating map of businesses maintained by LiveXYZ.[3]

[2] https://arxiv.org/abs/1908.10084
[3] https://share.livexyz.com/

There are nearly 50,000 retail businesses in New York City during a given year. With a high turnover estimate of 20%, we may assume that we capture an additional 100,000 businesses over the course of ten years. Given the high estimate of 150,000 businesses captured, if we identify the linkage success rate of just 385 businesses, we can be 95% certain that our found linkage success rate is within 5% of the true linkage success rate of the data.

4 Application Examples

4.1 Vacancy prediction

4.1.1 Methodology

We used the follow colums as feature values in each model: 'zonedist1', 'bldgclass', 'histdist', 'landmark', 'lotarea', 'bldgarea', 'comarea', 'resarea', 'officearea', 'retailarea', 'garagearea', 'strgearea', 'factryarea', 'otherarea', 'numfloors', 'unitsres', 'unitstotal', 'lotfront', 'lotdepth', 'bldgfront', 'bldgdepth', 'bsmtcode', 'assessland', 'assesstot', 'yearbuilt', 'yearalter1', 'builtfar', 'residfar', 'commfar', 'facilfar',"Months Active", 'GCP (NYC)', 'GDP (USA)', ' Payroll-Jobs Growth, SAAR - NYC', 'Payroll-Jobs Growth, SAAR - USA', 'PIT Withheld, Growth, NSA - NYC', 'PIT Withheld, Growth, NSA - USA', 'Inflation Rate, NSA - NYC', 'Inflation Rate, NSA - USA', 'Unemployment Rate, SA - NYC', 'Unemployment Rate, SA - USA'.

Although we cleansed the data and tried to make sense of address and business type values, given that our data is generated from a wide range of unclean data sources, many of the values of the features were NaN. We looked to pursue one of three options: we could eliminate rows with NaN values in any feature column, we could use a model robust to NaN values, or we could impute the NaN values (using the mode, mean, or a mix of both). Upon elimination of each row with any NaN value, we saw a decrease in data points from 643,597 to just over 60,000. Nearly 70% of these losses were due to a NaN value in just one feature column, and so we lost a lot of data pertaining to changes in the other columns. Also, eliminating all rows with any NaN values would disallow us from making predictions on 90% of businesses. Although a model robust to NaN score may be stronger, as it would not ignore the differences between rows with NaN and rows without, Scikit Learn does not yet allow the use of models robust to NaN values (although XGBoost does).[4] So, we decided to impute the missing values in all columns, using the mode of the feature column. In imputing rather than eliminating NaN, we were seeing higher F1 scores.

We transformed both the target column (survival value) and each feature column into NumPy arrays. Of the feature columns, some were categorical ('zonedist1', 'bldgclass', 'histdist', 'landmark'), so we needed to binarize the column data. The other columns were numerical, so we did not apply any mappings to them during the transformation to a NumPy array.[5]

We tested out seven different classifiers in order to see which model would best fit the data and have the closest, most accurate results. The seven we worked with were MLP Classifier, SVM SVC, Stochastic Gradient Descent, Decision Tree, Random Forest, Cox's proportional hazards model, Random Survival Forests.

MLP Classification.[6] This is a multi-layer Perceptron classifier that optimizes the log-loss function using LBFGS or stochastic gradient descent. We may specify the hidden layer size, the solver, and other parameters. The MLP classification is very flexible, and well suited to generally map input to output classification variables.

C-Support Vector Classification (SVM SVC).[7] This classifier utilizes the flexibility of kernel functions to separate a vector space into the target classification. We may specify the kernel type, along with other parameters. This support vector classifier works best on linearly separable data, and the runtime scales quadratically given the number of samples.

Stochastic Gradient Descent Classification.[8] This classifier determines a loss func-

[4] https://stackoverflow.com/questions/30317119/classifiers-in-scikit-learn-that-handle-n...
[5] https://dunyaoguz.github.io/my-blog/dataframemapper.html
[6] https://scikit-learn.org/stable/modules/generated/sklearn...
[7] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
[8] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html

tion of which is estimated the gradient at a given point. We may specify the loss function type, penalty, and other parameters. The gradient of the loss is estimated each sample at a time and the model is updated along the way with a decreasing strength schedule until convergence. Although computationally fast, the classifier steps toward the minima are noisy, and may lean the gradient descent in different directions.[9]

Decision Tree Classification.[10] Decision trees split data points by features, often beginning with those most deterministic of effects in the target variable (may make this selection random). It may do this to a max depth, which we may specify in order to prevent overfitting. Although decision trees often require relatively minimal preprocessing, small changes in data may yield large changes in predictions, as changes propagate through the tree.[11]

Random Forest Classification.[12] This classifier fits a number of decision tree classifiers on different sub-samples of the data. This allows the random forest classifier to improve predictive accuracy and control for overfitting. We may specify the number of trees, the criterion to determine the quality of a split, and the max depth for each tree, along with other parameters.

Cox's proportional hazards model. In order to predict survival after multiple years, we need to understand the conditions of a business in the future. We may use an above model on 2021 economic and location condition data to predict the survival of businesses in 2022. In order to then make predictions about 2023 business survival, we need to make assumptions about the economic and location data of 2022, and so on. There are multiple sources of data on predictions of economic trends, and there are building permits that may assist in the prediction of location data. Retrieving and organizing the data may be too complicated, so we can use the confidence of our prediction (if the prediction indicates that a business will survive) as an analog for longer-term survival predictions. This is to say that a business that is predicted to survive into the following year with low confidence, will likely not survive as long as a business with high confidence. We may test the correlation between confidence and business longevity by taking our 2015 predictions, measuring their confidence values, and retrieving the correlation between those confidence values and the survival time after 2015. Given a positive correlation, we may assume that the confidence values are an acceptable, although imprecise, measure of longevity.

Although this may work, Cox's proportional hazards model is built to predict survival times and thus may be used to circumvent the use of a confidence measure. The model identifies entities at a point in time and, along with important features, two variables are collected for each observation: "status" and "survival time." Status measures whether or not a record was censored–that is to say that the collection of data stopped at a point, and so the survival time listed serves as a minimum value. This may occur for businesses that are open during the time of observation, so we do not have a sense of their full survival time, only how long they have survived. When the "status" variable is false, however, the record was not censored, so the survival time is as recorded. For the purpose of this project, this means that the business has closed.[13]

The survival function, $S(t) = P(T > t)$, returns the probability of survival beyond time t, where T denotes a continuous non-negative random variable corresponding to survival time. We may estimate the value of the function: $S(t)$=number of patients surviving beyond t / total number of patients. Given censored values, however, we may not understand exactly how many patients survive beyond a certain time. The Kaplan-Meier Estimator can, however, make these estimations given censored information.

We must consider that there may be differences between the survival of different entities, given different features (for buildings, imagine traffic, rent, zoning, etc). Although we can divide data by features, this grows unwieldy as feature counts increase. Cox's proportional hazards model is a multivariate survival model that estimates the impact of each feature on survival.

Traditionally, we make one observation of each data point and do not capture time-

[9] https://www.asquero.com/article/advantages-and-disadvantages-of-stochastic-gradient-d...
[10] https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html
[11] https://dhirajkumarblog.medium.com/top-5-advantages-and-disadvantages-of-decision-tree...
[12] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClas...
[13] https://scikit-survival.readthedocs.io/en/stable/user_guide/00-introduction.html

dependent covariates. Between changes in building data and economic trends, however, we think our predictions are impacted by changes in these trends, and so we need to adapt Cox's model to take into account these time-dependent values. In order to account for these values, we set twelve yearly observation points (from 2010 to 2021), where each business observed has a start date before the observation date. If it exists after the end of the observation period, we mark that record as censored. To these observations, we apply the model, which fits a line to the features and returns a set of coefficients, which serve to estimate the effect of each covariate.

Although we are able to use time-dependent variables as features in our model, we still struggle to use them in predictions for many years into the future, as changes in the features are often unpredictable. As such, a time-dependent survival model with unpredictable time-dependent variables relies on the "martingale theory, a mathematical construct which arose out of the study of games of chance. A key underlying condition for a martingale-like game is that present actions depend only on the past." This is to say that our understanding of the future of a time-dependent feature depends only on the past/present values of that feature.[14]

Once the feature set is properly formatted. Cox's model fits a line to the data, and assigns coefficient values, which allows us to understand the impact of the marginal increase of each covariate. The default method, Breslow's method, assumes that each event at a given time is distinct. When this number of ties is low, Breslow's method will often return similar coefficient values to Efron's method, which works better with a high number of ties.[15]

Random Survival Forest. Although not a linear model, we can train a random survival forest model on the same dataset as the Cox proportional hazards model. Unlike the Cox proportional hazards model, however, random survival forest models allow risk estimates to vary over time. This flexibility allows for the understanding that the risk factors that affect business survival are not constant year to year. We must still use our understanding of the martingale theory to use past data to make assumptions about unpredictable features. Also unlike Cox's model, rather than a linear model, random survival forests fit individual de-correlated trees to bootstrap samples of the original data. Rather than selecting features by those that are most decisive of the independent variable, this model makes those selections randomly.[16]

4.1.2 Results

We explored some of the results of the applications as discussed in parts 4.1 and 4.2. With regard to predicting business vacancy (4.1), we identified some logical relationships between success and business features that we may use to create our prediction.

bldgclass=Z1	1.032099e+00
bldgclass=Z2	1.473641e+00
bldgclass=Z3	1.447877e+00
bldgclass=Z4	8.477341e-01
bldgclass=Z8	1.412394e+00
bldgclass=Z9	6.536553e-01
lotarea	-2.183483e-10
garagearea	-2.824758e-08
strgearea	8.090647e-08
numfloors	3.180696e-03
unitstotal	2.085759e-05
bldgdepth	-1.997813e-05
yearbuilt	-5.252067e-05
builtfar	-5.052990e-05
GCP (NYC)	-7.064690e-03
Payroll-Jobs Growth, SAAR - NYC	2.570808e+00
Inflation Rate, NSA - NYC	4.324422e+00
Unemployment Rate, SA - NYC	-2.679454e+00

[14] https://cran.r-project.org/web/packages/survival/vignettes/timedep.pdf
[15] https://scikit-survival.readthedocs.io/en/stable/api/generated/sksurv.linear_model...l
[16] https://scikit-survival.readthedocs.io/en/stable/user_guide/random-survival-forest.html

4.2 Understanding business success

4.2.1 Methodology

Just as we may use the model above to predict business success, we may use it to understand what exactly makes a business successful.

Brand leadership. We were able to match each business record with one of a selected set of brands. We then computed brand status and proximity for every record and used them as features in the above model. We figured that if there were a positive relationship between a brand-name retailer locating in a particular area and the longevity of surrounding stores, we might conclude brands lead site location for other businesses.

Features of business stability. Beyond brand status and proximity, it would be interesting to explore precisely how different features affect longevity and adaptability. That is to say, when environments change, which businesses survive, and why?

4.2.2 Results

With regard to brand status and proximity (4.2), our preliminary results show that there is a positive effect to being a brand, and a weaker negative effect to being close to a brand.

5 Concluding Note

5.1 Concerns

We have begun to explore some of the applications of the data, and determine results (i.e. the impact of brand status and proximity). Albeit promising, these results seem to suffer from multicollinearity issues. We have found that the outputs change by large margins given slight changes to the inputs. We must, instead, begin by examining where the model's assumptions fail, and adjust accordingly, before delivering impactful results.

5.2 Next Steps

1. We will begin by exploring our question from section 3, by interrogating the accuracy of the data. The data analysis value hinges on the validity of the data itself. Thus, we will begin by using Google Maps to establish a ground-truth of storefronts, and compare that truth to our dataset. We have spoken to a number of researchers who would want to use and cite the data, assuming it is generally "correct."

2. Consequently, we will reevaluate the storefront longevity and success models. We will begin by narrowing the model to key features, and understanding how imputation affects the results.

5.3 Contact and Onboarding

Feel free to reach out to Jeremy Ben-Meir at jsb459@cornell.edu with any questions or comments.