Data Wrangling on Building Permit Data

A Part of the Capstone Project Submissions

Introduction

A building permit is an official approval document issued by a governmental agency that allows you or your contractor to proceed with a construction or remodeling project on one's property. For more details click here. Each city or county has its own office related to buildings, that can do multiple functions like issuing permits, inspecting buildings to enforce safety measures, modifying rules to accommodate needs of the growing population etc. For the city of San Francisco, permit issuing is taken care by Permit Services wing of Department of Building Inspection (henceforth called DBI). The delays in permit issuance pose serious problems to construction industries and later on real estate agencies. Read this. Trulia study and Vancouver city article.

This Document

This document particularly describes the data wrangling steps that I undertook to clean the San Francisco building permit data set. It explains what are tricky parts of data wrangling phase in general. What kind of cleaning steps were performed on this particular set, how the missing values or the outliers handled. The code for data wrangling is here

Tricky Part!

The Tricky part of data wrangling,

1. Knowing what is present in the 'null' cells, is it NaN or simply ' ' or whitespace(s)
2. In the non-null cells, if the all values are meaningful.
3. Recognizing that even if a column has all non-null and meaningful values, the future updates to the column may have problems. Hence need to expect it and handle it
4. See the data and think if certain value make sense for the business and decide to drop those which are not relevant
5. Data Imputation: Filling null cells with non-trivial substitutes-values other than mean, mode or median of that column
6. Ensuring not to introduce bias while imputation or dropping.

Cleaning Steps Performed:

1. **Identifying the columns to retain:** It is not smart to waste time cleaning up columns that may not turn out useful in modeling. Hence I chose to eliminate some in the first pass, exercising caution.
2. **Checking permit number,type and file dates:** These three variables are very essential and should be present in all the records.
3. **Converting Invalid weekdays:** DBI is open only from Monday-Friday. I attributed dates corresponding to weekends to typing mistake. Converted them to the nearer valid date. This may not be accurate (as typo may not be from n to n+1 or n-1 in day part alone), however it will at least prevent outliers in the EDA part.
4. **Filling the blanks in variables having only 'Y' as the entries:** In the application forms (both physical or online), normally the applicant is supposed to tick the option if applicable. Otherwise nothing needs to be done. Hence it is understandable that

blanks mean not applicable, a "No". Filled Fire only permit, Site Permit and Structural notification blank entries with “N”

5. **Filling nulls in qualitative variables with > 2 categories, which has numbers as entries:** 4 variables, existing construction type, proposed construction type, existing number of stories, proposed number of stories have NaN’s. 20% of the records are NaN’s in all the 4 variables. A few thousands of records have any one to three of the 4 variables with NaNs. It is best to leave them as they are, since it is building permit application. NaN corresponds to “not applicable” option for that permit type.
6. **Filling nulls in qualitative variables with > 2 categories, which has strings as entries:** Replaced existing use, proposed use variables with a more explicit “Unknown” instead of general NaN.
7. **Converting Location (Longitude / Latitude) into numbers:** NaN’s in this variables are substituted with 0’s. They are the same rows for which supervisor district and neighborhoods are unknown. We might end up retaining only only one of these three in the next stages. Longitude and Latitude were read as string types, so converted them to float numpy array of 2 elements.
8. **Interpreting too small values for costs:** Cost of the project is an essential part of the application according to this post. Still, this dataset does not follow that rule. There seem to be invalid entries. Cost columns are retained even if they are too small, because there are about 30% of them less than 500$, and median cost is found to be 7500$. NaNs are replaced with zeros and retained. Imputation can be done with EDA after understanding the data in detail.
9. **Dealing with rows without issue date:** The rows that do not have valid entries for issue date will yield NaN when we evaluate the wait time. But then they can not be dropped out of the EDA because that would lead to *selection bias* and give the perception of having less wait time in an average sense and would give less value for maximum wait times. We can not assume they were all issued today, because this would still introduce the bias. The only option seems to be leaving it as is and handle it during next stage. Perhaps, some recently filed ones can go into test set to simulate real life scenario.
10. **Saving the Clean file:** After all the cleaning steps, I wrote the dataframe to csv file so that the next step can pick it from there if needed.