



Predicting Earthquake Damage

IBM Advanced Data Science Capstone

Cheng Seng Tan

It is evident that earthquakes do not kill people; rather the weak structures kill the people. Hence, if we stay in a well-constructed earthquake resistant building and the surroundings, we can live safely even in an earthquake prone area.

Mostly, old, non-engineered, adobe and masonry buildings collapsed and/or were severely damaged by the earthquake. In addition, some engineered buildings also damaged or collapsed due to poor workmanship and quality of construction materials.

Impacts of the 2015 Gorkha Earthquake: Lessons Learnt from Nepal
By Shiva Subedi and Meen Bahadur Poudyal Chhetri

Use Case

Question

Given the features of a building, can I predict the level of damage of a building when an earthquake strikes?

Challenge

Machine-learn from information of damaged buildings in the Gorkha earthquake of 2015 to create a predictive model.

Dataset

The dataset was collected through surveys by the Central Bureau of Statistics that work under the National Planning Commission Secretariat of Nepal. This survey is one of the largest post-disaster datasets ever collected, containing valuable information on earthquake impacts, household conditions, and socio-economic-demographic statistics.

Source : https://www.kaggle.com/mullerismail/richters-predictor-modeling-earthquake-damage

260,601

Buildings

Identified by building_id

38

Features

information on the buildings' structure and their legal ownership e.g. geographic region, number of floors, age, land surface condition, foundation type, roof type, superstructure type (adobe mud, etc

3

Damage Grade

- 1 represents low damage
- 2 represents medium amount of damage
- 3 represents almost complete destruction

Solution to Use Case

Use a Logistics Regression Model

To predict the level of damage in an earthquake with accuracy.

55%

Architectural Choices

Raw data

CSV Files from Kaggle site

Data **Repository**

IBM Cloud object storage and Github repository.

Data Exploration, Feature Engineering and Modeling

Jupyter Notebook with Python 3.6 Spark.

Python libraries used

- NumPy
- Pandas
- Seaborn
- Matlplotlib
- PySpark MLLib- Logistic Regression, Random Forest & Forward Feed Neural Network

Data Quality Assessment

Building_id	Feature 1	Feature 2	….	Feature 38
802906	6	487	….	1
28830	8	900	…..	0
94947	21	363	…..	1
590882	22	418	….	6
…	….	….	….	….

(260,601, 39)

Building_id	Damage _grade
802906	3
28830	2
94947	3
590882	1
…	….

(260,601, 2)

Building_id	Feature 1	Feature 2	….	Feature 38	Damage _grade
802906	6	487	….	1	3
28830	8	900	…..	0	2
94947	21	363	…..	1	3
590882	22	418	….	6	1
…	….	….	….	….	….

(260,601, 40)

Merge Files

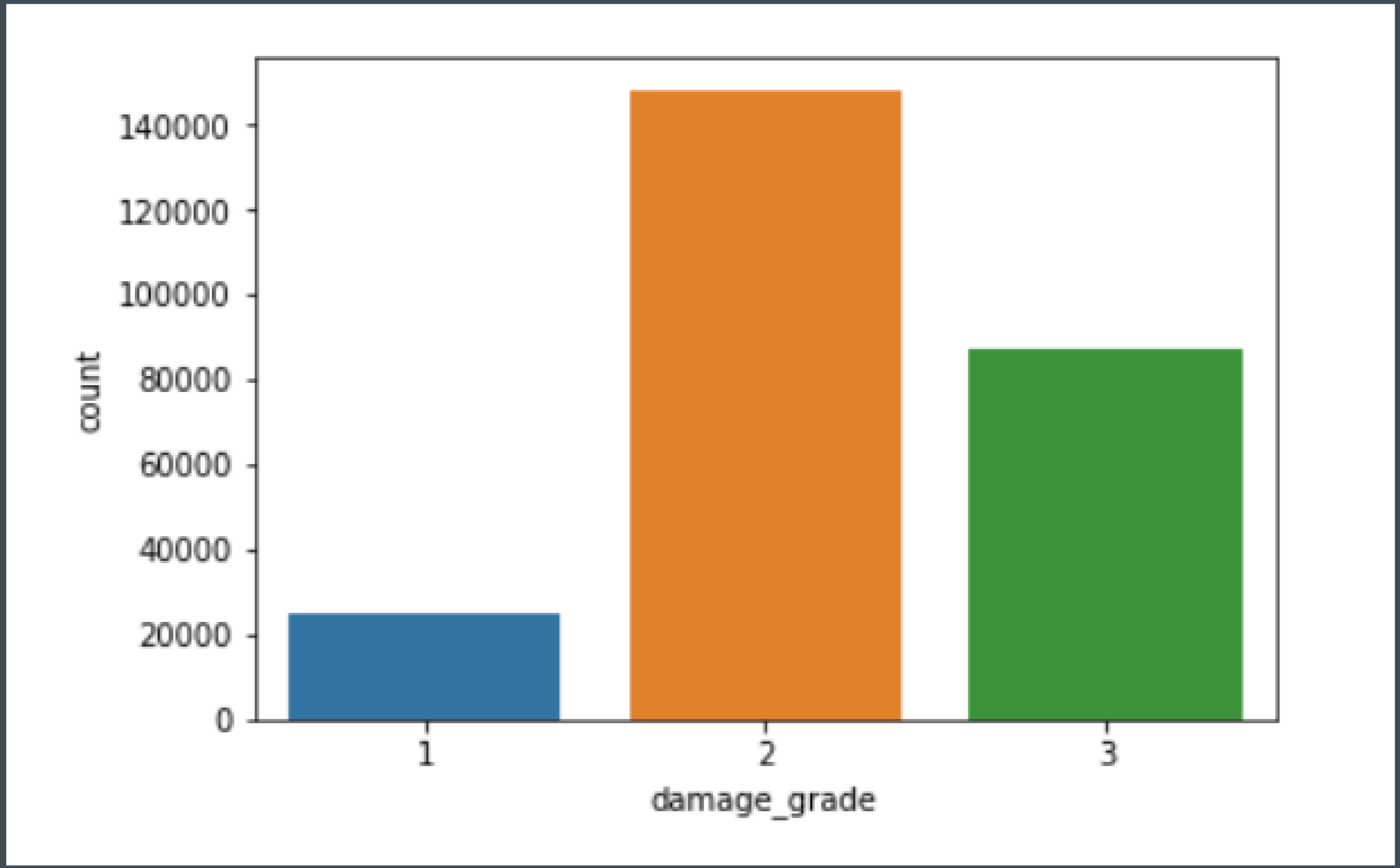
Two CSV files joined on building_id.

- No NAs
- Kaggle dataset was clean

Summary Stats

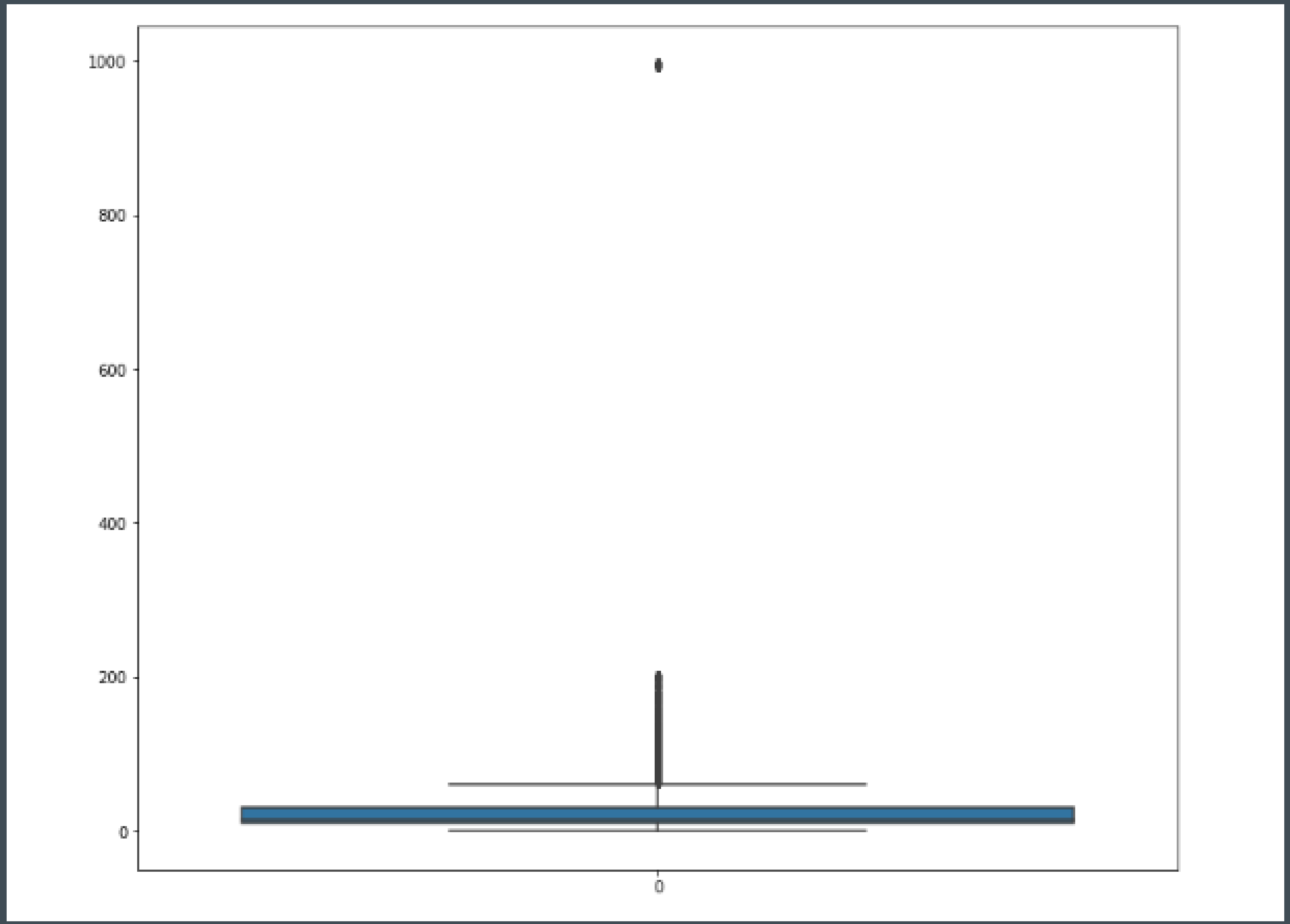
- **260,601** observations
- Key - **building_id**
- Label - **damage_grade**
- 38 variables (features)
  30 Numerical
  8 Categorical

Data Exploration



Histogram of Damage_Grade

- Skewed to Label 2
- Suggest to handle label imbalance in Machine Learning Algorithm

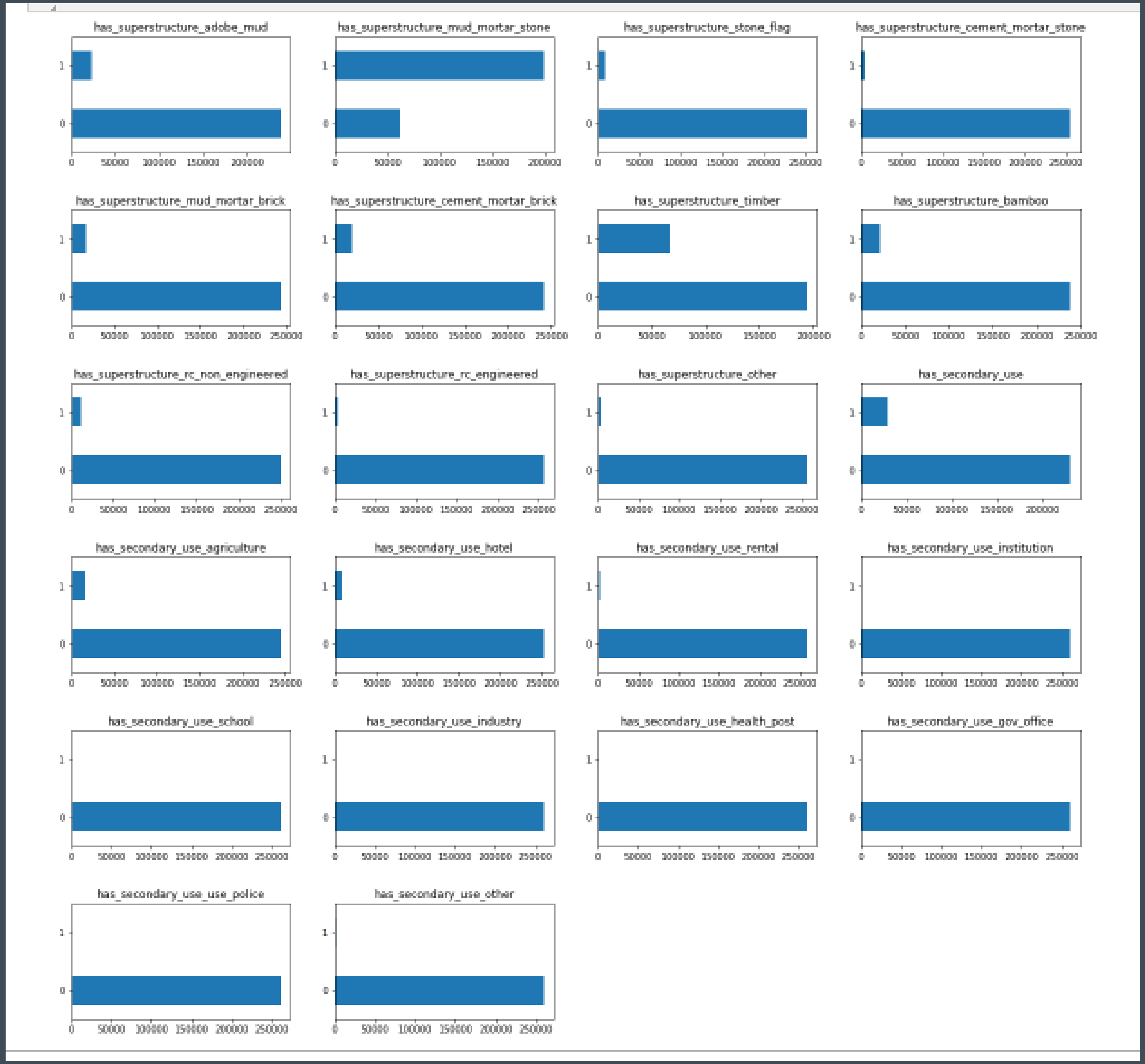


Outliers in Building Age

- Extreme outlier value of 900s
- Age > 200 account for 0.005% of total observations
- Suggest to create bins for Age to normalize the feature

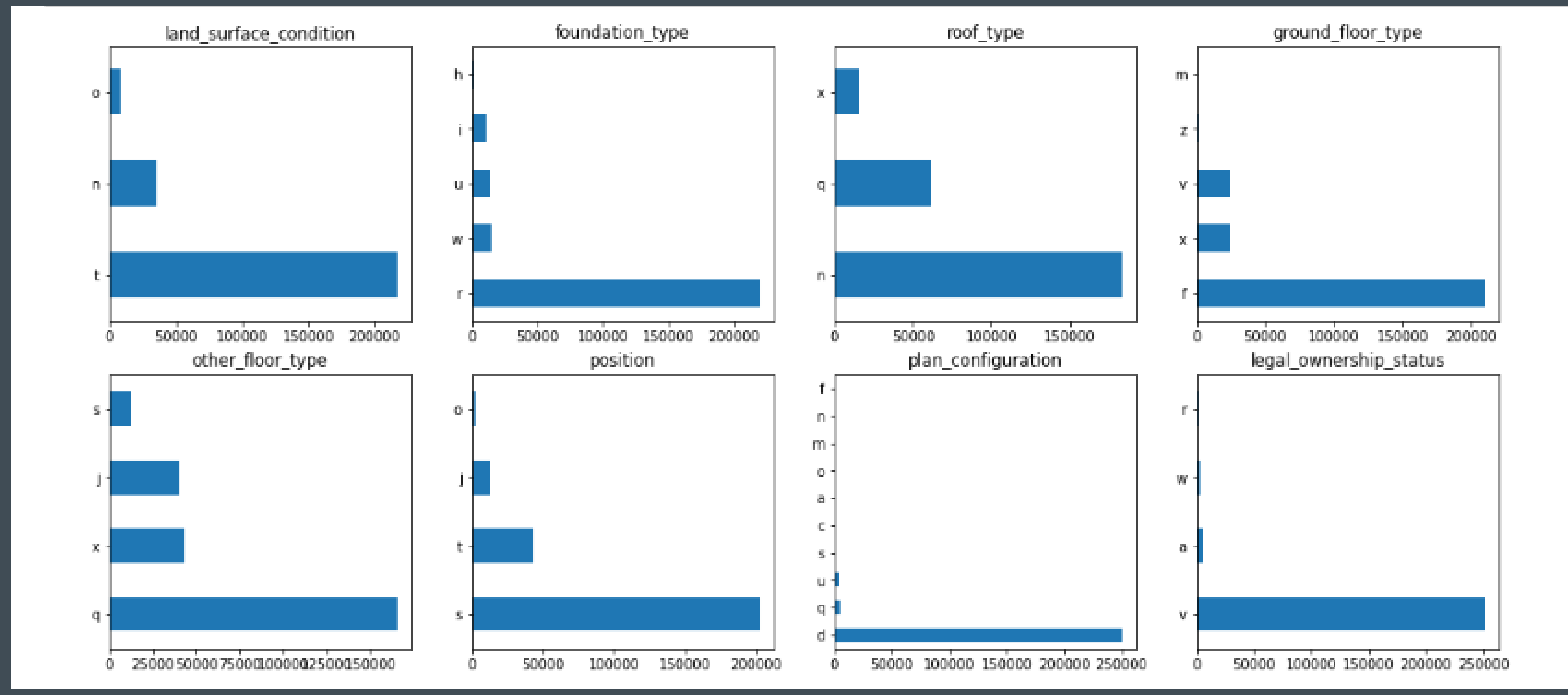
```
land_surface_condition : 3 labels values ['t', 'o', 'n'] foundation_type : 5 labels values ['r', 'w', 'i', 'u', 'h'] roof_type : 3 labels values ['n', 'q', 'x'] ground_floor_type : 5 labels values ['f', 'x', 'v', 'z', 'm'] other_floor_type : 4 labels values ['q', 'x', 'j', 's'] position : 4 labels values ['t', 's', 'j'
```

Data Exploration



Binarisation of Numerical Variables

- Columns beginning with "has_superstructure" and "has_secondary_use" are binary (0 or 1 values)
- Collapse "has_secondary_use" columns to a single numeric column.

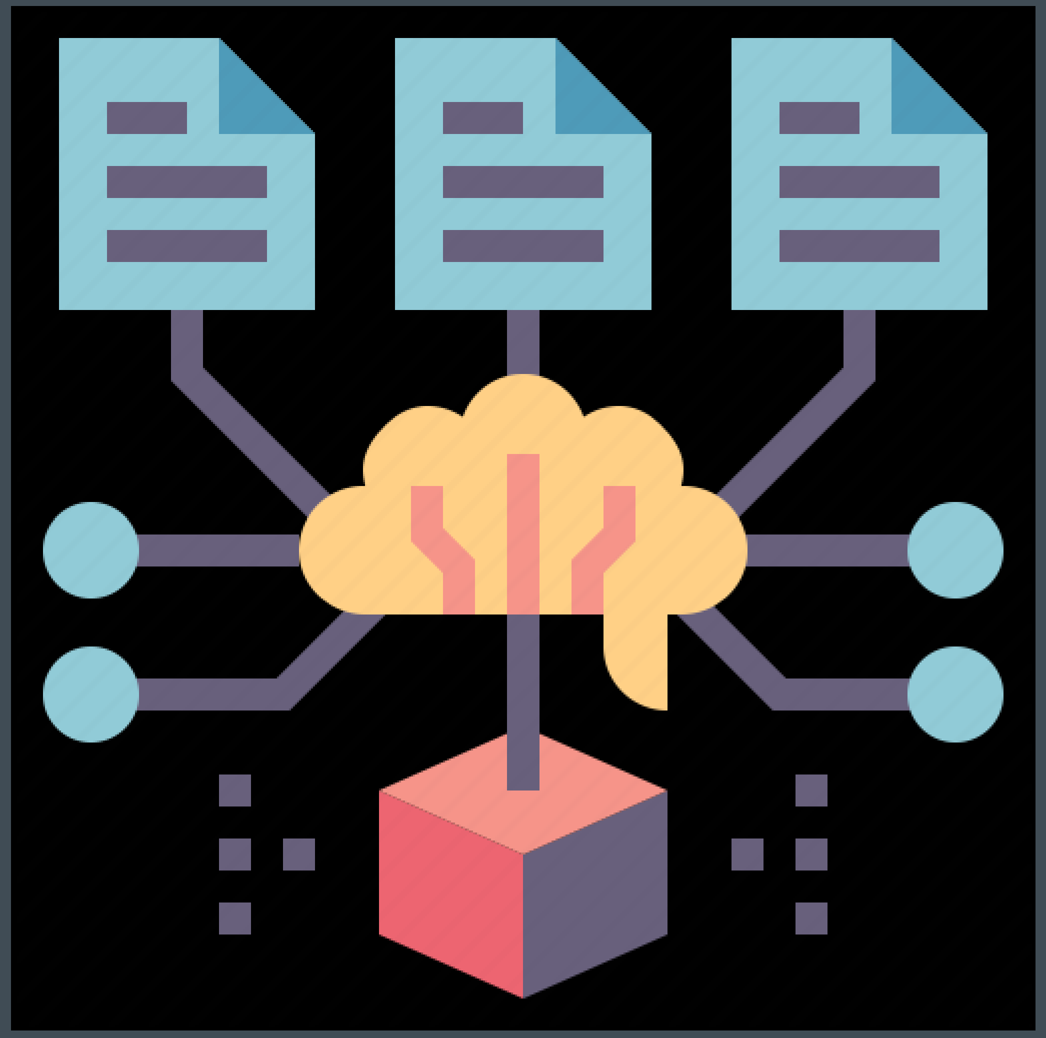


Distribution of Categorical Variables

- land_surface_condition, foundation_type, roof_type, ground_floor_type, other_floor_type, position, plan_configuration, legal_ownership_status

Feature Engineering

Age Group	Age	Count
0	< 5	26,041
1	5 - 9	33,697
2	10-24	107,088
3	25-49	68,374
4	50-100	21,913
5	> 100	3488



1. Bin Age of Building to remove Outlier effect.
   Create a new column “secondary_use” to collapse the 11 “has_secondary use” variables

2. Create pipeline to index Categorical variables, One-Hot Encode them, then normalize all variables. Fit a Random Forest algorithm to predict the label, damage_grade.

3. Extract top 25 features (out of a total of 59) of the classification algorithm. The feature importance score returned comes in the form of a sparse vector which will be ampped to obtain the actual variable names.

4. Construct a new features_top column to contain the top 25 features and create a new input vector column with only these variables.

5. Prediction accuracy dropped about 2.6% from reducing the full set of features to the top 25 features. Retain all the features.

Model Building & Evaluation

F1 score is used as the Model Evaluation Metric as it covers both Precision (False Postives) and Recall (False Negatives)

LOGISTIC REGRESSION

F1 score = 0.54

RANDOM FOREST

F1 score = 0.53

NEURAL NETWORK

F1 score = 0.41

Model Tuning

IMBALANCE HANDLING

F1 score increase to 0.55

HYPERPARAMETER TUNING

F1 score remains at 0.55 indicating that default parameters were okay.

✓ Best Performing Model is **Logistic Regression** at **55%** F1 score