Analysis and Prediction of severity of damage to residential buildings caused by earthquake using Machine learning model

Given the geo-location, structure details of the building, and household socio-economic demographics information predict the severity of damage to buildings caused by earthquake.

Table of Contents:

INTRODUCTION

Catastrophe modeling is an essential part of risk assessment and underwriting for insuring various property occupancies like residential, commercial, production plants, and so on.

It combines historical disaster information with current demographic, building, and financial data to determine the potential financial impact of catastrophes for a specified geographic area.

Catastrophe modeling firms like AIR Worldwide, RMS (Risk Management Solutions) and EQECAT have developed natural catastrophe models for a wide range of catastrophic risks like hurricanes, earthquakes, winter storms, tornadoes, and floods worldwide including specific industries. Insurers, reinsurers, rating agencies, risk managers, and major insurance brokers license models from these firms. Some also develop their own models.

The process of developing sophisticated catastrophe models is complex and draws on expertise from a broad range of technical and financial disciplines. The models utilize the skills of many experts, including meteorologists, seismologists, geologists, engineers, mathematicians, actuaries, decision scientists, and statisticians.

However, in this case study, we will only focus on earthquake peril and its damage to residential buildings. We will explore historical earthquake event data and build a machine-learning model to identify the level of damage to buildings caused by an earthquake. We will also include socio-economic demographics data to capture the physical and social status of affected areas and see if it helps in explaining the severity of damage.

MOTIVATION

As per statista.com, economic loss due to natural disaster events worldwide amounted to about 1.5 trillion US dollars in the last five years. So, it becomes important for any individual or business to safeguard themselves from huge financial loss caused by an unforeseen event like fire, theft, or any natural disaster like Earthquake, Flood, Windstorm, Tornado, Hail, Hurricane, or loss due to machinery breakdown in case of production plants. Providing protection against such financial loss by charging a small premium compared to property value is called Property Insurance. It is estimated that the global commercial insurance market is projected to reach from USD 692.33 Billion in 2020 to USD 1,227.8 Billion by 2028.

From a property insurer's perspective, it is important to price the premiums effectively against each peril and for various occupancy types depending on their expected financial loss to be competitive in the market and profitable at the same time. So, they employ a Catastrophe Risk Modeling analyst to scrape the client's property details, run the model, assess the risk to identify the vulnerability and severity of property damage to generate the expected loss and charge accordingly for various catastrophe perils

DATA COLLECTION

The data was collected through surveys by Kathmandu Living Labs and the Central Bureau of Statistics, which works 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. It is publicly available through the 2015 Nepal Earthquake Open Data Portal. Also available in my GitHub repository.

The data mainly contains the following:

1. Building Structural information, age of the building, geo-location, and target variables to determine the severity of damage.

2. Ownership details, household conditions, and resources data

BUSINESS CONSTRAINT

- Interpretability is important to some extent
- No strict low latency concerns
- High accuracy. Errors affect the pricing of premiums and can cost writing business
- Predicting the probability of a point belonging to each class is not necessary

PERFORMANCE METRIC

- Both precision and recall are important so the Micro-F1 score is a good choice as data is imbalanced and we care about the overall accuracy
- Confusion matrix, recall matrix, and precision matrix to see how our model is performing on train and test data for each class
- Classification report to see how our model is performing in each class

DATA CLEANING

1. Building structure and ownership data:

   There are 762106 records and 44 features. There are some features whose data is collected post-earthquake event and some records with missing Target values. We will drop them.

2. Identifying Target variable:

   There are 2 features ‘damage grade’ with 5 unique grades and ‘technical solution proposed’ with 4 unique values. Based on the 91.27% of data, we will create our target variable ‘severity of damage’ with 3 unique values ‘Mild’, ‘Moderate’, and ‘Severe’

3. Household conditions and resources data:

   There are 747365 records and 43 features. We will drop features whose data are collected post-earthquake event and the records with missing values

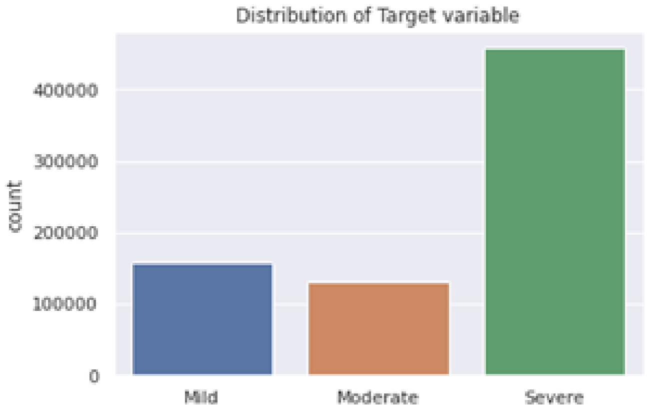
   On joining both these data tables, our final data contains 747124 records with 61 features and 1 target variable

Some of the key features include geo-location_ids, count_floors, height_building, age_building, plinth_area, foundation_type, has_superstructure, has_secondary_use, etc

EXPLORATORY DATA ANALYSIS

1. Target variable:

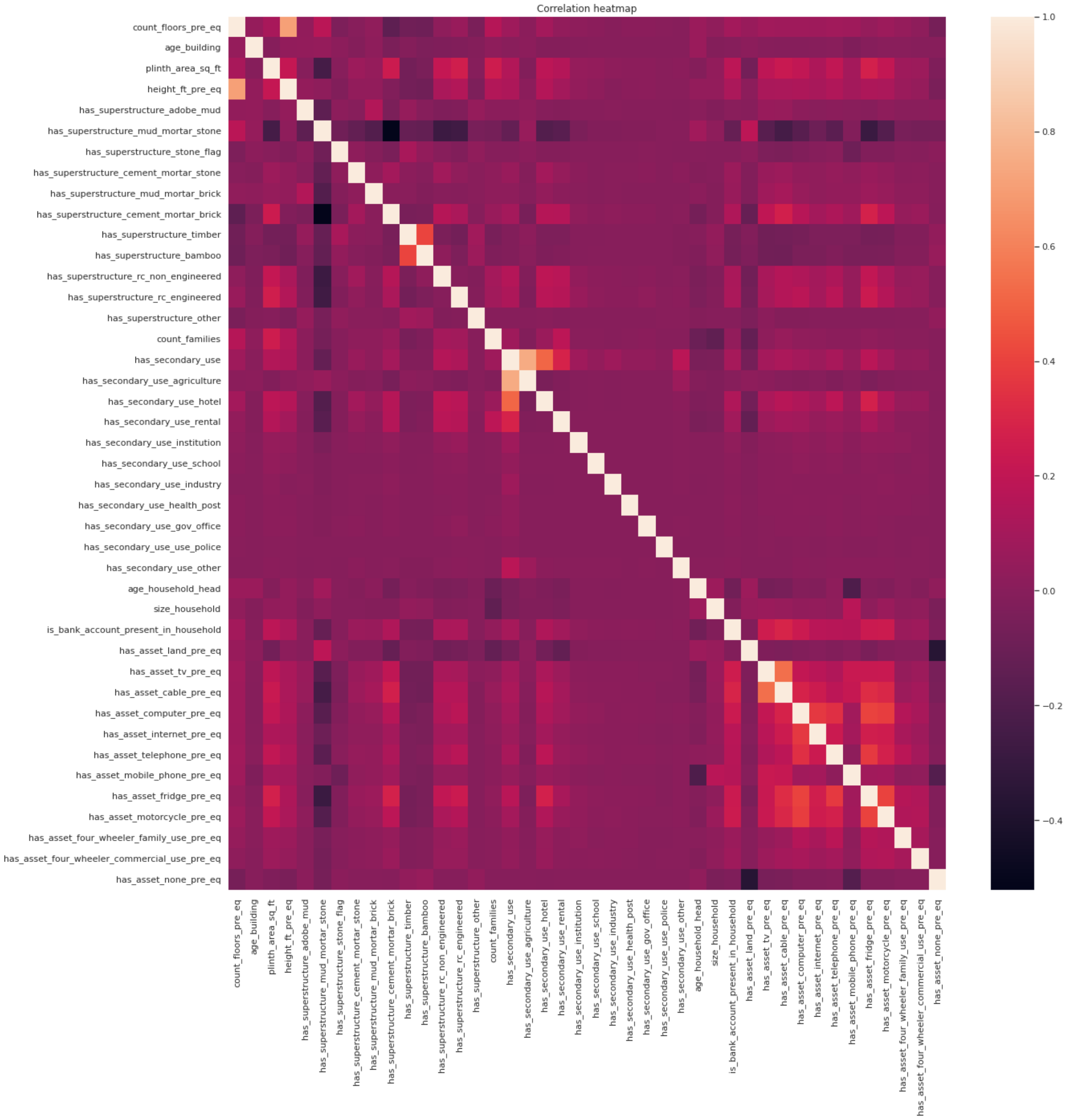
   The data is imbalanced. About 61.3% of records belong to class ‘Severe’, 21% of records belong to class ‘Moderate’ and the remaining 17.7% of records belong to class ‘Mild’



2. Exploring Numerical features

a. Correlation:

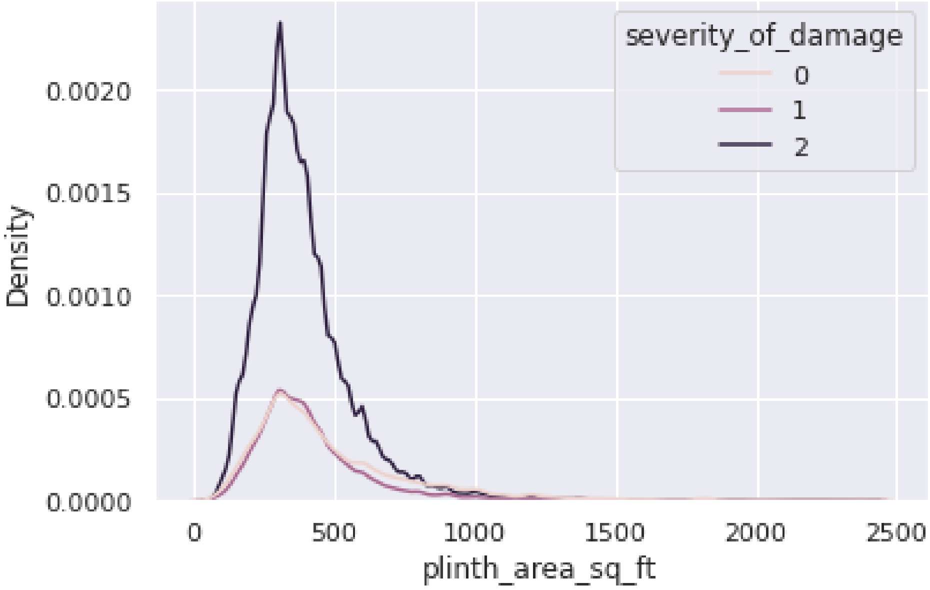
There are no features with a correlation greater than 0.8. Features 'height_ft_pre_eq', 'count_floors_pre_eq' and 'has_secondary_use', 'has_secondary_use_agriculture' are correlated and have correlation factor above 0.7



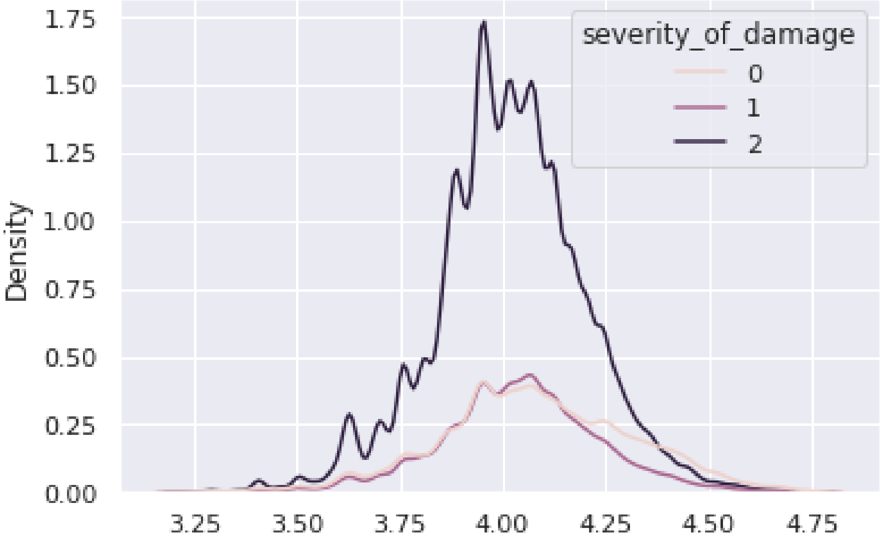
b. Univariate Analysis:

Features like count floors, age, height, plinth area, etc have skewed data. Applying box-cox transformation is beneficial. Sample graph below

Original distribution

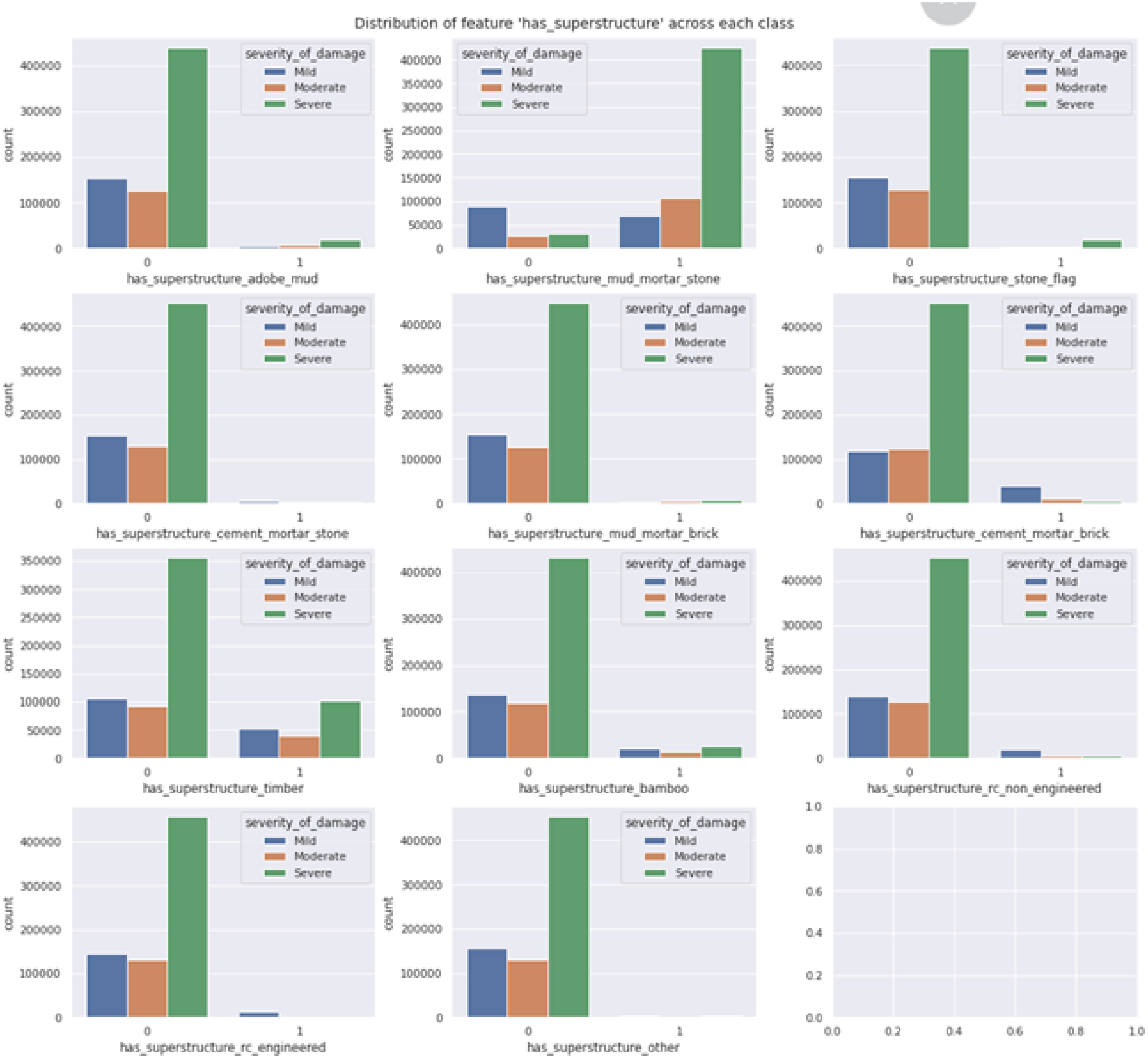


After BoxCox Transformation



c. Has_superstructure features

Below is the distribution of 'has_superstructure' across each class. We can see that superstructure with cement_mortar_brick or reinforced_concrete or cement_mortar_stone can withstand earthquakes to some extent compared to mud, mud_mortar_stone, etc



Features like geo-location, and structural details like age, height, plinth area, and construction material were some of the most important features in classifying the severity of damage. Also, some of the demographic's features were slightly useful in distinguishing the severity of damage

3. Exploring Categorical features

Features like vdcmun_id, ward_id, and caste_household have high cardinality. One hot encoding will result in high dimensionality and sparse data. Hence, we can look for other encoding techniques like Target encoding, embedding them using neural networks, etc

Let's build the model and feature importance to identify the most important ones at the last.

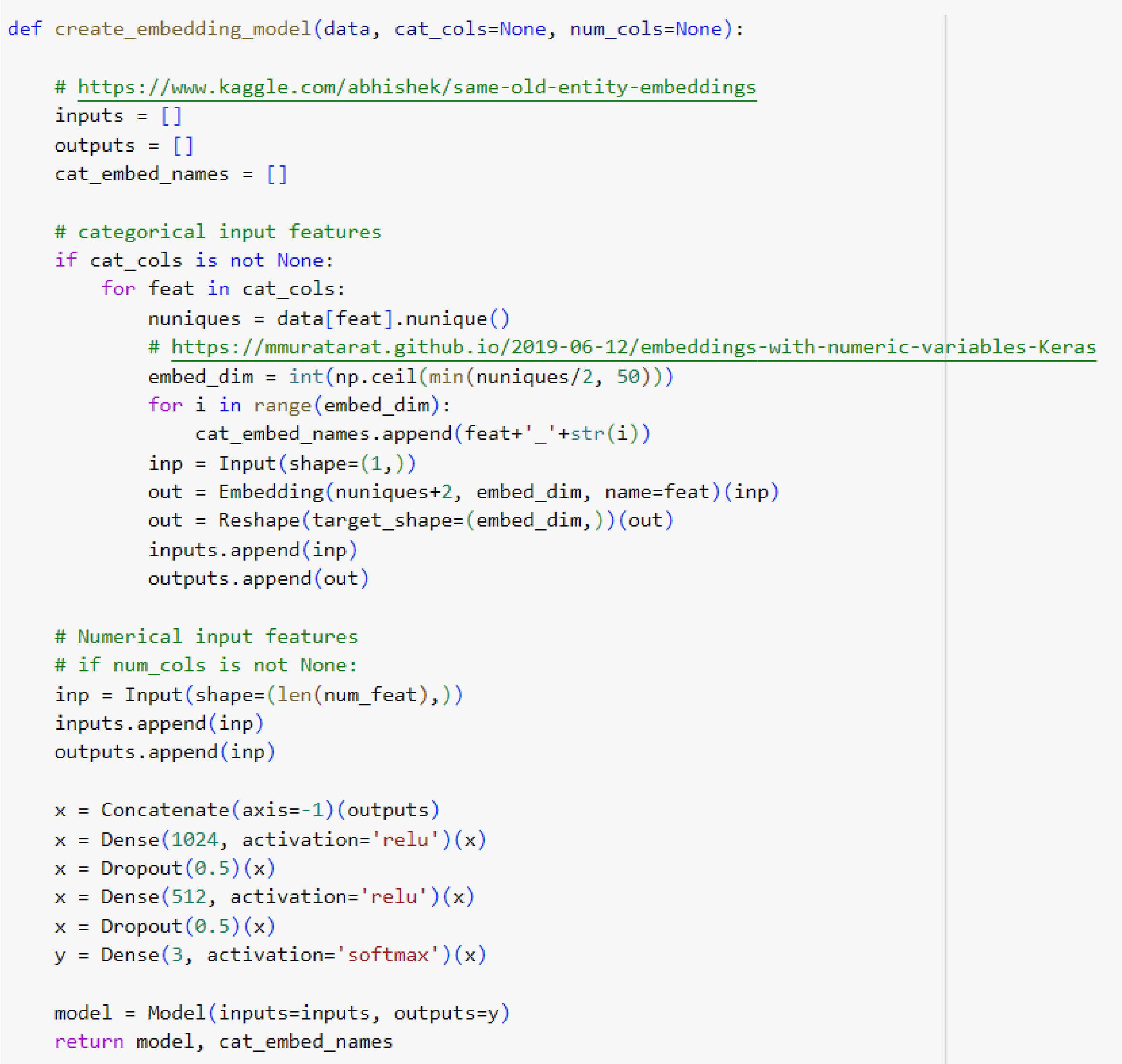
DATA PREPROCESSING

1. Cleaned data is split into train and test data in the ratio 85:15 and the distribution resemble the actual data

2. Target variable classes are mapped to integer

3. Numerical features: Outlier points for features like age-building, plinth-area, height-ft, age-household, and size-household are removed and applied box-cox transformation to reduce the skewness and resemble the normal distribution for Generalized Linear Models (GLM) that make assumptions on the distribution of data

4. Categorical features: Text data is preprocessed to remove any special characters. We have seen label encoding, one hot encoding, and some advanced feature transformation techniques like Target Encoding to convert categorical variables into a numerical representation. However, in this project entity embedding is tried wherein high cardinal categorical variables are embedded in lower dimensions while preserving the relationship between each of the categories using a shallow neural network as defined by a research paper. *(Reference in the resources below)*



```python
def create_embedding_model(data, cat_cols=None, num_cols=None):

    # https://www.kaggle.com/abhishek/same-old-entity-embeddings
    inputs = []
    outputs = []
    cat_embed_names = []

    # categorical input features
    if cat_cols is not None:
        for feat in cat_cols:
            nuniques = data[feat].nunique()
            # https://mmuratarat.github.io/2019-06-12/embeddings-with-numeric-variables-Keras
            embed_dim = int(np.ceil(min(nuniques/2, 50)))
            for i in range(embed_dim):
                cat_embed_names.append(feat+'_'+str(i))
            inp = Input(shape=(1,))
            out = Embedding(nuniques+2, embed_dim, name=feat)(inp)
            out = Reshape(target_shape=(embed_dim,))(out)
            inputs.append(inp)
            outputs.append(out)

    # Numerical input features
    # if num_cols is not None:
    inp = Input(shape=(len(num_feat),))
    inputs.append(inp)
    outputs.append(inp)

    x = Concatenate(axis=-1)(outputs)
    x = Dense(1024, activation='relu')(x)
    x = Dropout(0.5)(x)
    x = Dense(512, activation='relu')(x)
    x = Dropout(0.5)(x)
    y = Dense(3, activation='softmax')(x)

    model = Model(inputs=inputs, outputs=y)
    return model, cat_embed_names
```

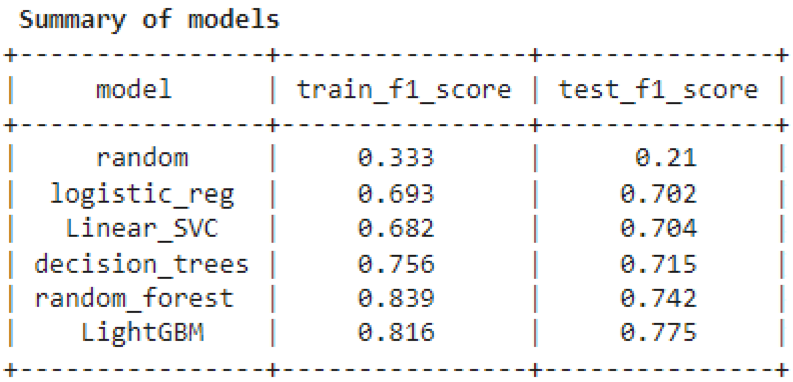
5. Now our data is ready. We need to handle an imbalance in the data. Three datasets are prepared by over-sampling, under-sampling, and combined sampling

6. Embedding model on undersampled data performed better in terms of overall micro-f1 score. However, the embedding model performed slightly better on oversampled data for the minority class ‘Moderate’ compared to that of undersampled data. So, Categorical features are embedded using oversampled data

7. Scaling the data: Tried Standardizing and Normalizing the data. Standardization with mean 0 and 1 standard deviation performed slightly better

DATA MODELING

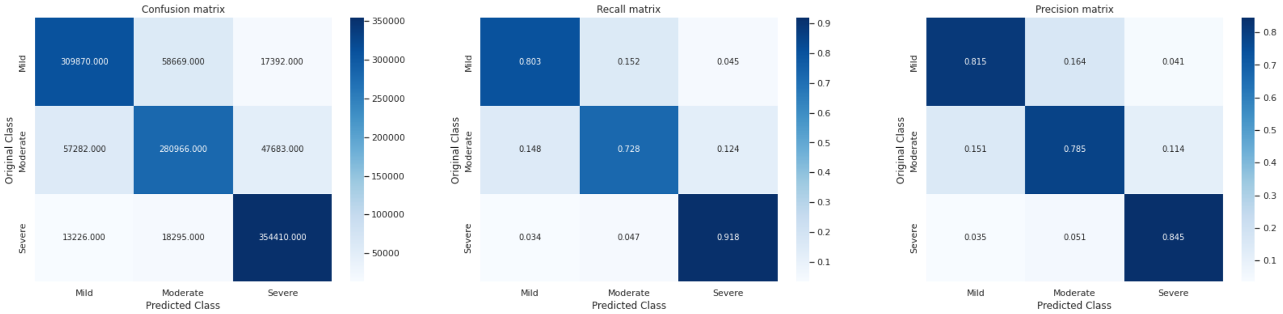
The preprocessed data is modeled using different models like a Dummy classifier that sets the base performance, Linear models like logistic regression, and linear SVC, and Non-Linear models like Decision Trees, Random Forest, and Light gradient boosting. The performance of all these models is shown in the table below. All these models are hyperparameter tuned using TunesearchCV which uses the Bayesian optimization technique.



```
 Summary of models
+----------------+----------------+---------------+
|     model      | train_f1_score | test_f1_score |
+----------------+----------------+---------------+
|     random     |     0.333      |      0.21     |
|  logistic_reg  |     0.693      |     0.702     |
|   Linear_SVC   |     0.682      |     0.704     |
| decision_trees |     0.756      |     0.715     |
| random_forest  |     0.839      |     0.742     |
|    LightGBM    |     0.816      |     0.775     |
+----------------+----------------+---------------+
```

We can see from the above table that LightGBM performed well compared to other models. Below is the confusion matrix and classification report for the LightGBM model.

Confusion matrix on train data



Confusion matrix

Original Class	Mild	Moderate	Severe
Mild	309870.000	58669.000	17392.000
Moderate	57282.000	280966.000	47683.000
Severe	13226.000	18295.000	354410.000

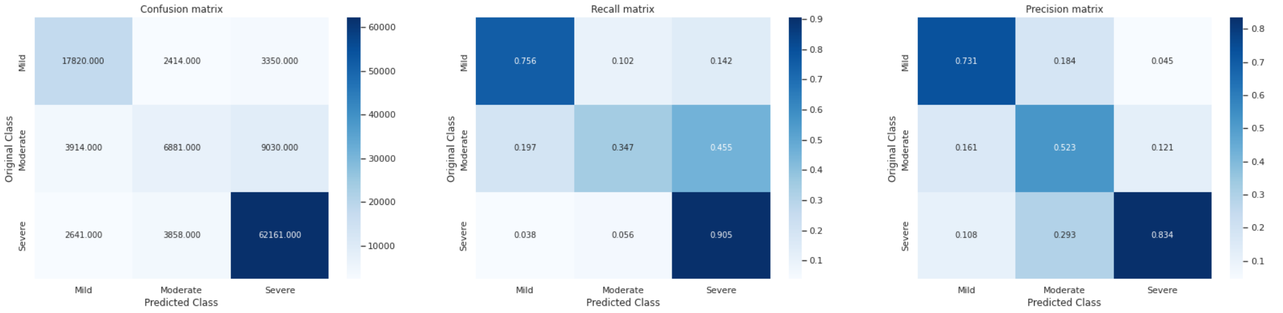
Recall matrix

Original Class	Mild	Moderate	Severe
Mild	0.803	0.152	0.045
Moderate	0.148	0.728	0.124
Severe	0.034	0.047	0.918

Precision matrix

Original Class	Mild	Moderate	Severe
Mild	0.815	0.164	0.041
Moderate	0.151	0.785	0.114
Severe	0.035	0.051	0.845

Confusion matrix on test data



Confusion matrix

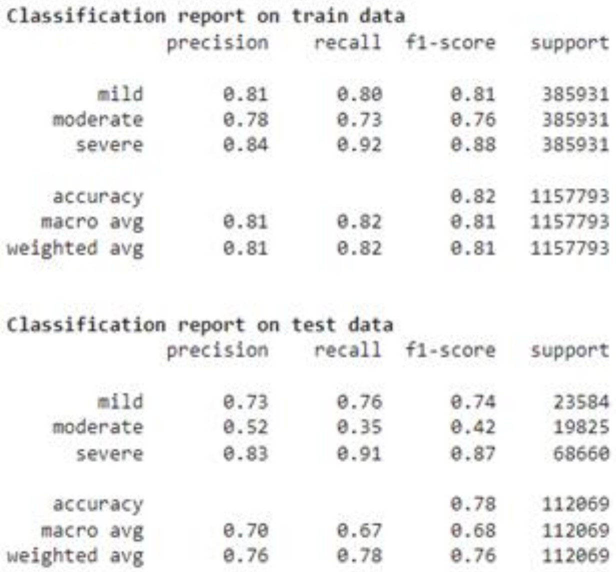
Original Class	Mild	Moderate	Severe
Mild	17820.000	2414.000	3350.000
Moderate	3914.000	6881.000	9030.000
Severe	2641.000	3858.000	62161.000

Recall matrix

Original Class	Mild	Moderate	Severe
Mild	0.756	0.102	0.142
Moderate	0.197	0.347	0.455
Severe	0.038	0.056	0.905

Precision matrix

Original Class	Mild	Moderate	Severe
Mild	0.731	0.184	0.045
Moderate	0.161	0.523	0.121
Severe	0.108	0.293	0.834

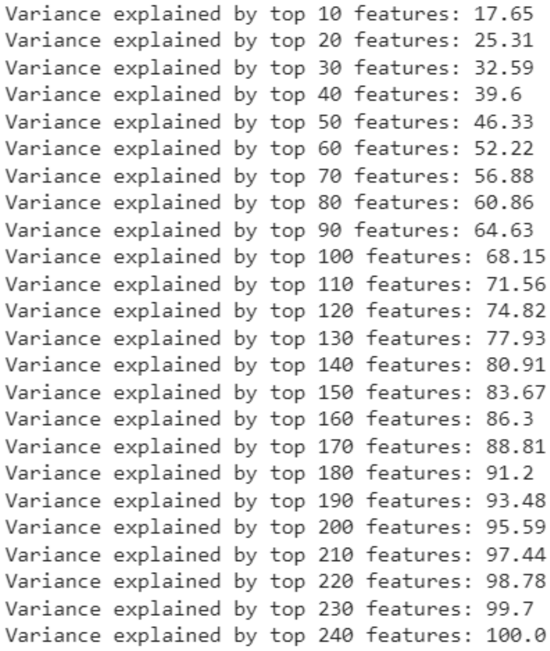


```
Classification report on train data
              precision    recall  f1-score   support

        mild       0.81      0.80      0.81    385931
    moderate       0.78      0.73      0.76    385931
      severe       0.84      0.92      0.88    385931

    accuracy                           0.82   1157793
   macro avg       0.81      0.82      0.81   1157793
weighted avg       0.81      0.82      0.81   1157793

Classification report on test data
              precision    recall  f1-score   support

        mild       0.73      0.76      0.74     23584
    moderate       0.52      0.35      0.42     19825
      severe       0.83      0.91      0.87     68660

    accuracy                           0.78    112069
   macro avg       0.70      0.67      0.68    112069
weighted avg       0.76      0.78      0.76    112069
```

From the above graphs and report, we can see that the optimized LightGBM model is able to generalize well on majority class ‘Severe’ followed by ‘mild’ and overfitting on ‘moderate’ class

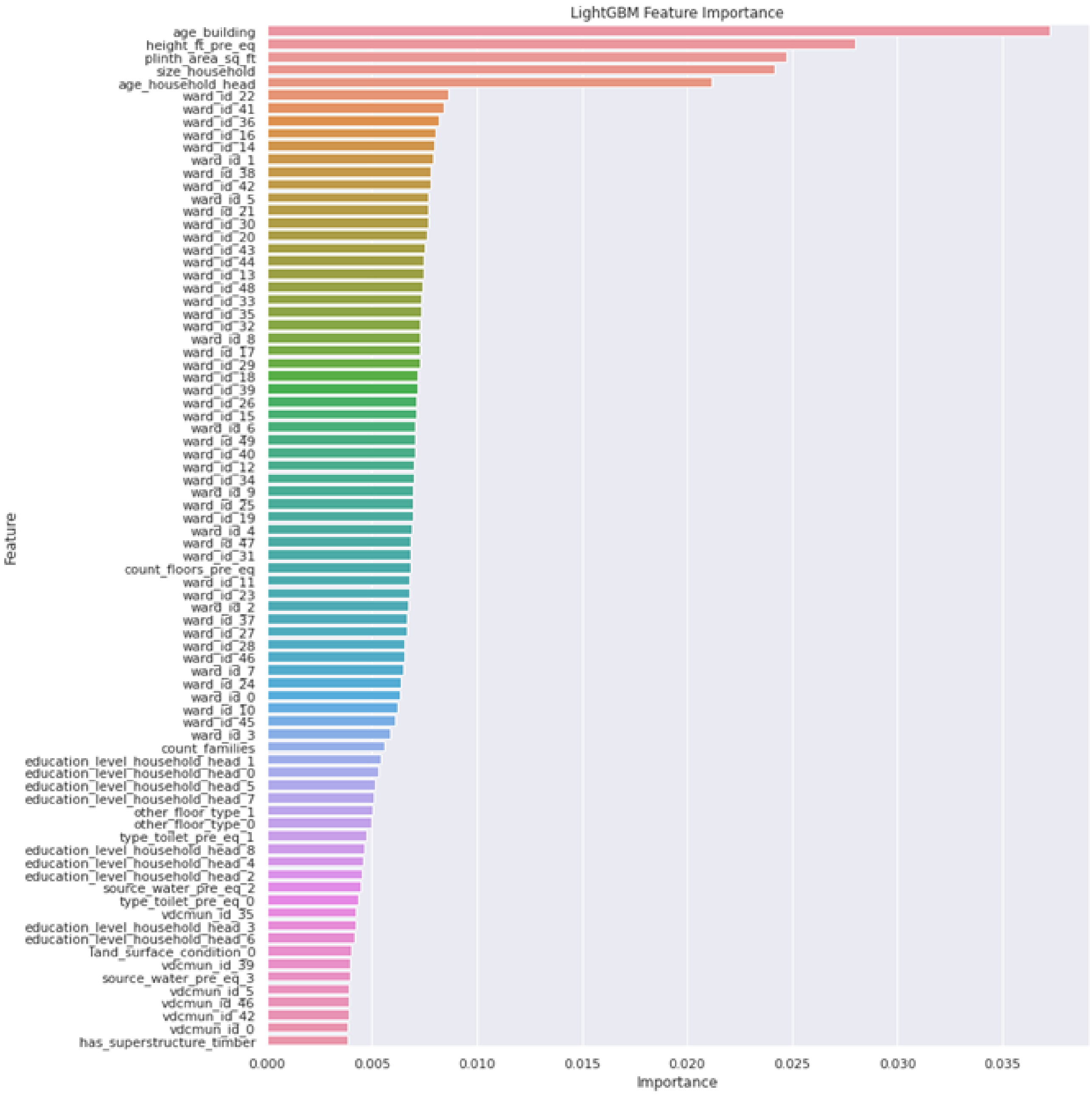
FEATURE IMPORTANCE:

Let’s find the top features that are super useful for distinguishing the severity of damage



```
Variance explained by top 10 features: 17.65
Variance explained by top 20 features: 25.31
Variance explained by top 30 features: 32.59
Variance explained by top 40 features: 39.6
Variance explained by top 50 features: 46.33
Variance explained by top 60 features: 52.22
Variance explained by top 70 features: 56.88
Variance explained by top 80 features: 60.86
Variance explained by top 90 features: 64.63
Variance explained by top 100 features: 68.15
Variance explained by top 110 features: 71.56
Variance explained by top 120 features: 74.82
Variance explained by top 130 features: 77.93
Variance explained by top 140 features: 80.91
Variance explained by top 150 features: 83.67
Variance explained by top 160 features: 86.3
Variance explained by top 170 features: 88.81
Variance explained by top 180 features: 91.2
Variance explained by top 190 features: 93.48
Variance explained by top 200 features: 95.59
Variance explained by top 210 features: 97.44
Variance explained by top 220 features: 98.78
Variance explained by top 230 features: 99.7
Variance explained by top 240 features: 100.0
```

- The performance of the model decreases if we reduce the number of dimensions as most features are required to explain most of the variance.



- The top 5 features are the age_of_building, height_of_building, plinth area, size of household, and age of household head.
- Building structure details like geo-location, age, area, number of floors, construction, etc play vital roles in determining the severity of damage.
- Interestingly we can see that socio-economic feature like size of household, age of household head, education level, type of toilet, source of water, etc also play important roles in determining the severity of damage.
- Embedded features obtained using neural networks also capture most of the variance in the data.

Resources:

Catastrophe modeling: A vital tool in the risk management box https://www.iii.org/article/catastrophe-modeling-vital-tool-risk-management-box

2. https://reliefweb.int/report/nepal/open-data-portal-2015-earthquake-launched-national-planning-commission

3. https://medium.com/analytics-vidhya/categorical-embedder-encoding-categorical-variables-via-neural-networks-b482afb1409d

4. https://machinelearningmastery.com/how-to-prepare-categorical-data-for-deep-learning-in-python/

5. https://www.drivendata.org/competitions/57/nepal-earthquake/page/134/

6. https://towardsdatascience.com/deep-embeddings-for-categorical-variables-cat2vec-b05c8ab63ac0

7. Entity Embeddings of Categorical Variables: *https://arxiv.org/abs/1604.06737v1*

For a more in-depth exploration of this analysis, check out the GitHub repository linked here.