Predicting Blight of Buildings in Detroit

--------- Capstone Project of "Data Science At Scale" on Couresa

Yanting Zhang 02/28/2018

This is a summary of the capstone project of "Data Science at Scale" on Coursera by University of Washington. The problem of interest is to predict if a building in the city of Detroit will be blighted or not by using the real incident records of violation, crime, 311 calls, and demolition permit.

Predicting blight of buildings is an important topic for city planning and intervention. Many cities like Los Angeles, Chicago etc have great effort in developing large systems to promote information availability and research in this field.

How to start from scratch with real data to a model generating good predictions is described in 6 sections below: getting to know the raw data, defining buildings to group data, model formulation, model feature engineering, feature selection, model training and ensemble, and summary and conclusion.

Section 1: Getting To Know The Raw Data

Below are the 4 data files (label 1-4) from Coursera assets and one additional data file (label 5) downloaded externally from https://data.detroitmi.gov/Property-Parcels. The reason of adding the fifth file is to complement the first 4, which will become more clear in the next section.

Data from Coursera assets:
1.detroit-blight-violations.csv : Each record is a blight violation incident.
2.detroit-crime.csv: Each record represents a criminal incident.
3.detroit-311.csv: Each record represents a 311 call, typically a complaint
4.detroit-demolition-permits.tsv: Each record represents a permit for a demolition

External data downloaded from https://data.detroitmi.gov/Property-Parcels:
5.parcels_in_Detroit.shp: Each record represents a property parcel in Detroit

The first 4 files can be loaded into R as data frames. For simplicity, they will be respectively called data of violation, crime, 311-call and permit thereafter. As shown above, each row of them is respectively an incident of a violation, a crime, a 311-call and a permit of demolition. And the columns contain miscellaneous variables such as time, address, type and so on for the corresponding incidents.
.
The last file (label 5) can be loaded into R as a spatial object from which there is a data frame. Each row of the data frame represents a parcel and each column is a parcel

variable such as parcel area and so on. Thus this data frame will be called parcel data for simplicity.

Table 1 summarizes the basic information of the 5 data frames. It can be learned that the factor variables having many levels can be a challenge for model training.

Table 1. Raw Data Basic Information

Data	Size (Mb)	Dim (Row x Col)	Number of Factor Variables	Median Number of Levels Per Factor Variable
violation	155.328	307804x31	27	265
crime	27.281	119931x17	8	374
311-call	6.067	19680x15	11	13660
permit	4.295	7133x55	48	129
parcel	242.896	383971x53	31	612

As expected in almost all real data, there are variables with missing or unreasonable values. Such variables will be treated case by case when necessary.

Section 2: Define Buildings To Group Data

The goal of the model is to predict the blight of a building, but the data of violation, crime, 311-call and permit is for an incident per row. Which building does an incident belong to?

In order to have a good model, it is important to assign an incident to its correct building as close as possible. Imagine an extreme case that building A which was blighted has very few violations while building B which was not blighted has a lot of violations. In this case, the model trained using the number of violations would not make sense. Therefore the parcel data was added to assist in assigning an incident to its building.

What defines a building in the data? Table 2 is a summary of the variables which can be used and their pros and cons.

Table 2. Variables For Defining Buildings

Variable Name	Building Definition	Availability in Data	Pros	Cons
Parcel ID	One building per Parcel	Data of permit & parcel only	Mostly unique** ; Additional parcel info for modeling	Not available in all data
Incident Address	One building per address	All	Sometimes unique	There are typos, format variations, incident addresses different from building addresses
Incident Location:(Lng , Lat)*	One building per defined range	All	Easy to use	Need to define the range/boundary for a building

**(Lng,Lat): (Longitude, Latitude)*
***uniqueness of a parcel ID: It's generally true that one parcel is used for one building. But it's possible that one big parcel is shared by more than one building.*

From Table 2, it's easy to tell a parcel ID would be the best to define a building, because one parcel is indeed used for one building most of the time. What is even better, the building with known parcel ID would have the parcel variables in addition to the incident variables for modeling. Therefore each parcel ID was used to represent one building. Special cases that one parcel is used for more than one building were not considered in this work.

But parcel ID is not available in the data of violation, crime and 311-call. Either address or (lng, lat) needs to be used to define buildings. How to use them and which is better? A few methods are introduced in Table 3 and their pros and cons are compared in Table 4. Please refer to Table 3 to learn about how each method works. Table 4 will be used to explain why one method was more favored than other for a dataset.

Ideally if the address and (lng, lat) are both accurate, "raw address" and "closest parcel" should be both close to the real number of buildings. Since the raw incident addresses usually have typos and format variations for the same address, "simple address + same 2-decimal", which is designed to reduce the impact of typos and format variations, is expected to give fewer buildings than "raw address". Therefore the number of buildings says something about the data and the methods. The total number of buildings generated by these methods are compared in Table 5 .

Table 3. The Methods Of Defining Buildings With Address or (Lng, Lat)

Method Name	Method Details
"raw address"	One building per incident address which was unprocessed raw data. Incidents which share the same raw incident address belong to the same building.
"same 3-decimals"	One building per value of (lng, lat) when it is rounded to 3 decimals. Incidents which share the same (lng, lat) which was rounded to 3 decimals belong to the same building
"closest parcel"	One building per parcel Incidents of which the (lng, lat) are within or closest to the same parcel belong to the same building. (lng, lat) of the incidents' locations are compared against those of the parcel polygons, for example, (lng, lat) of the center or the edge of the parcel polygons.
"simple address + same 2-decimals"*	One building per address-(lng,lat) combination. Incidents which share the same "simple address" and the same (lng, lat) when it is rounded to 2 decimals belong to the same building. "simple address" is the first component of the full incident address like the street number. "same 2-decimal" is expected to serve the purpose of the rest parts of the full incident address beyond the first component without being affected by the typos and different formats. "simple address + same 2-decimals" is based on the assumption that the first component of the full incident addresses like street numbers are mostly unique within a range confined by (lng, lat) rounded to 2 decimals, which is about 10000 feet scale.

It's easy to tell from Table 4 that "closest parcel" is better than the others if (lng, lat) is good enough. In Table 5, the number of buildings from " closest parcel " is slightly smaller than that from "raw address". This suggests (lng, lat) is good and consistent with the incident addresses. Thus "closest parcel" is chosen for the data of crime and 311-call to generate buildings with parcel IDs.

For the violation data, " closest parcel " generates about 10x fewer buildings than "raw address", which suggests problems in either (lng, lat) or the incident addresses. If the problem was caused by too many typos or format variations in the incident addresses, wild difference is also likely to be seen between "raw address" and "simple address + same 2-decimals". Since "simple address + same 2-decimals" only uses the first component of the raw addresses, the impact of typos and formats should be reduced a lot. But the two methods have very close number of buildings which suggests reasonable quality of the raw incident addresses. Thus the problem is more likely to be in (lng, lat). It's known from initial data inspection that some incidents of violation do not have good (lng, lat). For example, about 21114 records in the violation data with different incident addresses have the same (lng, lat) located exactly in the city center. In this case as well as cases when precision of (lng, lat) is low, the (lng, lat)-based methods would generate much fewer buildings than the actual number. Considering the data quality analyzed above and the pros and cons compared in Table 4 for different methods, "simple address + same 2-decimals" was chosen for the violation data.

Table 4. Comparison Of Building Defining Methods

Method Name	Advantages/Disadvantages	Details
"closest parcel"	Advantages	1. it assumes one parcel per building which is very realistic
		2. it automatically comes with the parcel ID which adds extra parcel variables for modeling.
	Disadvantage	1. it requires data (lng, lat) to have enough precision
"raw address"	Advantage	1.it is a quick way to get a rough estimation of the number of buildings
	Disadvantages	1*. it is expected to give more buildings than the actual number, because one building per incident address can be wrong (see the notes below the table)
		2. obtaining the parcel ID is difficult
"same 3-decimals"	Advantage	1. it is easy to be implemented in code
	Disadvantages	1. it assumes a square about 1000x1000 sq feet for each building which is not very realistic.
		2. it requires data (lng, lat) to have enough precision
		3. one building can have more than one parcel IDs
" simple address + same 2-decimals "	Advantage	1. it is less affected by typos and format variations in the raw incident addresses
		2. obtaining the parcel ID is easier than "raw address"
	Disadvantage	1. it could be a very rough estimation if the first component of the incident addresses like the street numbers are not unique in the area confined by (lng, lat)

**The addresses of incidents belonging to the same building can be different in the following cases:*
1. the address has typos and different formats (e.g. street vs ST)
2. the address has different unit numbers for the same multi-unit building
3. the address for the incident occurred in or close to the building is not the building address, but for somewhere close

Table 5. Total Number Of Buildings From Different Methods

	"raw address"	"same 3-decimals"	"closest parcel"	"simple address + same 2-decimals"
# of Buildings in Violation	110837	24327	16818	110474
# of Buildings in Crime	57324	26919	54939	34263
# of Buildings in 311-call	17907	11316	13968	16523

The methods chosen for each data file to define buildings is summarized in Table 6. It's reassuring to see that the mean records per building is consistent and reasonable. It's worth mentioning that only 88651 out of 110474 buildings generated by "simple address + same 2-decimals" from the violation data have matches in the parcel data. This could be caused by the deficiency of the method or the fact that one building could have

multiple incident addresses. To take advantages of the parcel variables, only the buildings with parcel ID were used in model training in this work.

Table 6. The Building-Defining Methods Chosen For Data

Data File	Raw Variables For Defining Buildings	Building Defining Method	mean # of records per building	# of buildings with parcel ID
Parcel	parcel ID	one building per parcel ID	1	383971
Permits	parcel ID	one building per parcel ID	1.1	6182
Violation	incident address + (lng, lat)	"simple address + same 2-decimals"	2.8	88651*
Crime	(lng, lat)	"closest parcel"	2.2	54939
311Calls	(lng, lat)	"closest parcel"	1.4	13968

**Only 88651 out of the 110474 buildings in the violation data had matches in the parcel data.*

Section 3. Model Formulation

The question of interest is if a building is blighted or not. This is a 2-class classification problem which requires the following ingredients for modeling:

1. the response variable. The response variable was the buildings' blight status which is 1 for blighted and 0 otherwise. This work assumed that all buildings in the permit data were blighted. Special cases that buildings with permits were not blighted and buildings without permits were blighted were not considered. Since the variable of the demolition permit is used as the response variable, it cannot be used as a feature for model training.

2. at least one predictor variable. The number of violations (called "vio_count") for example can be a predictor.

3. equal sample size for each class of the response variable. Table 6 shows there are 6182 buildings with demolition permits (response =1). Thus another 6182 buildings without permits (response = 0) are needed.

Where to draw the samples with response "0"? Sampling population should be consistent between class 0 and 1. The characteristics of the samples determine what population the model is applicable for. Plot 1 shows that the blighted buildings (class "1") does not have any special composition of the incident records. Consistently the non-blighted buildings (class "0") can be zero and nonzero for any incident too. Therefore the parcel data was used for class-0 sampling. Table 7 shows that the models which used the same one predictor "vio_count" have very different accuracies. This is because the class-0 samples not drawn from the parcel data in Table 7 are biased to have nonzero incidents.

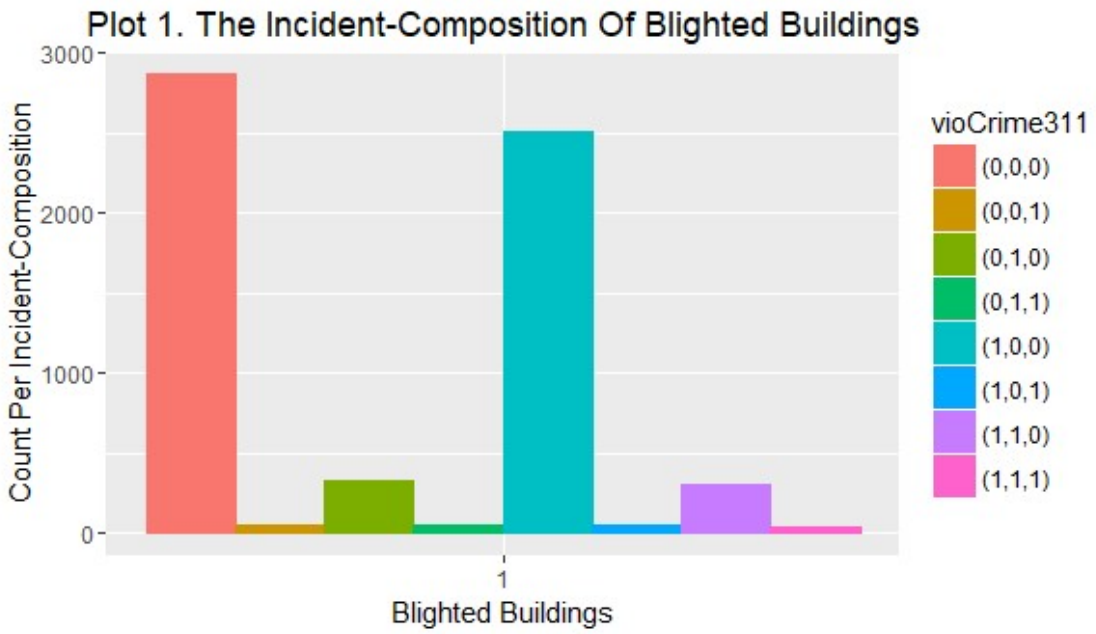


Table 7. 1-predictor Model With Different "0"-Class Sampling

Predictor Name	Sampling Population For Class "0"	Model Accuracy
"vio_count"	Violation Data	0.7652
"vio_count"	Data of Violation, Crime and 311-Call	0.5833
"vio_count"	Parcel Data	0.6211

"vio_count": the variable which is the number of violations per building

Section 4. Model Feature Engineering

To improve from the 1-predictor model, more relevant features are needed. There are about total 200 variables in the data. But there are problems of using them as features.

1. many variables are categorical with far more than 2 levels, which would not be efficiently handled by the methods conventionally used for two levels.

2. some variables have missing or unreasonable values which are not good enough to be represented by 0 or NA. For example, 12249 records in the violation data have for example "year" =38706 etc.

3. many variables are specific to an incident (violation, crime or 311-call).
1) the incident-specific variables are not valid per building. For example, the variable "type" is well defined for each incident , but not for a building, since a building could have 20 different types of violations/crimes/311-calls.

2) the incident-specific variables would have missing values for buildings without the corresponding incidents. For example, what is the value for the variable "violation year" for buildings without violations?

How to engineer features from the variables like above? Below are the methods used in this work.

1. To conveniently study the many-level categorical variables using algorithms like RF and XGB, they are made numeric by replacing each their level value with the corresponding level count

For example, the categorical variable "taxstatus" has 29 levels ---"board of education", "hospital", "city land bank" ... and the corresponding level count is respectively 767, 93, 45039,... After making the variable numeric, " taxstatus" has values of 767, 93, 45039,...

2. To better represent the missing or unreasonable values for the variables of interest when necessary, linear model is used for extrapolation when applicable.

For example, the unreasonable hearing year of a violation incident can be fixed by extrapolating using the violation ticket ID which has a linear relation with the hearing year.

3. To make an incident-specific variable a valid building feature, aggregate the records belonging to the same building by using appropriate statistics (e.g. sum/mean/frequency...) or adding new features

Here are some examples.
1). If the violation-specific variable is numeric, for example, "FineAmt" can be used to engineer a new building feature "mean_vio_fine" as below:
For each building, if vio_count>0, mean_vio_fine=(total violation fine per building)/vio_count per building, else mean_vio_fine=0
2). If the violation-specific variable is categorical with levels far more than 2, for example, "type" can be used to engineer a new building feature "weight_vio_type" as below:
For each building, if vio_count>0, weight_vio_type =(level count summed over all levels per building/ vio_count per building/total number of samples), else weight_vio_type =0
3). If the violation-specific variable is categorical with levels close to 2, for example, "PaymentStatus" has 3 levels--"paid in full", "partially paid", "no payment". 3 new features could be added ---"vioFine paid in full", "vioFine paid partially", "vioFine not paid", the values of which can be the corresponding level count or level frequency. This method was not good for the variables with far more than 2 levels since it could add a lot of noise.
4). The old variables of the year for the violation, crime and 311-call can be combined to create a new variable "incident year" as below:
For each building, if vio_count+crime_count+311_count>0, incident year = latest year of all incidents, else incident year= 3000 (pick any unrealistic year number)

Section 5. Feature Selection And Model Training

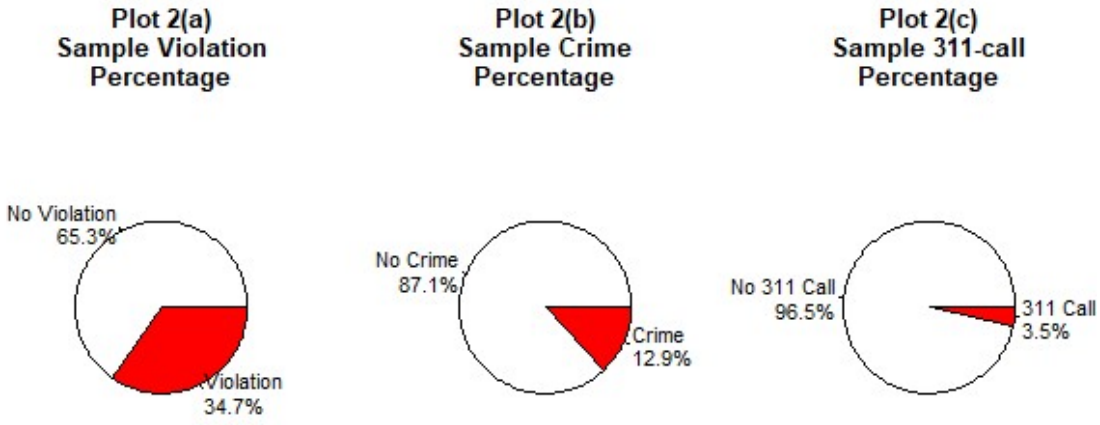
In machine learning, sometimes less is more. Irrelevant features can make the models worse. Even the relevant features can be double-edged sword: while they usually help to improve the models, too many can add more model variance than the bias reduction. Therefore feature filtering and selection is important.

There are 2 categories of features--Not-incident-specific and incident-specific, as summarized in Table 8. The incident-specific features were more likely to have reduced predicting power than the Not-incident-specific ones, since they cannot differentiate the samples which don't have the corresponding incidents. Plot2 shows that in a random sample which consists of equal number of blighted or non-blighted buildings, 34.7% has violation count>0, 12.9% has crime count>0, 3.5% has 311-call count>0. This means the violation-specific, crime-specific, 311-call-specific features cannot differentiate respectively 65.3%, 87.1%, 96.5% of the samples, which is significant reduction in the predicting power. It's worth mentioning that the variables directly related to the demolition permit (e.g. permit_count) are not discussed here, because the permit is used as the response variable and the variables directly related to the permit cannot be used as features.

Table 8. Categories of Building Features

Feature Category	Data Source	Example Feature Name	Expected Predicting Power
Not Incident Specific	parcel	taxstatus, total_sqft,…	good for all buildings
Incident Specific	violation	vio_count, mean_vio_fine, freq_vio_type,…	reduced for buildings without violation
	crime	crime_count, freq_crime_type,…	reduced for buildings without crime
	311-Call	311_count, mean_311_rating,…	reduced for buildings without 311-call

Plot 2. The Percentage of Incidents In A Random Sample



Section 5.1 Selection Of Incident-specific Features

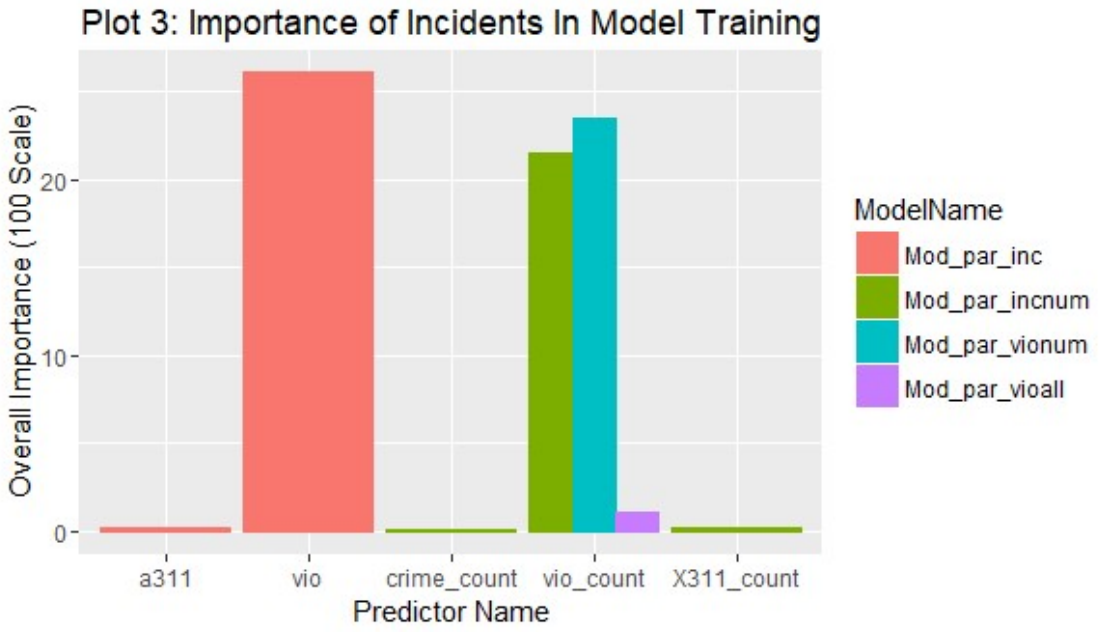
Many incident-specific features can be engineered from the incident data, but the following 6 are the most basic, and all other incident-specific features are contingent on their values.

1. the incident-indicators (called "vio", "crime", "a311") which is 1 if the corresponding incident per building is nonzero and 0 otherwise
2. the incident-count (called "vio_count", "crime_count" "X311_count") which is the number of corresponding incidents per building.

To know how many incident-specific features are needed to improve the model, different subsets of features were used to train models with 5-fold cross-validation using the tree-based XGB. The results are summarized in Table 9 and the relative feature importance in each model is plotted in Plot 3 if available.

Table 9. Models (XGB Tree) Trained With Different Subsets of Features

Model Name	Predictors	Accuracy	Accuracy SD
Mod_par	parcel only	0.8604	0.0057
Mod_par_inc	Parcel + incident indicators	0.8678	0.0032
Mod_par_incnum	Parcel + incident counts	0.8682	0.0112
Mod_par_vionum	Parcel + vio_count only	0.8665	0.0121
Mod_par_vioall	Parcel + all violation related features	0.8674	0.0134



In Table 9, only the third decimal of the accuracy is different among the 5 models. It's worth mentioning that since the accuracy are pretty close especially between the last 4 models in Table 9, using different random seed or tweaking the model parameters slightly could change how the models are ordered in accuracy.

In Plot 3, the importance score of "vio" or "vio_count" is above 20 on a 100 scale except in "mod_par_vioall" which was fed total 24 violation-specific features. The low score of "vio_count" in this model is because other two violation-specific predictors (not shown in the plot) were used and scored 10 each by the model. Plot 3 also shows that the importance score for crime and 311-call in the two models they were fed to is either slightly above 0 or not available (predictor "crime" is not plotted because it was not picked by the model thus dose not have an importance score).

The following are suggested by Table 9 and Plot 3:

1. as expected, the incident-specific features have less predicting power than the parcel features. The incident features only improved the accuracy by less than 1% in Table 9.

2. Including the violation-specific feature in model training is better than excluding it . Adding "vio_count" improves the model by about 0.7% in Table 9.

3. more violation-specific parameters than "vio_count" do not cause more improvement. "mod_par_vioall" was fed 22 more violation-specific features than "mod_par_vionum", but it did not perform better.

4. the crime-specific and 311-call-specific features have very minimal if any impact on the model accuracy. "mod_par_vionum" without crime-specific and 311-call-specific features have accuracy very close to those which have them.

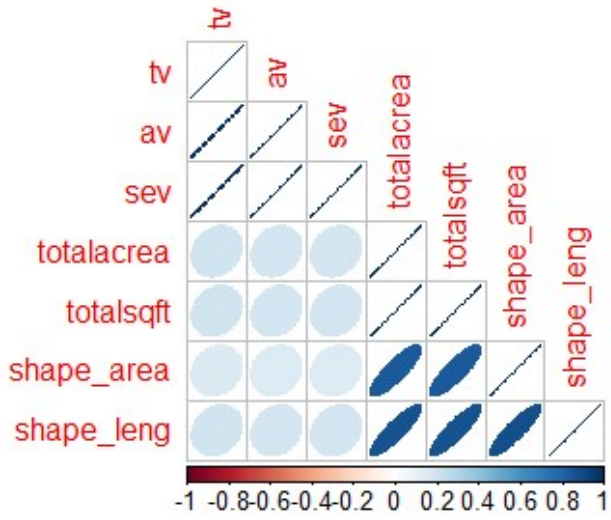
Based on the findings above, only "vio_count", "crime_count" and "311-count" was kept for model training and all the rest incident-specific parameters were left out.

Section 5.2 Selection Of Parcel (Not-Incident-specific) Features

In the models above, the features in the parcel data are all kept for training without a careful look so far. Are there redundant or irrelevant ones? The methods below were used for feature selection.

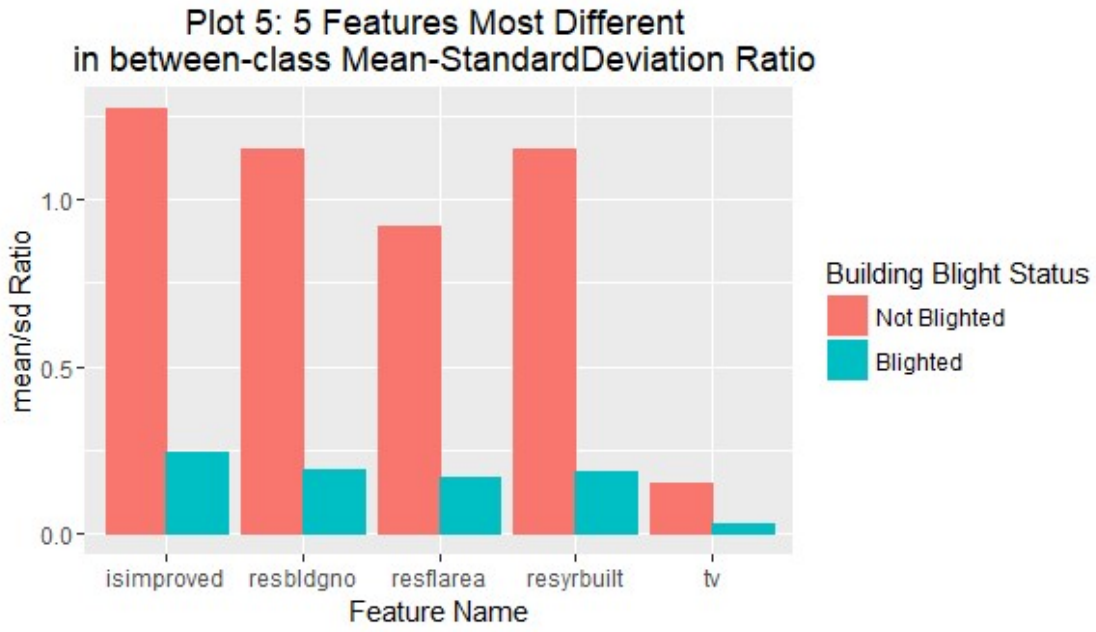
Correlation matrix was used to check the correlation between the continuous numerical variables. Some parcel features are highly correlated, as shown in Plot 4 , the variable "tv" is perfectly correlated with "av", "sev", and "totalacrea" is perfectly correlated with "totalsqft". Co-linearity and multi-colinearity is known to increase the model variance in linear regression, though the model accuracy is usually not affected. The impact of co-linearity and multi-colinearity on nonlinear models like RF is less well-known, but it's shown in some studies the machine learning algorithm is less likely to find other independent predictors given Co-linearity and multi-colinearity. Therefore it's good to remove the highly-correlated predictors, (e.g., correlation coeficient (c.c.) >0.95) .

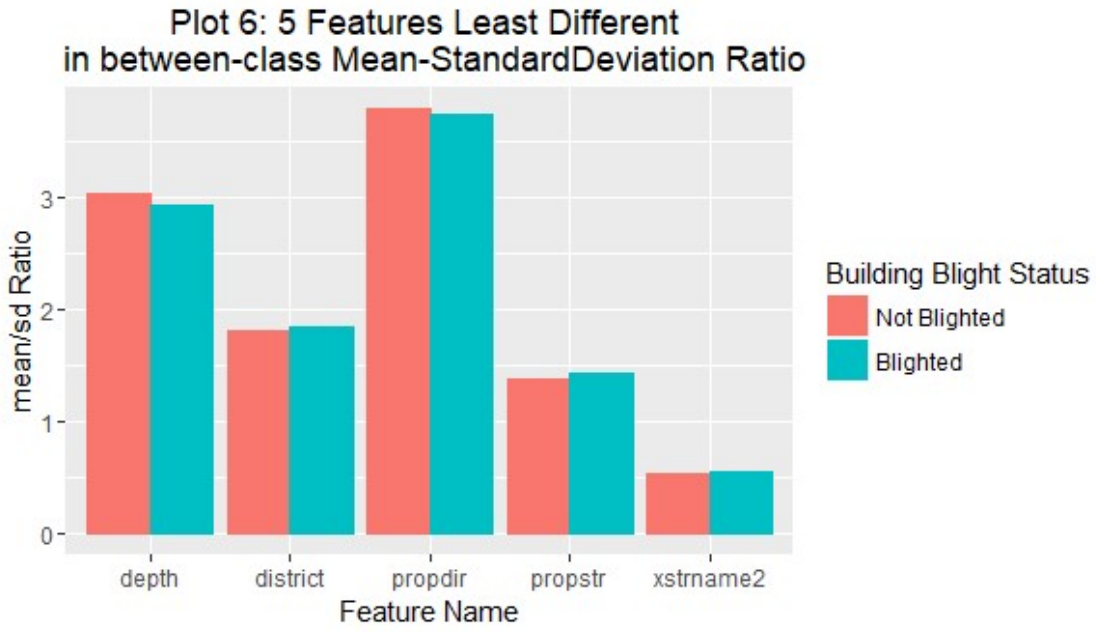
Plot 4: Correlation Plot of Parcel Features



As the problem of interest is a 2-class classification problem, good features are expected to be different in some way between class 0 and class 1. The ratio of Mean-to-standard deviation (SD) ratio was used to measure the difference of the variable distribution. Plot

5 and Plot 6 below respectively shows the 5 features which are most/least different in the ratio of mean-to-SD.





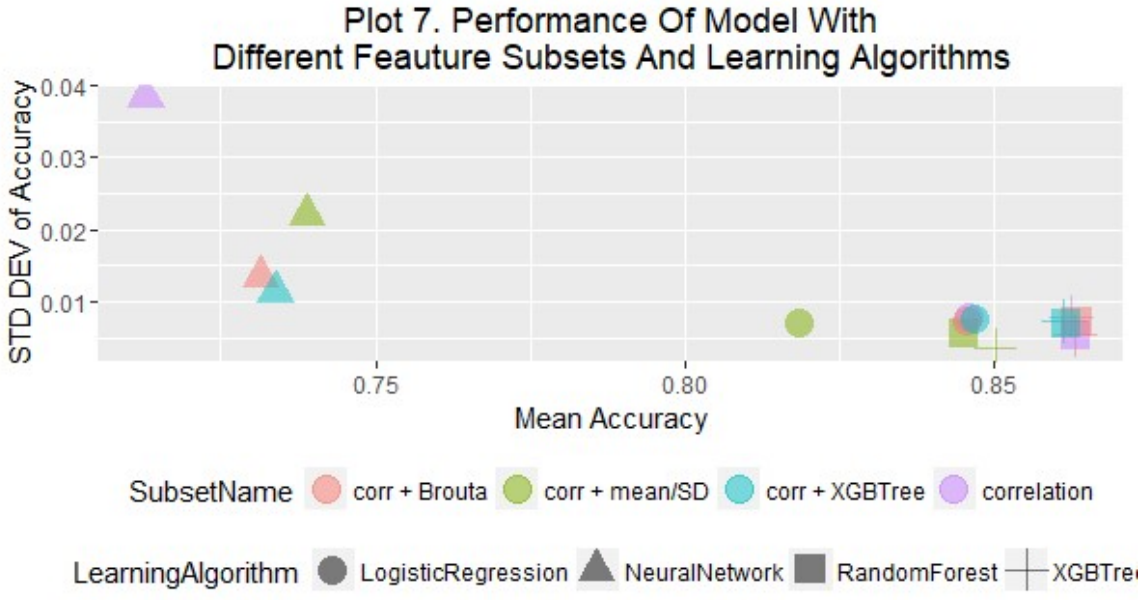
Besides the correlation and mean-SD ratio, Brouta which is a feature selection package built in R and the machine learning algorithm like XGB Tree was also used for feature selection.

Section 5.3 Model Training And Ensemble

The methods in Section 5.2 were used alone or combined to generate several feature subsets which are named and explained in Table 10. The samples were partitioned into training and testing dataset with 4:1 ratio. Different machine learning algorithms were used to train models with 5-fold cross validation. The mean accuracy and accuracy standard deviation are plotted in Plot 7.

Table 10. Subset of Features And The Feature Selection Method

Subset Name	Num of Features	Feature Selection & Filtering
"correlation"	53	1. no features with c.c.>0.95
"corr + mean/SD"	32	1. no features with c.c. >0.95
		2. features with different mean/SD between class 0 and class 1
"corr + Brouta"	47	1. no features with c.c. >0.95
		2. features picked by the Brouta Package
"corr + XGBTree"	25	1. no features with c.c. >0.95
		2. features with importance score >2 from XGB Tree



It's easy to see that there is not a best learning algorithm or a best subset of features in Plot 7. The learning algorithms in Plot 7 have very comparable performance except Neural Network. The different subsets of features do not show significantly different model performance either except the subset " corr+mean/SD " , which consistently give a

lower accuracy than the other subsets in models trained with the same learning algorithms (except Neural Network).

What is more, considering the number of features in Table 10 as well as the model performance in Plot 7, it seems better to have redundant or irrelevant features for model training than to miss the relevant features. Subset "correlation" has the largest number of features and subset "corr+XGBTree" has the smallest number of features, but in Plot 7 the models trained on them using the same algorithm (except Neural Network) have pretty close performance.

It's also not straightforward to pick the best model from Plot 7 because of the bias-variance trade off. Ideally, the best model is expected to have both the highest accuracy and the lowest variance located in the most right and bottom corner in Plot 7. But a few models are very close and the model with the highest accuracy does not have the lowest variance in the plot.

An ensemble model which combines far more than one model for prediction can increase the accuracy and decrease the variance at the same time. An ensemble model was created by using 12 models in Plot 7 through the simple voting strategy: the final prediction is 1 if more models vote for 1 and 0 otherwise. The 4 models trained with neural network were excluded from ensemble because of the much lower accuracy.

To compare the performance of the ensemble model and the single model, 5 different random seeds were used to partition the data for training and testing. And the mean accuracy and the accuracy standard deviation is plotted in Plot 8. It's shown that the ensemble model indeed has higher accuracy and lower variance than most of the single models. It's worth noting that there is a single model which had the same accuracy with even slightly lower variance than the ensemble model. It is the model trained with tree-based XGB on the subset "corr+Brouta" generated by the R-built in package "Brouta". No wonder XGB has become the most popular machine learning model in Kaggle competitions.

Summary And Conclusions

This report describes how to start from scratch to train a model which predicts the blight status of a building in Detroit by using the real incident records of violation, crime, 311 calls and blight permit in the city.

Section 1 was a basic introduction about all the data used in this work. Section 2 explained why the Building IDs were needed to label the incident data and how the labels were created. Section 3 translated the problem about predicting the blight status of a building into the language of a 2-class classification model. Section 4 talked about the variables in the data which could not be directly used for modeling and discussed how to engineer features from them. Section 5 studied the incident-specific and Not-incident-specific features and performed feature selection and model training by experimenting with different feature subsets and learning algorithms.

A few things can be learned from this project, which can be useful in general machine learning:

1. It's important to study the data in relation to the question of interest BEFORE training models. For example, in this project, data analysis can uncover the limited predicting power of the incident data of violation, crime and 311-call, which can save tremendous efforts in dealing with the irrelevant model features.

2. There may not be a best subset of features or a best learning algorithm. Models trained on different feature subsets with different learning algorithms can have very close performance. It's better to have redundant or irrelevant features in model training than to miss relevant ones.

3. There may not be a best model in both accuracy and variance. An ensemble model which combines far more than one different models with close accuracy can improve in both accuracy and variance. But it does not guarantee to be better than a single model.