Team Name

The Three-Body Problem

Team Members

Bei Zhang

Ruochen Geng

Yao Lu

November 13, 2018

Table of Contents

Executive Summary

The goal of this project is to identify potential fraudulent records in the New York City property data through unsupervised machine learning models and to analyze some of the examples of top fraudulent records identified by the algorithms.

The dataset consists of over 1 million records of properties in New York City which includes information about the location, usage classification, lot and building sizes and assessed values. Since property taxes are strongly related to the assessed value of the property, the actual value of the property could be misrepresented for tax evasion purposes. For the fraud analysis, many of the factors such as building and lot size, together with the geographical location have been put into consideration when considering the accuracy of the final property value.

The following methods have been applied to the data to determine if the assessed value of the property had been misrepresented. The data had been standardized and components were isolated using PCA (principal component analysis). From there, an autoencoder and mahalanobis distance algorithms have been applied to the data to determine whether a given data entry is an outlier compared to the general trends. Each algorithm has assigned a fraud rank to the data, and the ranks have been combined to get a more accurate result. Both algorithms gave very similar results to entries that have been deemed more fraudulent, and vice versa.

Finally, the top 10 outlier entries were further analyzed to determine how they might have been flagged as a fraudulent record by both algorithms, and to estimate whether they were possibly erroneously entered or fraudulently misrepresented.

Data Description

This part provides data description for New York property data in November 2010. There are 1,048,575 records and 30 fields in the data.

Numeric variables:

#	variable	count	mean	std	min	max	% populated	0 value
1	**LTFRONT**	1048575	36	74	0	9999	100	168867
2	**LTDEPTH**	1048575	88	75	0	9999	100	169888
3	**STORIES**	996433	5	8	1	119	95.03	0
4	**FULLVAL**	1048575	880488	11702927	0	6.15E+09	100	12762
5	**AVLAND**	1048575	85995	4100755	0	2.67E+09	100	12764
6	**AVTOT**	1048575	230758	6951206	0	4.67E+09	100	12762
7	**EXLAND**	1048575	36812	4024330	0	2.67E+09	100	484224
8	**EXTOT**	1048575	92544	6578281	0	4.67E+09	100	425999
9	**EXCD1**	622642	1605	1388	1010	7170	59.38	0
10	**ZIP**	1022219	10935	527	10001	33803	97.49	0
11	**BLDFRONT**	1048575	23	36	0	7575	100	224661
12	**BLDDEPTH**	1048575	40	43	0	9393	100	224699
13	**AVLAND2**	280966	246365	6199390	3	2.37E+09	26.8	0
14	**AVTOT2**	280972	716079	11690165	3	4.5E+09	26.8	0
15	**EXLAND2**	86675	351802	10852484	1	2.37E+09	8.27	0
16	**EXTOT2**	129933	658115	16129808	7	4.5E+09	12.39	0
17	**EXCD2**	90941	1372	1105	1011	7160	8.67	0

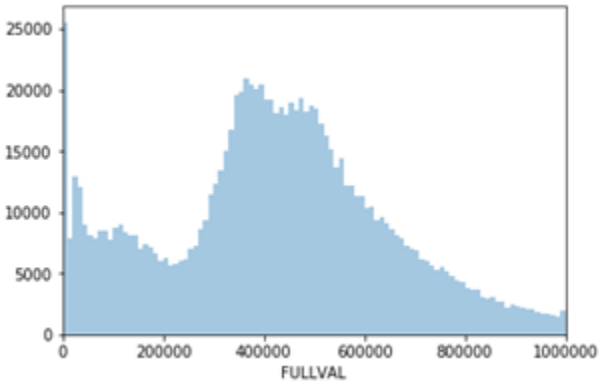
Categorical Variables:

#	variable	# NaN	# unique	% populated	most common value
1	**BBLE**	0	1048575	100	4017830023
2	**BLOCK**	0	13949	100	3944
3	**LOT**	0	6366	100	1
4	**EASEMENT**	1044532	12	0.39	E
5	**OWNER**	31083	847053	97.04	PARKCHESTER PRESERVAT
6	**BLDGCL**	0	200	100	R4
7	**TAXCLASS**	0	11	100	1
8	**STADDR**	641	820637	99.94	501 SURF AVENUE
9	**EXMPTCL**	1033583	14	1.43	X1
10	**PERIOD**	0	1	100	FINAL
11	**YEAR**	0	1	100	2010/11
12	**VALTYPE**	0	1	100	AC-TR

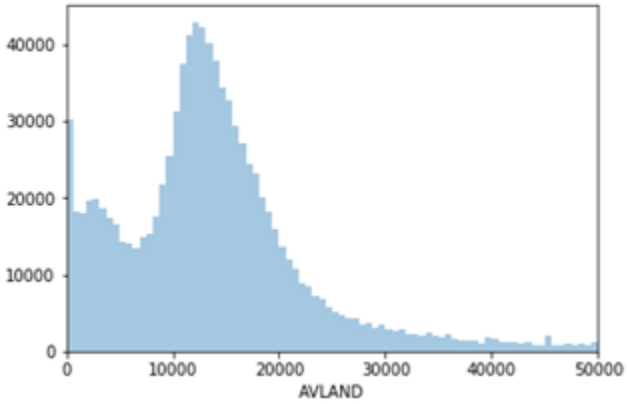
As seen from the table, apart from the missing values, there is a large amount of data with 0 values, which is obviously abnormal. Thus, we will regard them as missing values and fill in those values in the next step. Important parts of process are shown below:

- The property data includes geographical information:
  - ZIP the zip code for the property.
  - BBLE is the concatenation of BORO, BLOCK, LOT, EASEMENT and it is unique for each record.
  - BLOCK shows valid block ranges by BORO.
  - LOT is the unique number of the property within BORO/BLOCK.
  - EASEMENT is used to describe easement.
- Each property's physical information:
  - LTFRONT and LTDEPTH: lot frontage and depth
  - BLDFRONT and BLDDEPTH: building frontage and depth
- Regulatory information:
  - TAXCLASS: Tax class
  - BLDCLASS: Building class. There is a direct correlation between the Building Class and the 1st position of the Tax Class.
  - OWNER: property owner's name
- Core variables in fraud detection (the distribution plot is attached):

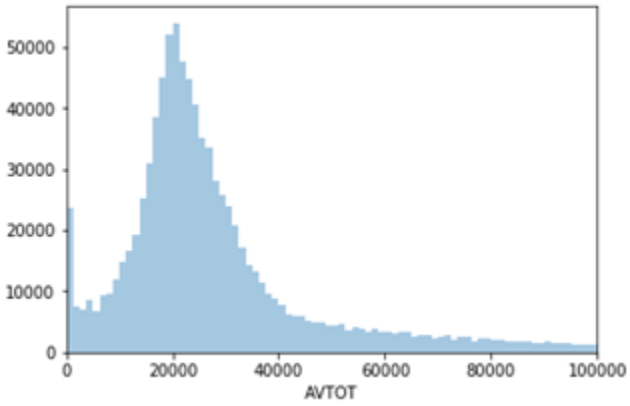
- FULLVAL: total market value of the property.



- AVLAND: The total land area.



- AVTOT: Assessed value of the property.



More details of each variable can be seen in the Appendix (Data Quality Report)

Data Cleaning Process

We need to use 9 fields that contain missing values (or 0 values) to create variables. We fill in the missing values in the following steps.

ZIP:

2.51% of the records do not contain a ZIP in the dataset.

We sort all the records by BBLE and use the ZIP from the previous property to fill in the missing values. BBLE is the combination of BORO, BLOCK and LOT information of the property. Properties with similar BBLE (similar BLOCK) are located close to each other so that they have the same ZIP.

LTFRONT & LTDEPTH:

For entries that have either one of LTFRONT/LTDEPTH, we take the average ratio, and derive the missing value from that, since they are highly correlated (Pearson correlation of 0.486).

For the remaining entries, we look at the mean LTFRONT/LTDEPTH values grouped by BOROUGH and TAXCLASS.

BLDFRONT & BLDDEPTH:

For entries that have either one of BLDFRONT/BLDDEPTH, we take the average ratio, and derive the missing value from that, since they are highly correlated (Pearson correlation of 0.60).

For the remaining entries, we look at the mean BLDFRONT/BLDDEPTH values grouped by BOROUGH and TAXCLASS.

STORIES:

To fill in the missing values for field STORIES, we assume that the average price per square feet is similar among properties with the same ZIP and same TAXCLASS. (There is a direct correlation between the Building Class and the 1st position of the Tax Class.)

Therefore, we group all the records by ZIP and then TAXCLASS and calculate the average STORIES (not include missing value) for each group.

Then, we replace the missing value by the average STORIES of its group.

FULLVAL:

To fill in the missing values for field FULLVAL, we assume that the average price per square feet is similar among properties with the same ZIP.

Therefore, we first calculate the average price per square feet for all the records that contains FULLVAL:

average price per square feet = FULLVAL/(STORIES* BLDFRONT * BLDDEPTH)

Second, we group all the records by ZIP and calculate the mode of average price per square feet within one ZIP. Let us call it ‘mode price per square feet of ZIP’.

Then, we use ‘mode price per square feet of ZIP’ to calculate the FULLVAL for the properties with missing value:

FULLVAL = (STORIES * BLDFRONT * BLDDEPTH) * mode price per square feet of ZIP

AVLAND & AVTOT:

Since this is an assessed tax value, it is grouped by BOROUGH, TAXCLASS, lot size and building size (for AVTOT). The average AVLAND/AVTOT values are taken.

Variables Used

Variables Transformation

1) Create 3 sizes: LOTAREA, BLDAREA and BLDVOL

There are three target values: FULLVAL, AVLAND and AVTOT, which stands for total market value of the land, assessed land value and assessed total value respectively. To normalize these three values, we will create 3 sizes using existing variables:

· Lot area: LOTAREA = LTFRONT * LTDEPTH

· Building area: BLDAREA = BLDFRONT * BLDDEPTH

· Building volume: BLDVOL = BLDAREA * STORIES

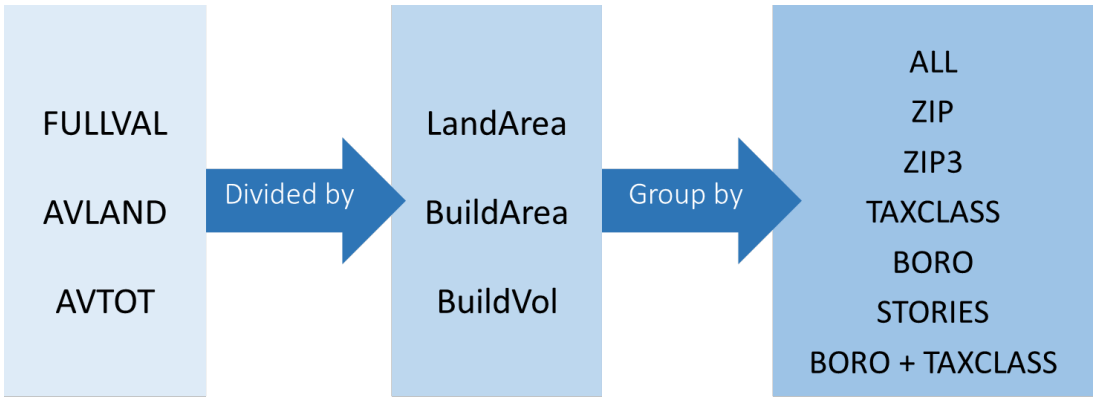
2) Calculate 9 core variables

Intuitively, land value usually has a positive relationship with the area of the lot, building and the volume of building. In this way we then normalize FULLVAL, AVALAND and AVTOT by these three sizes (e.g. FULLVAL/ LOTAREA) and derive each land value per unit. This step creates another 3 * 3, which is 9 variables.

3) Calculate another 54 variables

Finally, we take advantage of several entities in the data to further scale the 9 core variables. The entities include ZIP, ZIP3 (first 3 digits of the ZIP), TAXCLASS, BORO and STORIES. The basic intuition here is that within the same entity group mentioned above indicating geographical location, property tax class and building structure, the difference among land values is not significant. After each of the record is divided by certain entity group average, the deviation of value from the average level could be captured. What is worth mentioning in the preprocessing step is that we have transformed STORIES into 10 bins instead of the nominal value.

To be more specific, first, we calculate averages grouped by each of 5 entities and also the combination of BORO and TAXCLASS. By considering location and tax class at the same time, we can get more reasonable averaged land values. From the perspective of feasibility, there are 11 unique tax classes and 5 unique borough codes. This total group number 55 helps to nicely divide the target values. Secondly, divide each of the 9 core variables by the 6 scale factors from these groupings.



Thus, there are 3 * 3 * 7 = 63 new variables in all calculated through this step to be put in our unsupervised model later.

Standardization and Dimension Reduction (PCA)

Variable Standardization

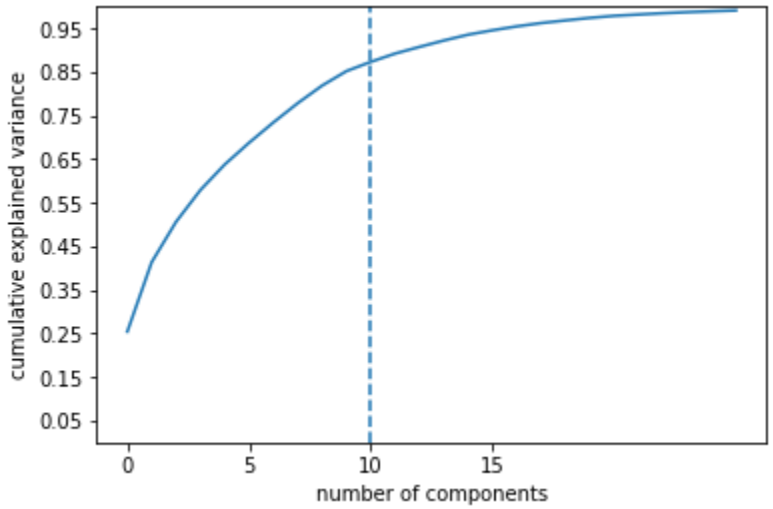
As our 63 variables are in different scales, we need to standardize them in order to make a better comparison. Here we use the z-scaling method, which ensures linearly transformed data values having a mean of zero and a standard deviation of 1. To standardize a variable, subtract the mean and divide by the standard deviation. The formula used is shown below:

$$z_i = \frac{x_i - \mu_i}{\sigma_i}$$

Where x_i is the variable, μ_i is the mean, σ_i is the standard deviation.

Principal Component Analysis

As we calculate 54 variables by using the 9 core variables, it is possible that variables are correlated more or less. Therefore, we use PCA to remove these correlations and achieve a reduction in dimensions. Principal Component Analysis (PCA) is a tool used to reduce a large set of variables to a small set that still contains most of the information in the large set. The first principal component accounts for as much of the variability in the data as possible, and each succeeding component accounts for as much of the remaining variability as possible. Generally speaking, the number of components we choose should can explain for 80% - 90% variance of data. So we next look at the cumulative explained variance ratio as a function of the number of components:



	Percentage of explained variance
PC1	25.44%
PC2	15.94%
PC3	9.26%
PC4	7.29%
PC5	5.87%
PC6	4.99%
PC7	4.67%
PC8	4.42%
PC9	4.03%
PC10	3.32%

To ensure more than 85% of the variance is explained, we choose to include the first 10 components, which can account for 85.23% of the variance. PC 1 can explain up to 25.44% of the variation, and PC 2 has 15.94%.

Principal Component Standardization

Finally, the principal components chosen have to be scaled using Z-scale method again to ensure those 10 components are comparable before being put into the model. Thus, the transformed shape of PC data contains 1048575 records and 10 columns.

Model Algorithms

Mahalanobis Distance

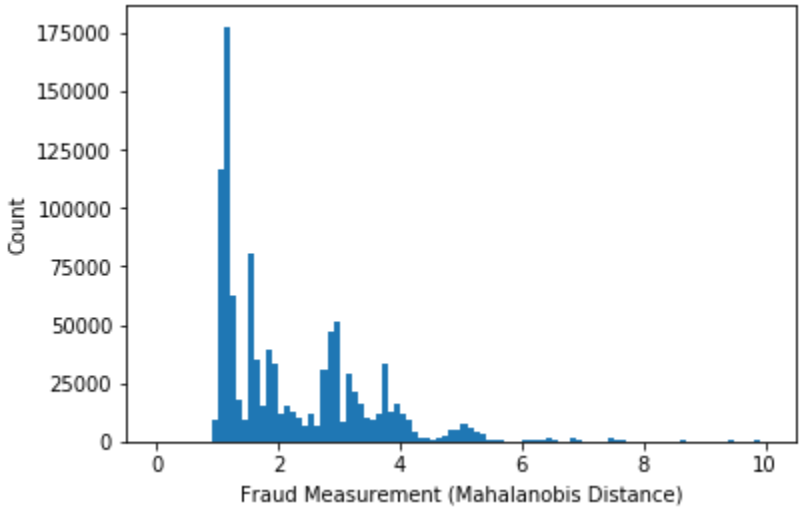
The first way that we use to determine outliers is the Mahalanobis distance method. The mahalanobis distance D is a measure of how far away any given point p is from the center C. Instead of the euclidean distance, the mahalanobis distance gives the number of standard deviations a point is from the center with the spread of each axis put into consideration.

Given that the dataset has already been broken down into each principal component, all that is required to obtain the mahalanobis distance is to standardize the number and measure the deviations in euclidean distance.

For each data point p, which has n principal components, each principal component F_i has been scaled down to a standardized z-score. The z-scores have been combined using the root sum squared. The method used for the mahalanobis distance is shown below:

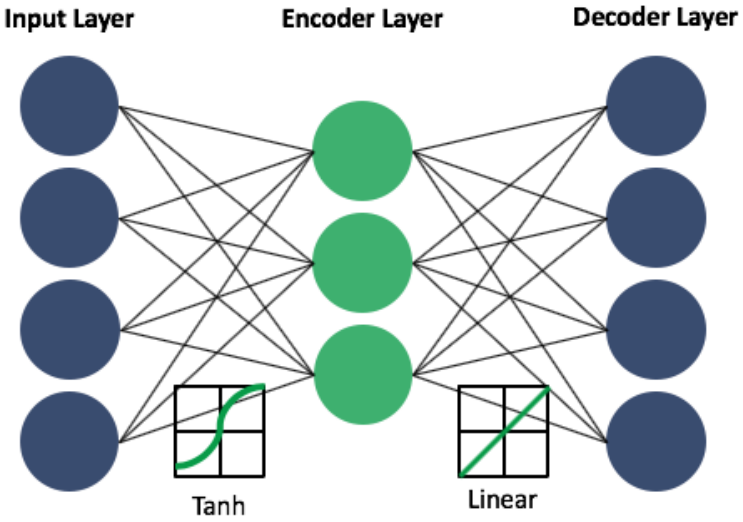
$$D(p) = \sqrt{\sum_{i}^{n} \left(\frac{F_i - \mu_i}{\sigma_i}\right)^2}$$

where n is the number of selected principal components and F_i is each principal component. The following plot shows the distribution of the mahalanobis distance representing fraud scores that have been assigned to each entry. A smaller score represents lower likelihood of fraud and vice versa.



Autoencoder

An autoencoder learns to compress data from the input layer into a short code, and then decompresses that code into something that closely matches the original data. In this project, we use the simplest form of an autoencoder. It has an input layer, an output layer (decoder layer) and one hidden layer (encoder layer).



Input layer

The first layer is the input layer. The input dimension is 10, which is the number of principal components we have chosen from PCA.

Encoder layer

The autoencoder consists of two parts, the encoder and the decoder. The encoding dimension is that same as the input dimension divided by the compression factor. In this case, we set the compression factor as 1.1. The reason we choose a relatively small compression factor is that we have already performed a dimensionality reduction in PCA. We have also tried larger compression factors, but the results however are not as ideal as 1.1.

We use the **tanh activation function** in the encoder layer. There are three reasons. Firstly, we need a non-linear activation function in order to differentiate the autoencoder model from the previous model and learn the non-linear relationships. Secondly, our labels (the output from PCA) contain both positive and negative numbers. Therefore, we want the output from the encoder layer to also contain positive and negative numbers since the output range of tanh activation function is (-1, 1). Thirdly, the tanh function is chosen since it is the most simple activation function that satisfies all the previous requirements.

Decoder layer

The output layer has the same **number of nodes** as the input layer with the purpose of reconstructing its own inputs.

We use a **linear activation function** in the decode layer because we want the range of output to be from negative infinity to positive infinity (the same as PCA output). And the linear activation function is the simplest.

Optimizer

We tried several optimizers and adadelta performs the best.

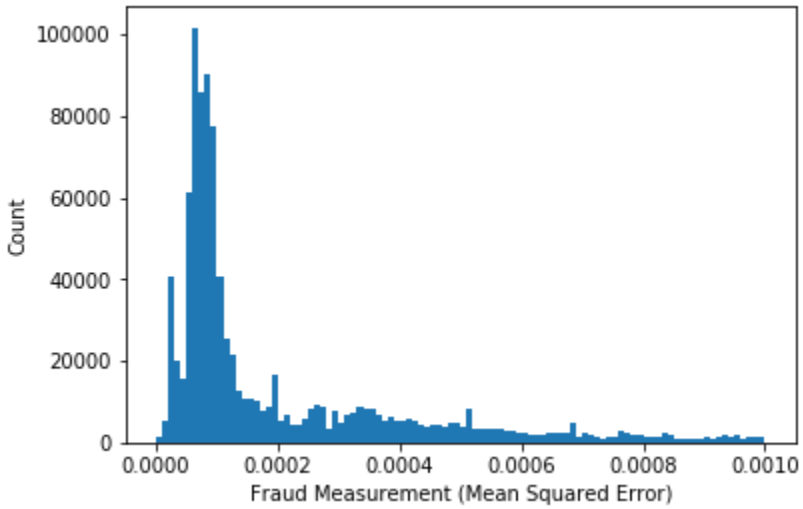
Loss function

We use the mean square error (MSE) as the loss function. Since we only want to consider regression loss functions and the MSE function performs the best among all the regression loss functions.

Fraud score calculation and distribution

After using the autoencoder to reproduce a matrix of PCA, we compare the input matrix and the output matrix and calculate the difference. The metrics we tried include mean absolute error, mean square error and root sum squared with different values for n.

After plotting the distributions of all errors, we find the distribution of MSE is the best. The errors of most records are right-skewed, meaning that fraud scores of most records are very low. Thus, the fraud records are easier to detect. Therefore, we use MSE as our fraud score for the autoencoder. We also compared different fraud rankings using three metrics. The rankings are the same for MAE, MSE and root sum squared.



Combined Fraud Scores

After both fraud scores from the autoencoder and mahalanobis distance have been assigned to each entry in the data, a ranked value had been assigned for both fraud scores to each entry. Two values (*MahalRank* for Mahalanobis distance and *AERank* for AutoEncoder) have been assigned to represent a fraud score where a higher rank number would represent a higher likelihood of fraud and vice versa.

For a comparison of performance of both methods, the data has been sorted in ranked order and the amount of overlapping entries within each portion of entries. For the top 1% (10,000 entries) of the highest fraud score entries, there is a 49% overlap between the two fraud scores. On the other hand, for the bottom 1% of the which represent the lowest fraud scores, there is only a 9% overlap between the two fraud scores.

In conclusion, since both fraud scores have been given the same weight, the combined score used is simply a mean of both fraud ranks. By using the mean of two ranks, it gives no bias to either ranking system, and at the same time, does not provide a higher than necessary chance of false positives and a lower than necessary chance of false negatives.

Results

	RECORD	LTFRONT	LTDEPTH	STORIES	FULLVAL	M's Distance Rank	Auto Encoder Rank	Combined Rank
1	632817	100	100	2.5	330000	1	1	1
2	126405	159	159	2	817000	2	2	2
3	126400	131.2	131.2	6	175778	3	3	3
4	138830	100	100	3	800000	4	4	4
5	585119	100	100	2	340000	5	5	5
6	108897	100	100	2	700000	6	6	6
7	85887	101	101	31	281677	7	7	7
8	918306	100	100	2	370000	8	8	8
9	934794	100	100	2.7	419000	9	10	9.5
10	934781	131.2	131.2	16	158864	10	11	10.5

1. **111-15 212 STREET**
FULLVAL is cheaper than average (372000) with same ZIP, same STORIES and BLOCK

2. **141-37 13 AVENUE**
FULLVAL is cheaper than average (880000) with same ZIP, same STORIES and BLOCK

3. **350 EAST 30 STREET**
Big lot area

4. **25-88 48 STREET**
FULLVAL is more expensive than average (730000) with same ZIP, same STORIES and BLOCK

5. **81-59 102 AVENUE**
FULLVAL is cheaper than average (446000) with same ZIP and same STORIES

6. 2059 62 STREET
FULLVAL cheaper than average (724068), taxclass C3

7. 30 WEST 61 STREET
31 Story building, BLDVOL does not match

8. 336 Underhill Ave Bronx NY
LTFRONT is a lot smaller than listed 100

9. 221-15 FAIRBURY AVENUE

FULLVAL is cheaper than average (435000) with same ZIP and same STORIES

10. 41 WEST 72 STREET

FULLVAL is cheaper than average (200000) with same STADDR and same STORIES

Conclusions

In this project, we went through data exploratory analysis, data cleaning and feature selection and feature engineering before building the algorithms. Then we use the two algorithms to evaluate the fraud level of all records and combine the results of the two algorithms. In the end, we manually look at the suspicious records and analyze them in detail.

The fraud score distributions of the two algorithms are seriously right-skewed, which means most of the records are not fraudulent. Meanwhile, most of the top ten records we selected are unusual and are not just data errors.

We still have future work if we have more time. First, we do not use variables with a lot of missing values, such as EXTOT, in this project. We would like to explore whether these variables have impact on fraud scores. Second, we would like to explore the relationship between 'OWNER' and fraud score. Third, currently we are using 'tanh' as encode layer activation function and 'linear' as decode layer function. We would like to try more advanced activation functions such as LeakyRelu. Also, we would like to try more advanced algorithms such as convolutional autoencoder.

Appendix: Data Quality Report

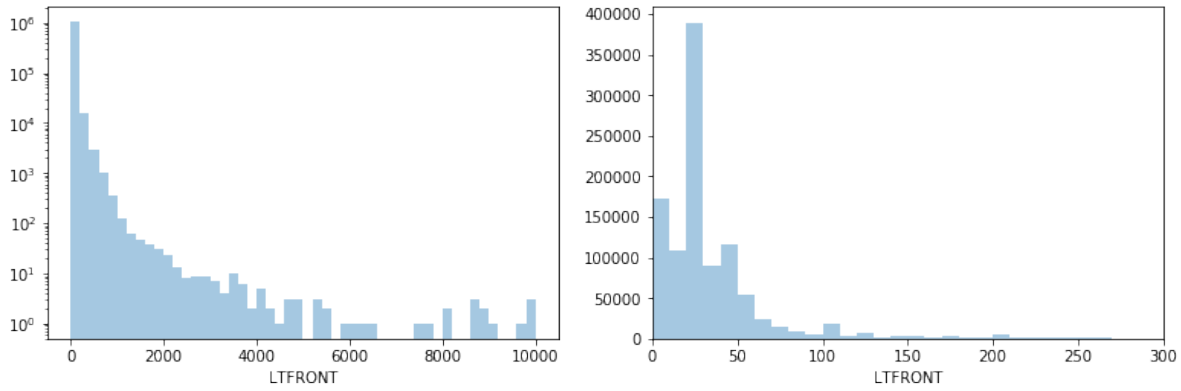
Continuous Variables

Variable	Dtype	Description
LTFRONT	Int64	Lot Frontage in feet. No null values. Zero-value represents a large proportion.

Top 5 frequent values and counts:

```
0      168867
20     134447
25     116301
40      81802
18      40188
```

The distribution plot is shown below. The y-axis in the left plot has been applied with log-transformation. The right one shows a closer look at LTFRONT from 0 to 300 feet.

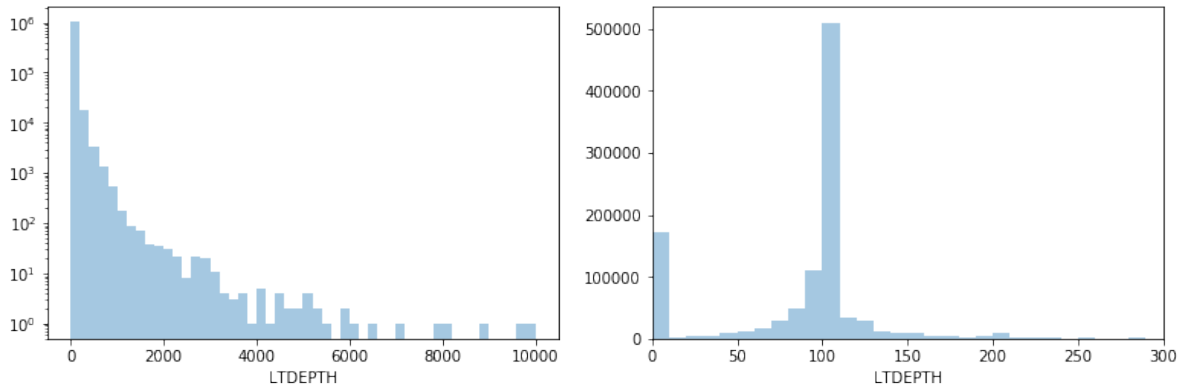


Variable	Dtype	Description
LTDEPTH	Int64	Lot Depth in feet. No null values. Zero-value represents a large proportion.

Top 5 frequent values and counts:

```
100     457583
0       169888
95       31022
90       19941
80       16414
```

The distribution plot is shown below. The y-axis in the left plot has been applied with log-transformation. The right one shows a closer look at LTDEPTH from 0 to 300 feet.

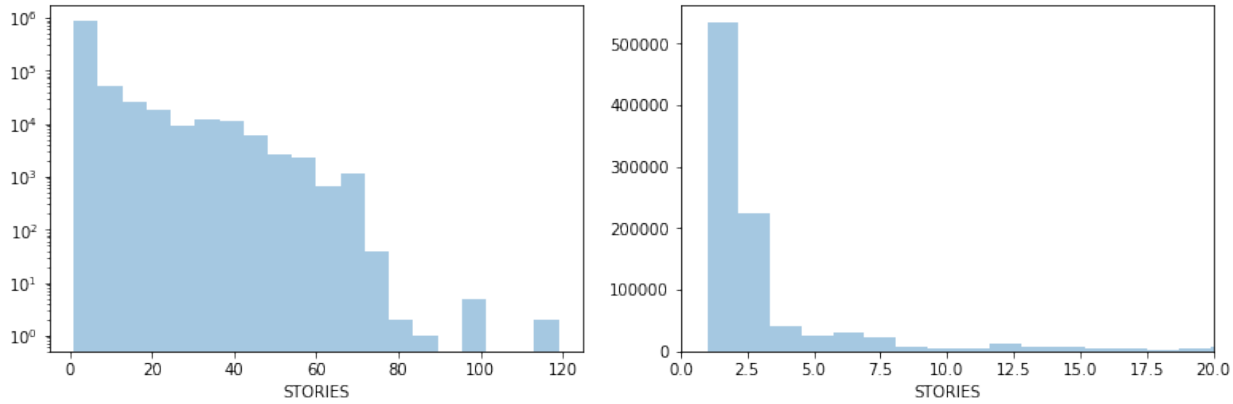


Variable	Dtype	Description
STORIES	float64	The number of stories for the building (# of Floors). Involve null values.

Top 5 frequent values and counts:

```
2.0    403318
3.0    128493
1.0     93606
2.5     81304
4.0     38337
```

The distribution plot is shown below. It excludes null values. The y-axis in the left plot has been applied with log-transformation with 20 bins. The right one shows a closer look at STORIES from 0 to 20 floors.

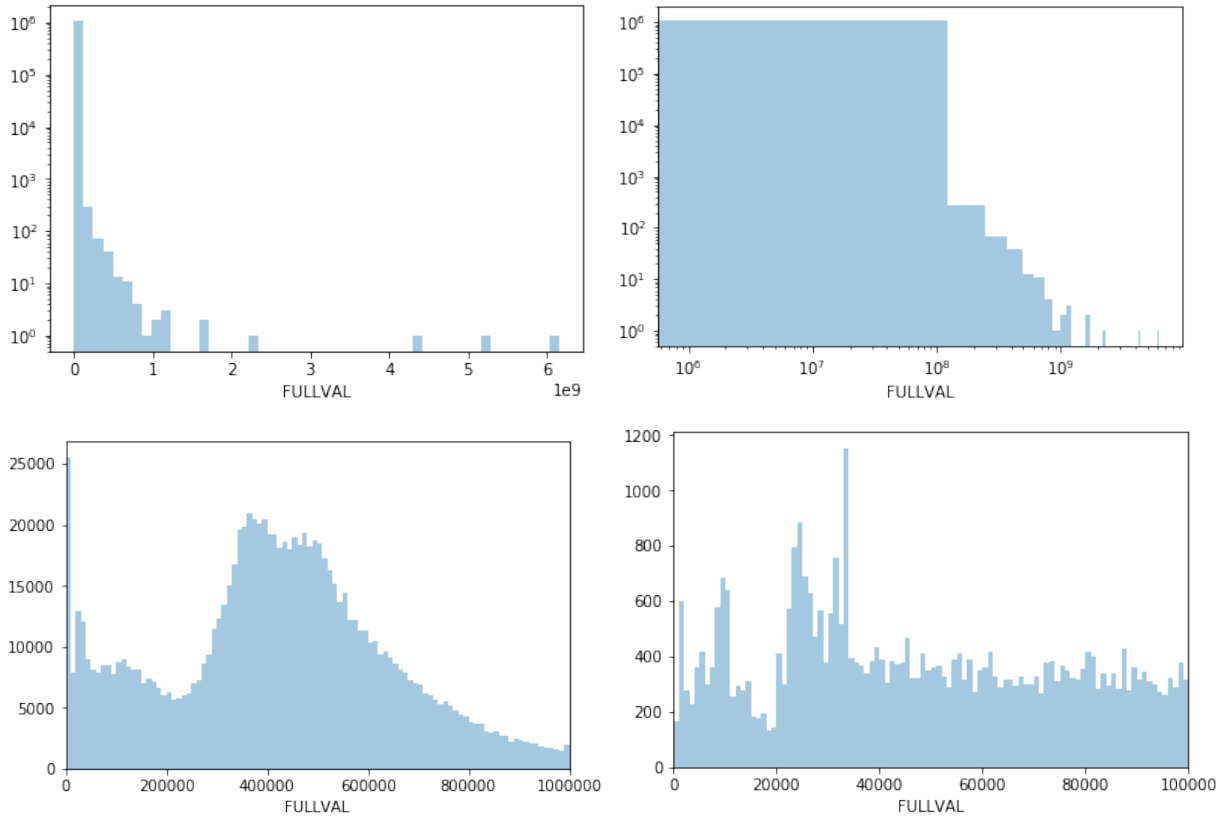


Variable	Dtype	Description
FULLVAL	Int64	If not zero, total market value. No null values

Top 5 frequent values and counts:

```
0         12762
502000     2751
366000     2260
397000     2189
472000     2179
```

The distribution plot is shown below. The y-axis in the left plot has been applied with log-transformation. The upper right one also includes log-transformed x-axis. The third plot takes a closer look at values lower than 1,000,000. The last plot adds another limitation which is FULLVAL of non-zero values based on the third plot.

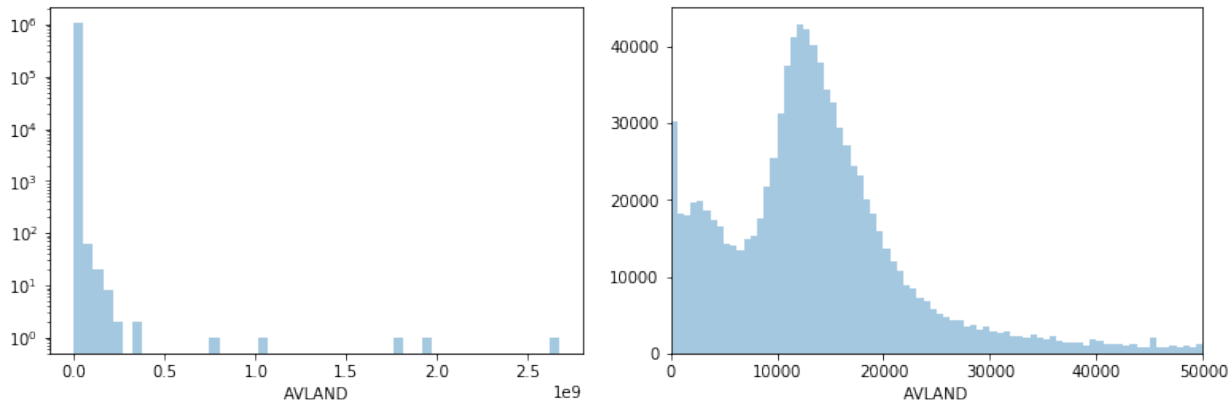


Variable	Dtype	Description
AVLAND	Int64	Assessed Land Value. No null values.

Top 5 frequent values and counts:

```
0        12764
45000     1225
90000      995
3045       873
22500      825
```

The distribution plot is shown below. The y-axis in the left plot has been applied with log-transformation. The right one plot takes a closer look at values lower than 50,000.

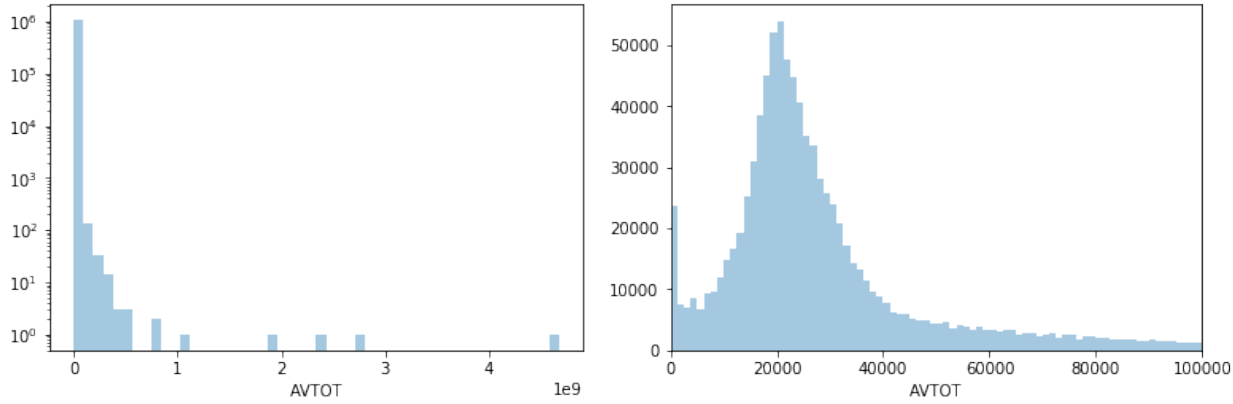


Variable	Dtype	Description
AVTOT	Int64	Assessed Total Value. No null values.

Top 5 frequent values and counts:

```
0        12762
16588     3111
17914     2953
18973     2447
19780     2435
```

The distribution plot is shown below. The y-axis in the left plot has been applied with log-transformation. The right one plot takes a closer look at values lower than 100,000.

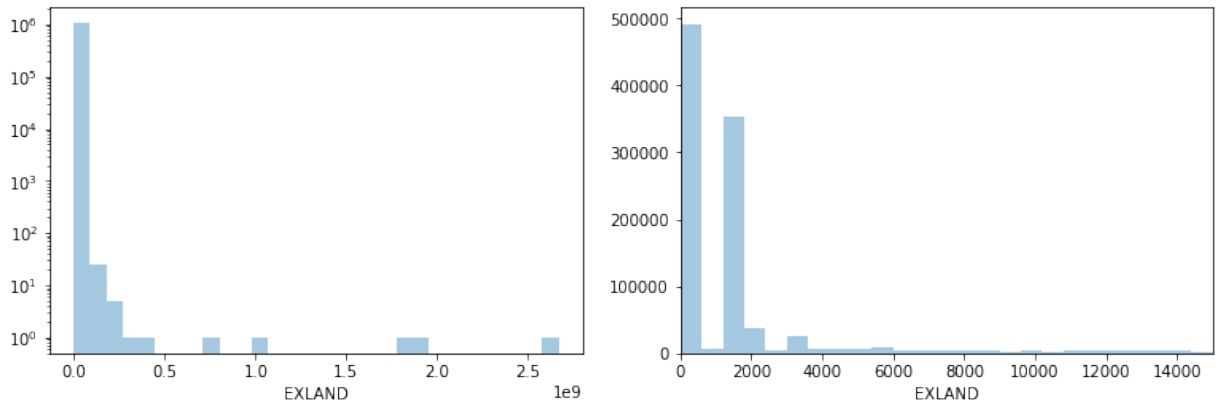


Variable	Dtype	Description
EXLAND	Int64	Exempt Land Value. No null values.

Top 5 frequent values and counts:

```
0          484224
1620       346604
2090        31111
3240        21181
5760         3365
```

The distribution plot is shown below. The y-axis in the left plot has been applied with log-transformation. The right one plot takes a closer look at values lower than 15,000 with bins of 25.

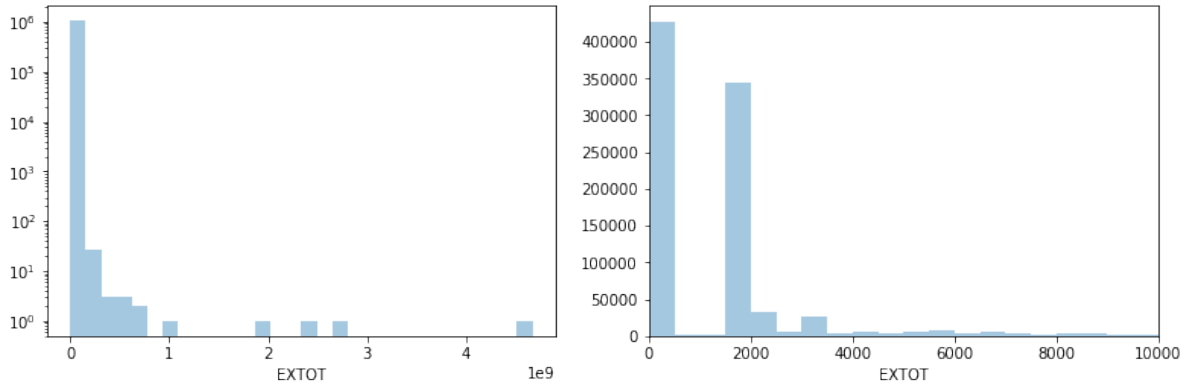


Variable	Dtype	Description
EXTOT	Int64	Exempt Total Value. No null values.

Top 5 frequent values and counts:

```
0       425999
1620    344273
2090     30068
3240     21436
5760      3353
```

The distribution plot is shown below. The y-axis in the left plot has been applied with log-transformation. The right one plot takes a closer look at values lower than 10,000 with bins of 20.

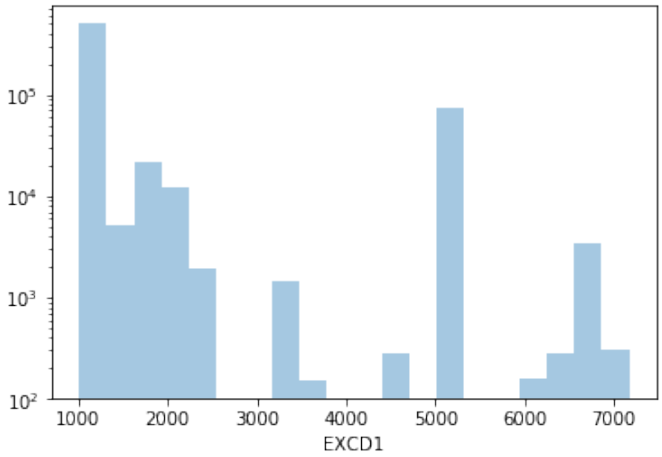


Variable	Dtype	Description
EXCD1	Int64	Involve null values.

Top 5 frequent values and counts:

```
1017.0    414222
1010.0     48322
1015.0     30849
5113.0     23842
1920.0     17594
```

The distribution plot is shown below. It excludes null values. The y-axis in the left plot has been applied with log-transformation with 20 bins in the x-axis.

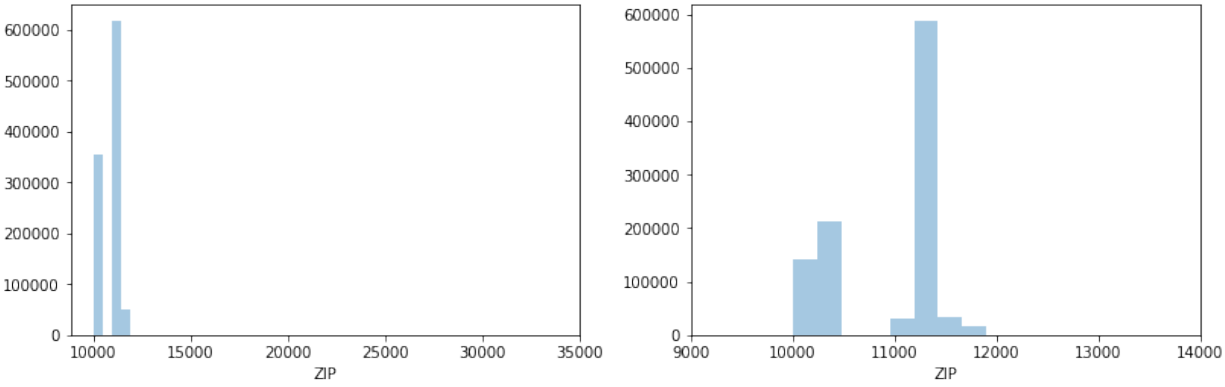


Variable	Dtype	Description
ZIP	floatt64	Zip code. Involve large null values.

Top 5 frequent values and counts:

```
10314.0    24605
11234.0    20001
10462.0    16905
10306.0    16576
11236.0    15678
```

The distribution plot is shown below. It excludes null values. The right plot takes a closer look at the ZIP from 10000 to 14000.

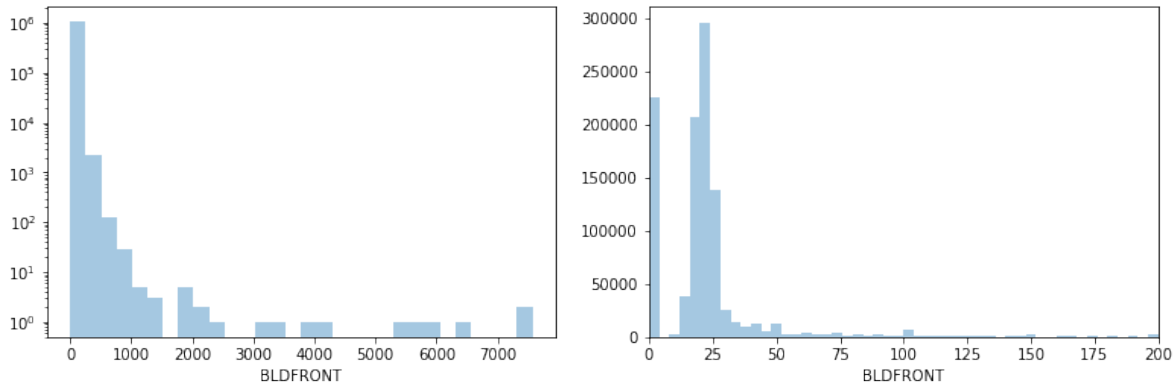


Variable	Dtype	Description
BLDFRONT	Int64	Building Frontage in feet. No null values.

Top 5 frequent values and counts:

```
0     224661
20    193812
18     76808
16     73671
25     61770
```

The distribution plot is shown below. The y-axis in the left plot has been applied with log-transformation. The right one plot takes a closer look at values lower than 200 with bins of 100.

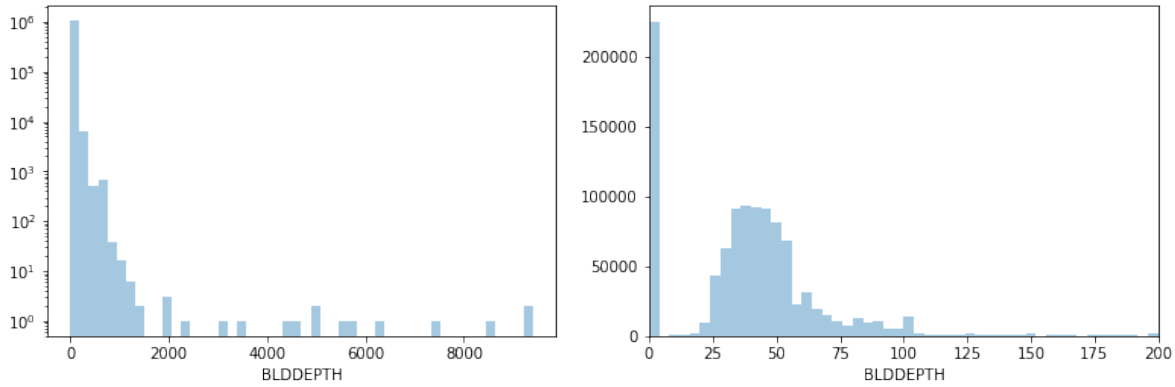


Variable	Dtype	Description
BLDDEPTH	Int64	Building Depth in feet. No null values.

Top 5 frequent values and counts:

```
0      224699
40      47185
50      44303
36      39514
45      38921
```

The distribution plot is shown below. The y-axis in the left plot has been applied with log-transformation. The right one plot takes a closer look at values lower than 200 with bins of 50.

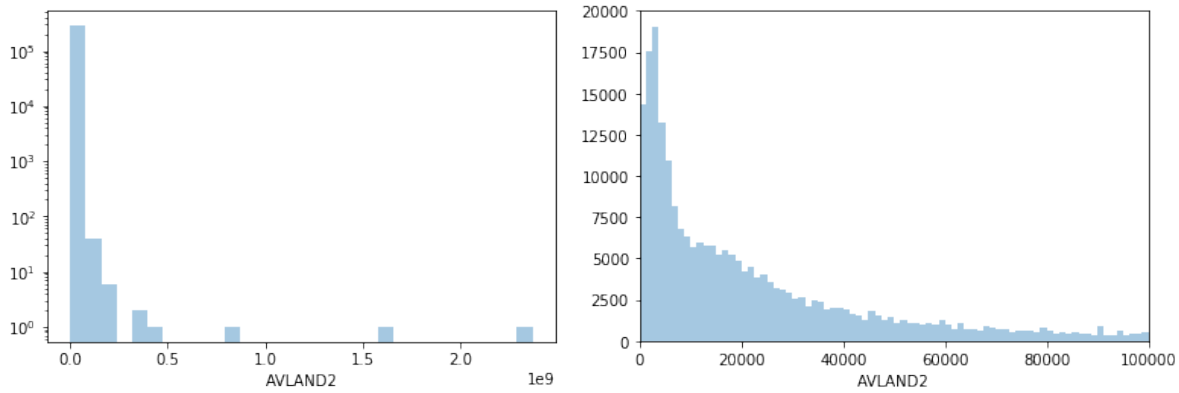


Variable	Dtype	Description
AVLAND2	float64	Assessed Land Value 2. Contains null values.

Top 5 frequent values and counts:

```
2408.0     767
2233.0     610
45000.0    596
750.0      547
90000.0    511
```

The distribution plot is shown below. It excludes null values. The right plot takes a closer look at the AVLAND2 smaller than 100,000.

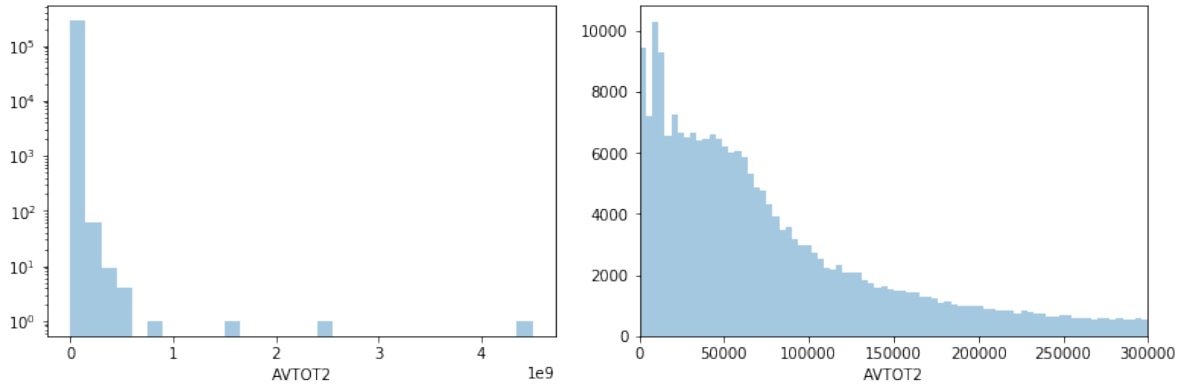


Variable	Dtype	Description
AVTOT2	float64	Assessed Total Value2. Contains null values.

Top 5 frequent values and counts:

```
750.0      656
9468.0     233
9104.0     232
9349.0     225
9687.0     215
```

The distribution plot is shown below. It excludes null values. The right plot takes a closer look at the AVTOT2 smaller than 300,000.

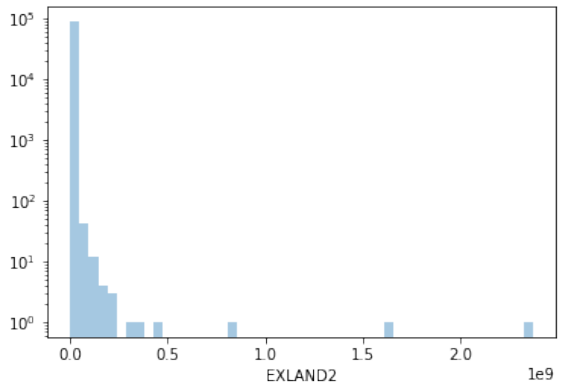


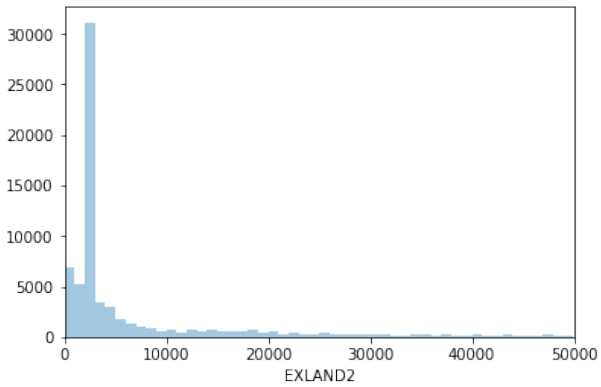
Variable	Dtype	Description
EXLAND2	Float64	Exempt Land Value 2. Contains null values.

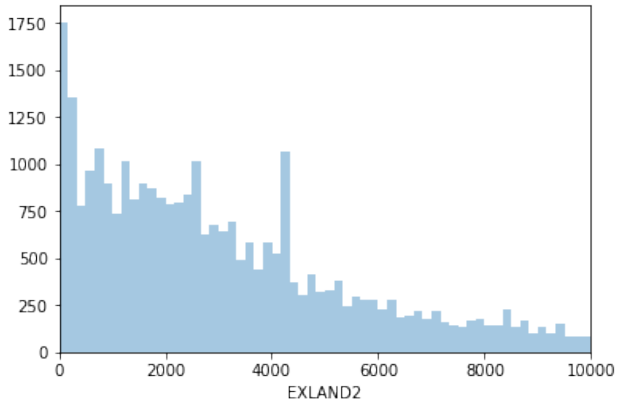
Top 5 frequent values and counts:

```
2090.0      26393
4180.0        734
2650.0        390
62640.0       192
126180.0      150
```

The distribution plot is shown below. It excludes null values. The right plot excludes the OUTLIER and takes a closer look at the EXLAND2 smaller than 50,000. The third plot excludes the OUTLIER 2090.0 and takes a closer look at the EXLAND2 smaller than 10,000.





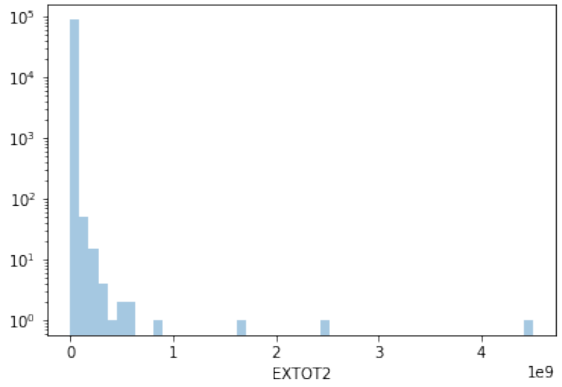


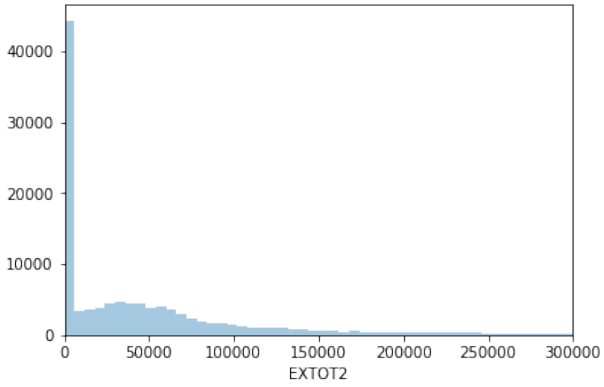
Variable	Dtype	Description
EXTOT2	float64	Exempt Total Value. Contains null values.

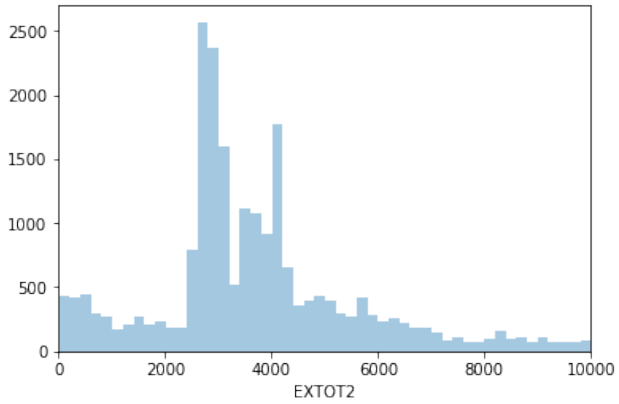
Top 5 frequent values and counts:

```
2090.0      26393
4180.0        734
2650.0        390
62640.0       192
126180.0      150
```

The distribution plot is shown below. It excludes null values. The right plot excludes the OUTLIER 2090.0 and takes a closer look at the EXTOT2 smaller than 300,000. The third plot excludes the OUTLIER 2090.0 and takes a closer look at the EXTOT2 smaller than 10,000.





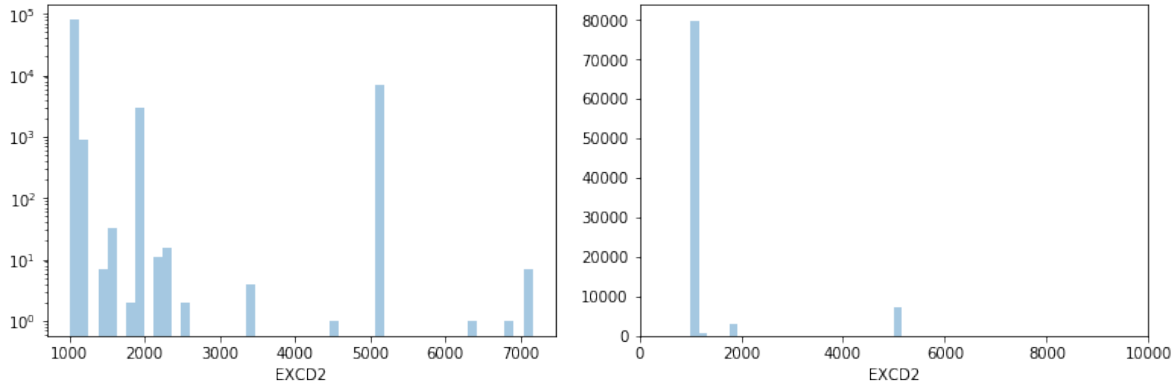


Variable	Dtype	Description
EXCD2	float64	Contains null values.

Top 5 frequent values and counts:

```
1017.0    64223
1015.0    12038
5112.0     6867
1019.0     3034
1920.0     2961
```

The distribution plot is shown below. It excludes null values. The right plot takes a closer look at the EXCD2 smaller than 10,000.



Categorical variables

Variable	Dtype	Description
BBLE	object	Concatenation of other variables.

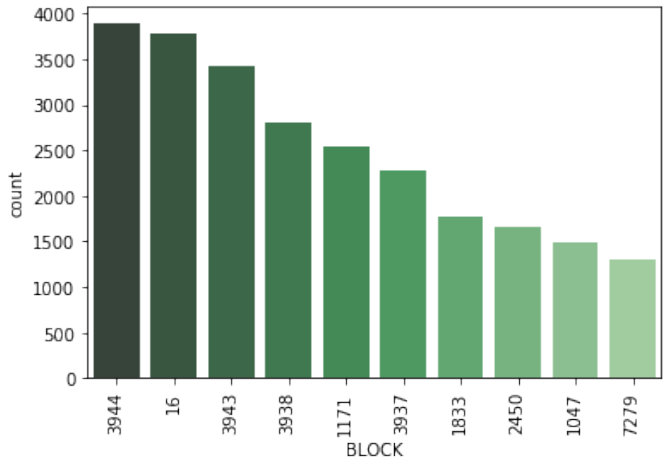
Top 10 frequent values and counts:

```
2027120134    1
1012621010    1
2050550044    1
3002340031    1
3039240059    1
4135130007    1
3007571205    1
5054590087    1
2037620049    1
4136990026    1
```

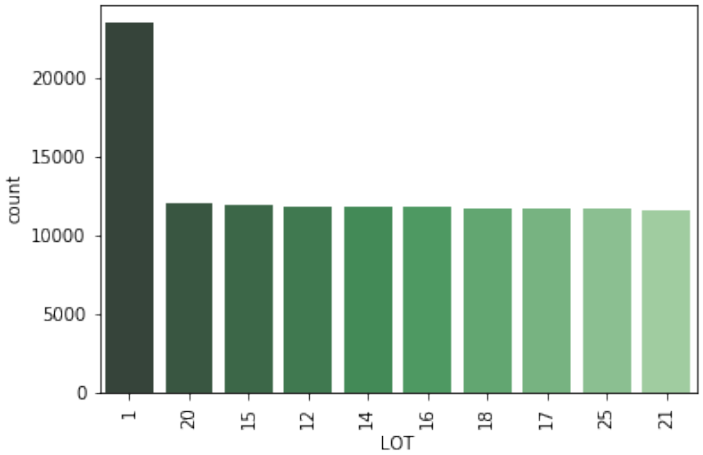
Thus, each BBLE remains unique in the data.

Variable	Dtype	Description
BLOCK	object	Blocks.

The distribution plot of top 10 frequent BLOCKs is shown below.

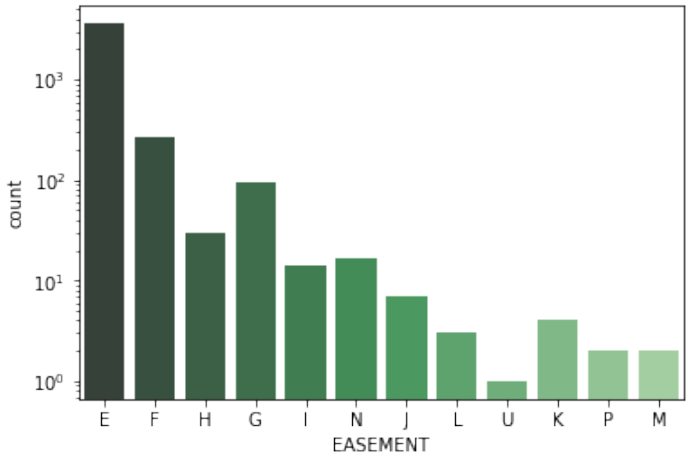


Variable	Dtype	Description
LOT	object	Lots.



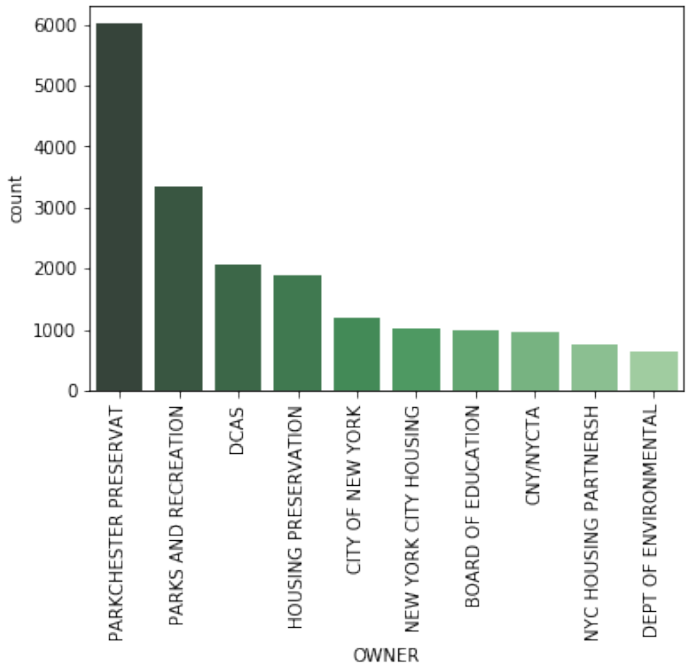
Variable	Dtype	Description
EASEMENT	object	Is a field that is used to describe easement.

The distribution plot is shown below. The y-axis in the plot has been applied with log-transformation.



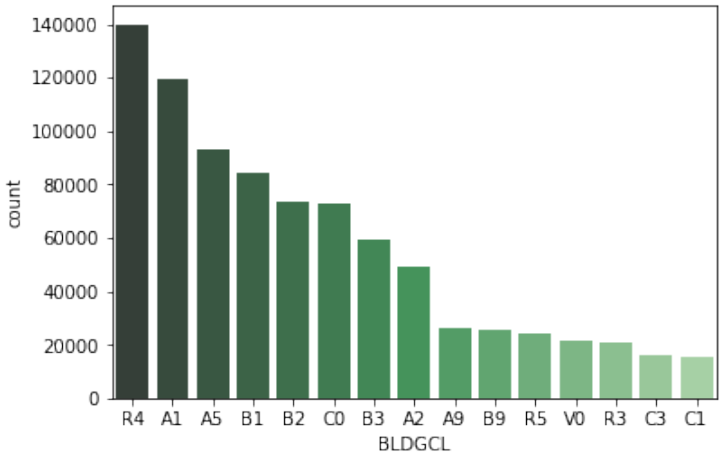
Variable	Dtype	Description
OWNER	object	The owner's name

The distribution plot is shown below. The plot only shows the top 10 frequent OWNER.



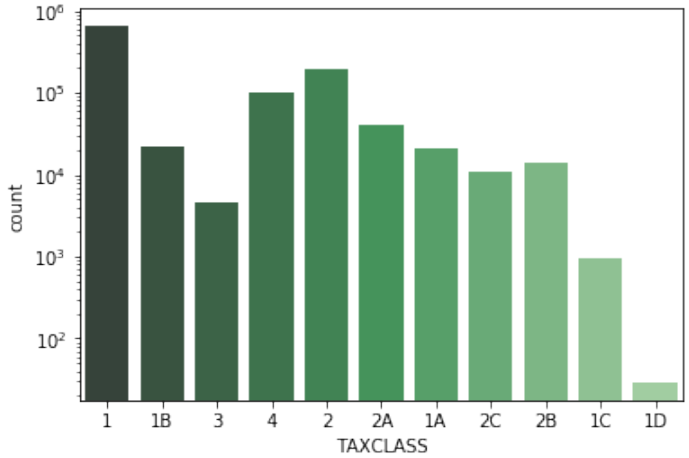
Variable	Dtype	Description
BLDGCL	character	Building class.

The distribution plot is shown below. The plot shows the top 15 frequent BLDGCLs.



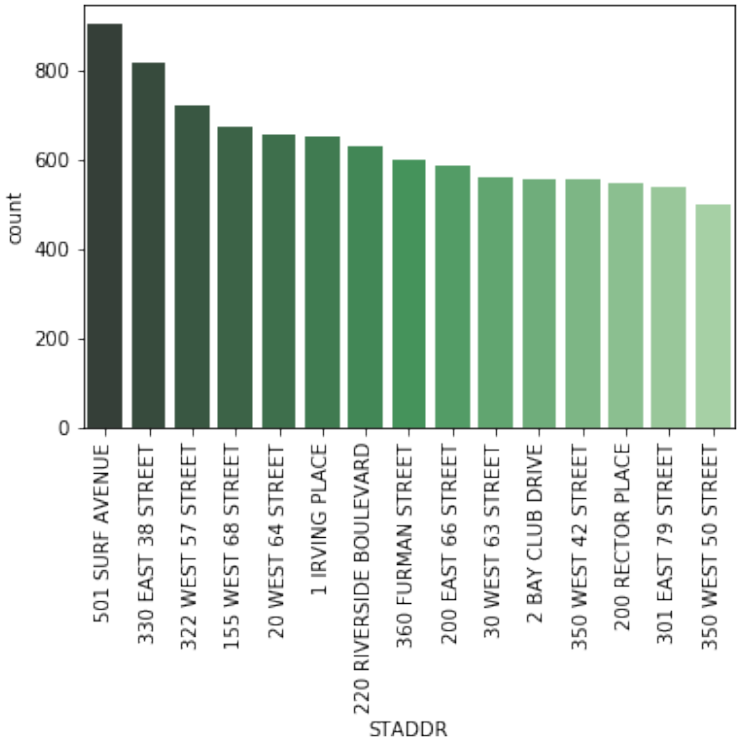
Variable	Dtype	Description
TAXCLASS	object	Current Property Tax Class Code. There is a direct correlation between the Building Class and the 1st position of the Tax Class

The distribution plot is shown below. The y-axis in the plot has been applied with log-transformation.



Variable	Dtype	Description
STADDR	character	Street Address

The distribution plot is shown below. The plot shows the top 15 frequent STADDRs.



Variable	Dtype	Description
EXMPTCL	character	Exempt Class used for fully exempt properties only.

The distribution plot is shown below. The y-axis in the plot has been applied with log-transformation.