Predicting Building Permit Issue Times

A Data Science Project

By Aparna Shastry

Content

- Business Problem
- Data / Data Wrangling
- Assumptions / Choices
- Exploratory Data Analysis
- Hypothesis Tests
- Predictive Modeling
- Conclusion
- Future Work

The Business Problem

- Building Permit: A Document issued by city council to builders
- Delays in issuance: Major inconvenience and Revenue loss, Refer:
    - Trulia Study
    - Vancouver City Problems
- This is a problem originally formulated by myself
- Possible Clients: Builders / Contractors / Real Estate Agencies
- How this can help:
    - Reduce uncertainty in when permit will be in hand
    - Plan the construction / alterations accordingly
    - Expect delays, proactively follow up
    - Know what factor in application causes delay and take care

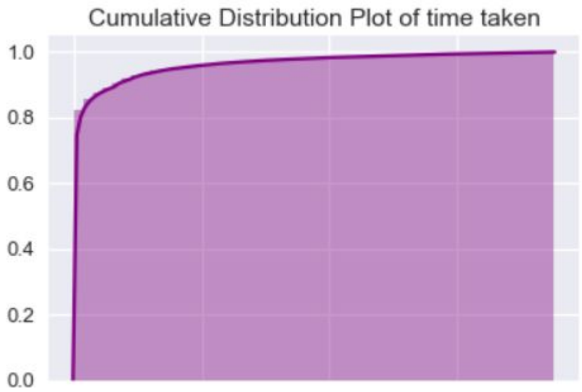
Data / Data Wrangling

- Any data with building permit application details would do
- Took San Francisco city open data, to make a prototype
- Raw data: 198900k records, 43 columns
  - Download Date Feb 25th, 2018
  - Database has starting from 1968, took from Jan 2013
- Cleaning:
  - Understanding all columns
  - Retaining useful columns and rows
  - Filling a few blank cells with “N” - Because in any application, one ticks only Y
  - Leaving a few blank cells as it is - Blank means not applicable
  - Converting object to float where necessary
  - Changing invalid weekdays into valid entries
- After clean up, left with 180811 records, 17 useful features

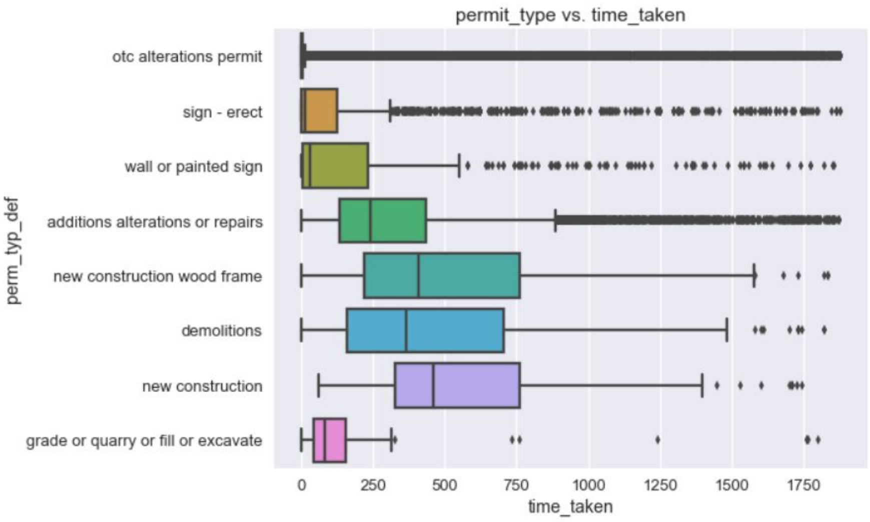
Assumptions and Choices

- Choices:
  - Taking only most recent 4-5 years of data.
  - Dropping rows corresponding to file date beyond Sept 30, 2017
  - Dropping rows with blanks for location
  - This project could go on and on, stopping at a logical point!
- Assumptions:
  - Certain blank issue dates to be same as download date, because the other option of dropping the records would make the mean/median wait time appear smaller!
  - Whenever current state is other than cancelled, withdrawn or disapproved, assumed good chance of issuance, hence retained.

Exploratory Data Analysis (1)

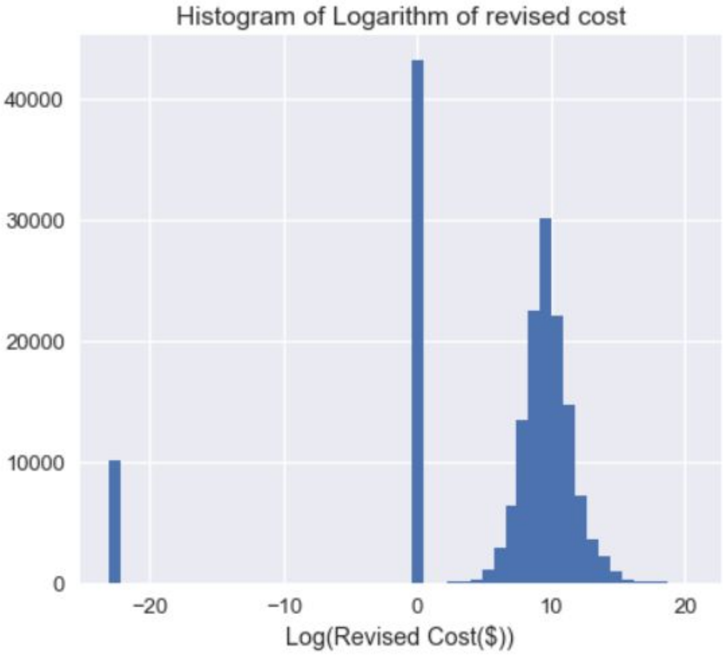


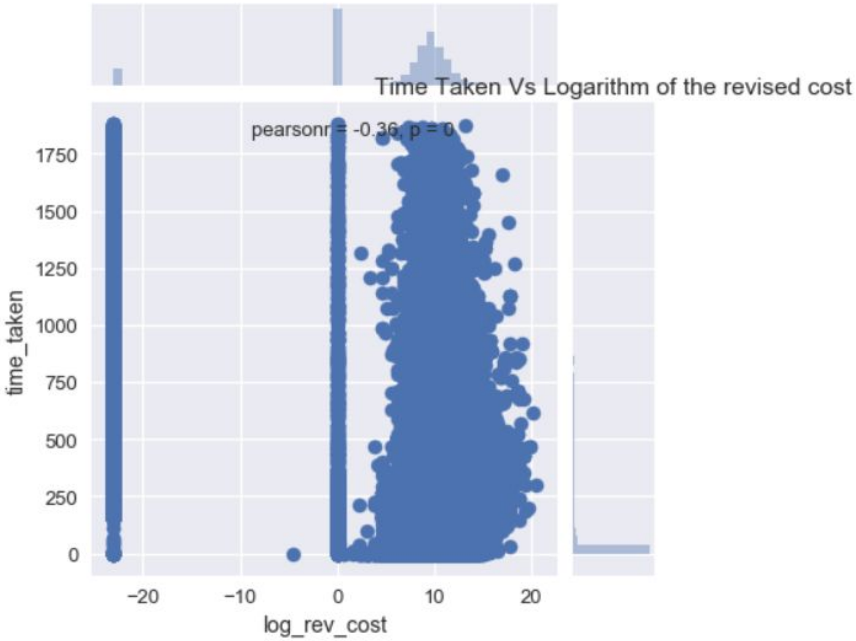
CDF of Time Taken to issue



Time taken by permit type

Exploratory Data Analysis (2)





Exploratory Data Analysis: key findings

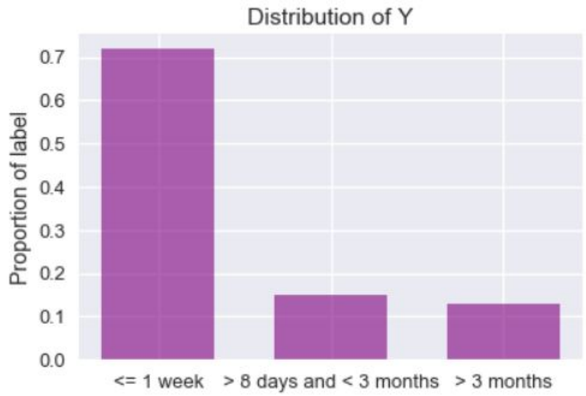
- Monday is normally least crowded day;
    - Best day to visit DBI.
- The data set has around 90% of OTC alterations permit types.
- OTC Alterations permit types are mostly issued the same day.
    - 75% of them are issued within 5 days.
- New construction type takes at least 2 months to get issued.
    - Median is about 1 year 3 months, wood is a bit less
- Median of time taken for alterations is close to 6 months
- Majority of the applications have one among 2 plansets.
- Applications without cost field filled take longer time to issue.
    - Recommended practice is to fill the cost field.
- Remaining findings and plots can be found in notebook.

Hypothesis Tests

- Assumed availability of only 7500 records, as a sample representing the entire population
- Checked conditions required
- One sample Population proportion test:
  - Is DBI’s claim that 65% of applications are processed same day true?
  - There is not enough evidence to believe so.
  - Observation: 57.73% with a margin of error of 1.12% applications are processed on the same day, with 95% confidence
- Two sample test for Population means comparison:
  - Is there a significant difference between mean issue times of fire only permit Vs non-fire only permit?
  - Yes, the 2 sample hypothesis test revealed so.
- In both tests, alpha used = 0.01

Predictive Modeling

- Definition of Machine Learning Problem: Predict the time taken window to issue the permit as,
  - Y = 0 if time < 8 days
  - 1 elseif time < 92 days
  - 2 else
- Why not estimate in days (regression)?
  - Tried and documented
- Why not just have 2 classes?
  - This is too trivial and not of much use even for learning
- Metrics used: Accuracy, Recall and weighted F1 score
- Volume under surface (VUS) was studied as concept, but not considered.
- Pairwise ROC is also not considered, as it is approximate of VUS

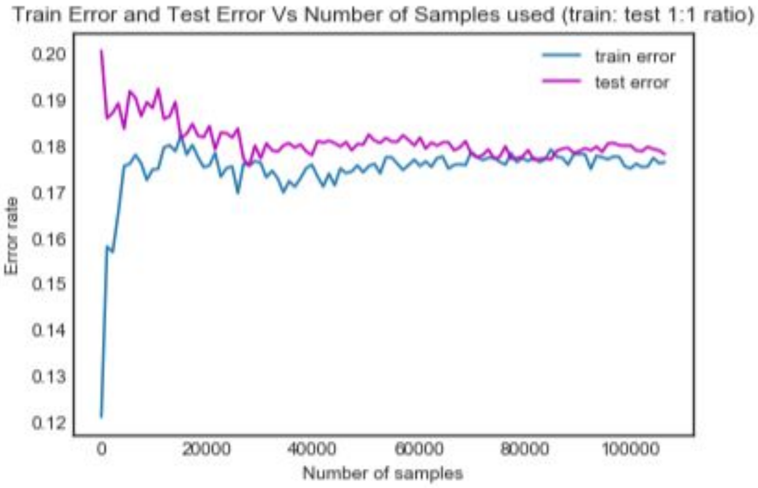


Training and Testing Method

- Data is split as Train : Validation : Test ratio 60:20:20 , stratified with target
- Hyperparameter tuning done using 5-fold cross validation(CV) method on Training data, then tested on validation set. Test set was not touched
- Score function used in CV is “f1 weighted”
  - There’s a plan to try some more
- Misclassifications from label 1 to 0 or 2 is not very good, as this is the class in between the other two
- Several methods of balancing possible
  - Undersampling No Class
  - Synthetic Oversampling Yes Class (SMOTE)
  - Giving more penalty on misclassification of Yes to No class
- Undersampling and the penalty methods are tried.

Inherent Bias in the Data

- A logistic regression model was run with increasing the train and test samples and recording the error.
- Train and Test Errors reached steady state like shown in the figure.
- This gave an initial hint on how much improvement might be possible with high variance models (not much!)



Feature Engineering

- Revision cost was converted to Log of revision cost
- Square and cube of revision costs were generated
- dff_use = existing_use != proposed_use,
- diff_story = existing_use != proposed_story
- Week day and month were extracted from filing date
- Location was extracted from string, and opened up as latitude and longitude

Comparisons of Models Tried

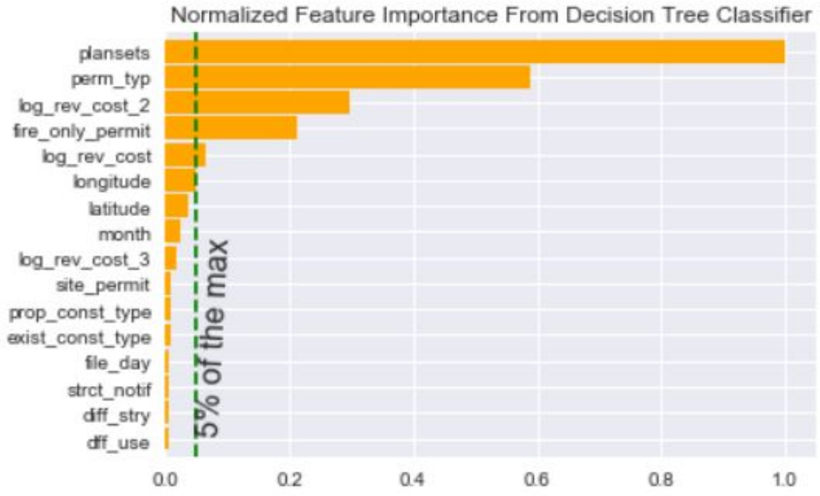
Model Name	Hyperparameters Tried	Hyperparameters Selected	Sensitivity on labels 0,1,2	F1 score
Logistic Regression	C=[0.1,1,10,100,10000],class_weight:['balanced',None]	C = 0.1, class_Weight = balanced	0.82,0.75,0.67 Overall 79%	0.81
Decision Tree	max_depth: [6,8,12,14], min_samples_leaf: [1,2,4,6],class_weight:['balanced',None]	max_depth=12,min_samples_leaf=6,class_Weight = None	0.94,0.41,0.67 Overall 83%	0.82
Random Forests	Same as above and n_estimators: max_depth: [6,8,12,14,None],[10,50,100,200]	max_depth=None,min_samples_leaf=2,class_Weight = balanced,n_estimators =300	0.87,0.69,0.73 Overall 84%	0.84

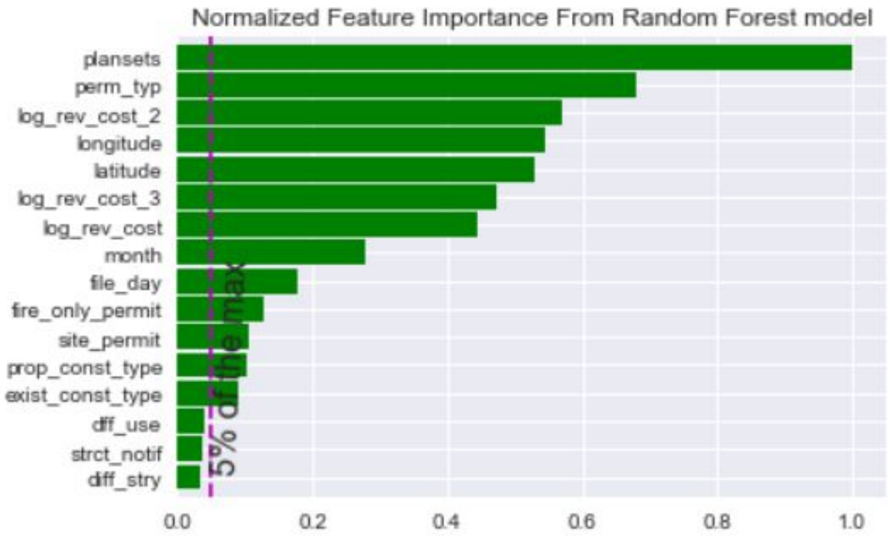
Feature Selection

- Decision Tree Classifier got saturated with 4 features
- Random Forest Classifier: Summarized for a few features below.

Features	**Accuracy and f1 score**
Plansets alone	63.73%,0.65
Plansets, Permit type	69%, 0.73
Plansets, Permit type, log_rev_cost_2	73.65%,0.77
Plansets, Permit type, log_rev_cost_2, longitude	80%,0.81
Plansets, Permit type, log_rev_cost_2, longitude, latitude	**82%,0.83**
Plansets, Permit type, log_rev_cost_2, longitude, latitude, log_rev_cost, log_rev_cost_3, month, file_day	83%, 0.83

Feature importance





Model Selection and Conclusions

- **The Random Forest** is currently chosen to be the model to be used.
  - This might change based on future work!
- It is found that the times are dependent on Latitude and Longitude, this suggests correlation with location data
- Point above opens up several possibilities to collect more data
- Logarithm of revised cost and its engineered versions are important features
- Finally 5 features are found to be deterministic factors **Plansets, Permit type, log_rev_cost_2, longitude, latitude**

Scope for Further Work

- Can gather more location based data, like housing price or crime data to reduce the bias
- Try undersampling, oversampling techniques to handle class imbalance
- Review the problem definition from business sense once again
- Try cross validation with other score functions

Current code is here