NEPAL 2015 EARTHQUAKE

June 9th, 2019		
Drexel University	Jinesha Jain Apurva Tawde Michelle Do	Yagna Ganesh Easwaran Khanjan Patel

TABLE OF CONTENTS

ABSTRACT

The main target of this project is to detect a damage level to the building. we try six different classifiers: Support Vector Machine, K-Nearest Neighbor, Naïve Bayes, Multi Nomial Logistic regression and Random Forest. Random Forest performs best based on my experiments. A fine-tuning of Random Forest is performed to check if a better performance can be achieved.

INTRODUCTION

Our topic dates to one of 1583 reported natural disasters in the world within the last 5 years – the Nepal 2015 Earthquake.

An earthquake of 7.8 magnitude struck Nepal on April 25, 2015 less than 50miles from the Country's Capital, Kathmandu causing devastation across the impoverished Himalayan Nation. Apart from Nepal, the earthquake's affected Himalayan's regions of Tibet, India and Bangladesh. As a matter of fact, scientists have agreed that the Mt. Everest moved south by 3cms after this natural disaster. Massive destruction from the earthquake has, 11 districts affected, with over 700,000 buildings damaged and over 3M individuals affected. From an economic point of view, the damage done due to the earthquake was estimated to be around 10billion USD which is about 50% of Nepal's GDP. Nepal's biggest contributor to GDP is tourism, which decreased by 72% after the Earthquake. Additionally, World Bank reported that an additional 3% of Nepal's population has been pushed into poverty.

Though a Natural Disasters of any kind on the globe, has a negative affect overall. Unlike Japan which is well developed country, Nepal on the other hand is a developing country. And for a country that was not well equipped technologically and lacked awareness to handle such conditions with most the infrastructure and buildings in Nepal being old, contributed towards being one of the most destructive earthquakes the country has seen since 1934.

Therefore, our aim is to classify the level of damage done to the buildings. Where a damage of 1 is referred to as low damage, 2 as medium damage and a damage of means almost complete destruction of the building. To generate insights on the building structure and the damage caused. This will not only aid in estimating the damage that will be done and know what type of buildings from the type of land, soil and structure contribute to more destruction and what type doesn't for similar developing countries like Nepal that are prone to Earthquake but also in helping Nepal rebuild into a stronger country so as to avoid such massive destructions and be better prepared.

DATA OBSERVATION

The data was collected by a survey conducted the National Planning Committee of Nepal. For this project we used a total of over 260,000 samples with 39 variables including the dependent variable *damage_grade*. Since most of our data is categorical we have 58 variables upon dummyfication. The dataset mainly consists of information on the building's structure and its legal ownership, where each row in the dataset represents a specific building in the region that was hit by the Earthquake. The data mainly consisted of 4 types of variables (refer to Table 1)

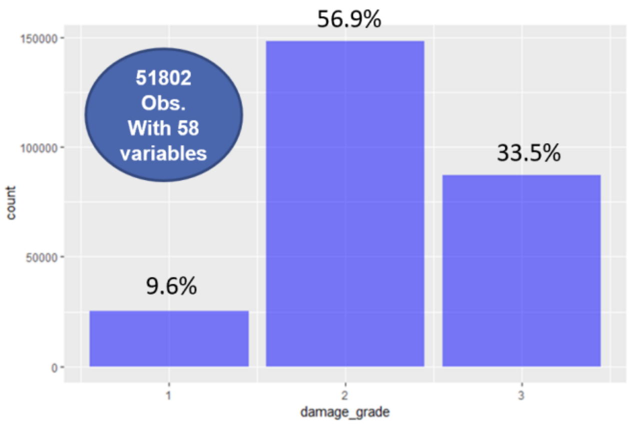
Table 1: Summary of independent variables

Variable type	**Quantity**	**Examples**
Binary (Boolean)	22	Has_superstructure_XXXX, has_secondary_use_XXXX
Integer, interval	4	Count_floors_per_eq, age, area
Float	1	Count_families
Categorical	11	Geo_level_1_id, geo_level_2_id, geo_level_3_id, land_surface_condition, foundation type, roof_type, ground_floor_type, position, plan_configuration, legal_ownership_status
Ordinal	1	Damage_grade (the dependent variable)

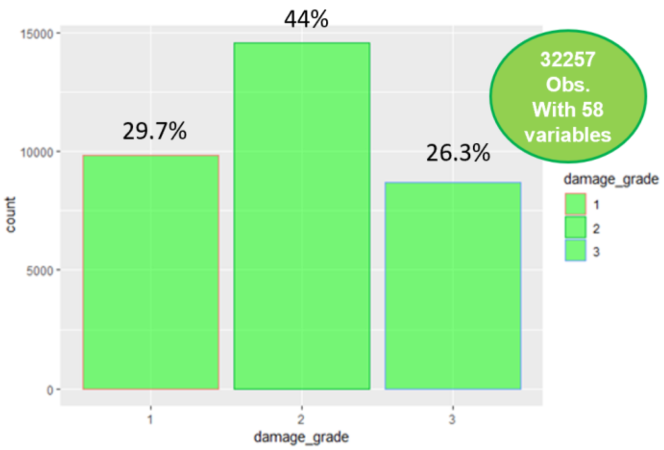
IMBALANCED DISTRIBUTION OF CLASS

We have a total of 260,600 observations, working with big data set was an issue, we divided the data set into 5 groups with equal weight and randomness in the data as the original data set. The issue we have was of imbalance dataset with the highest observation of 57% belong to damage_grade 2 followed by 33% to damage_grade 3 and 10% belong to damage_grade 1 for the classification variable damage_grade.

Graph1: Imbalanced Data



Graph 2: Balanced Data



Synthetic Minority Over-sampling Technique (SMOTE) solves this problem. SMOTE() loops through the existing, real minority instance. At each loop iteration, one of the K closest minority class neighbors is chosen and a new minority instance is synthesized somewhere between the minority instance and that neighbor. We are able to achieve a class distribution of 30% in damage_grade 1,44% in damage_grade 2 and 26% in damage_grade 3.We use the new balanced data set to perform feature selection using 3 techniques, Maximum relevance minimum redundancy, Boruta & Stepwise.

FEATURE SELECTION

MAXIMUM RELEVANCE MINIMUM REDUNDANCY

A challenge in our dataset is large no of predictor variables, which may be later used as inputs to classification models. Removing irrelevant features may lead to improved accuracy and increased interpretability of the classification model. 43 variables were selected from a total of 58 variables in this method (Refer to Appendix I.A).

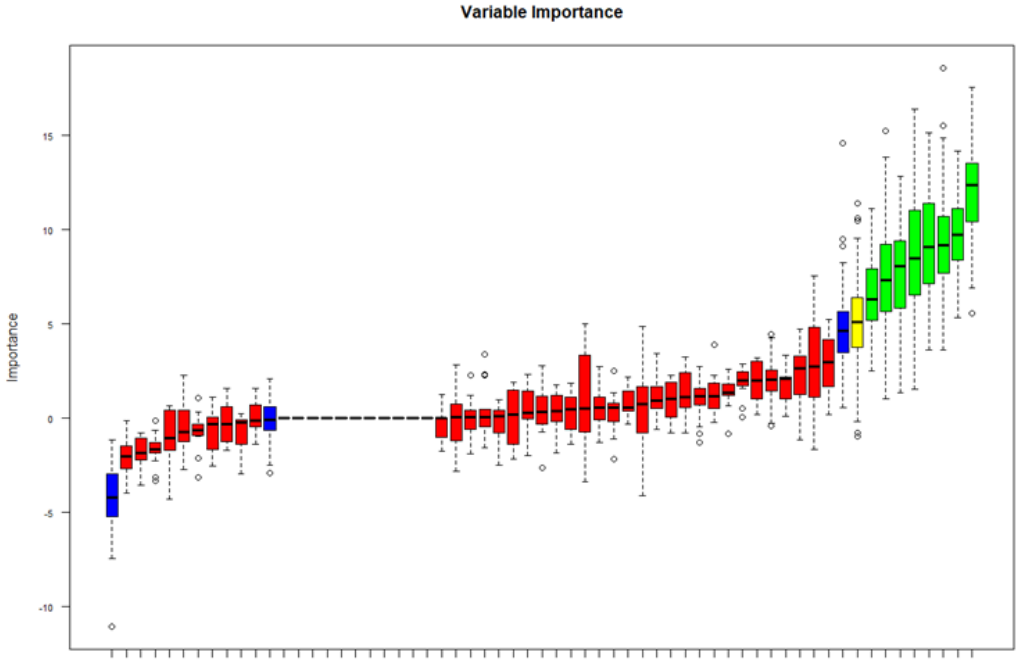
The main advantage of this method is that the maximum relevance criteria along with minimum redundancy criteria is used to choose features that are maximally relevant to the criteria and minimally redundant with respect to the criteria. Using mRMR the representative power of the feature set can be expanded, and their generalization properties can be improved.

BORUTA

It is one of the feature selection packages available in R. The basic idea of the algorithm is that we make a randomized copy of the system, merge the copy with the original and build the classifier for this extended system. To assess the importance of the variable in the original system we compare it with that of the randomized variables. Only variables for whose importance is higher than that of the randomized variables are considered important.

In our case, the damage_grade is the dependent variable and all the others are predictors. The Boruta algorithm starts by duplicating every independent variable by making a row-for-row copy, it permutes the order of the values in each column. So, in the copied columns, there should be no relationship then between the values and the target variable. Boruta then trains a Random Forest to predict the number of variables. Boruta package runs random forest on both original and random attributes and computes the importance of all variables. Since the whole process is dependent on permuted copies, we repeat a random permutation procedure to get statistically robust results.

Graph 3: Variable Importance



The algorithm then compares the variable importance scores for each variable with its copied columns. If the distribution of variable importance's is significantly greater in original than it is in a copied one, then the Boruta algorithm considers that variable significant. This generated 43 significant variables (Refer to Appendix I.B).

STEP-WISE SELECTION

Stepwise regression is a feature selection method that compare features one by one instead of grouping features to consider as a whole. In this project we use stepwise regression's backward elimination, which starts with all candidate variables, tests the deletion of each variable using a chosen model fit criterion, deletes the feature (if any) whose loss gives the most statistically insignificant deterioration of the model fit and repeats this process until no further variables can be deleted without a statistically significant loss of fit. At the end of the elimination process, it returns 33 most significant features (refer to Appendix I.C), which do not have much overlap with MRMR selected features. In hindsight, the stepwise selected dataset performed worse than that from MRMR.

CLASSIFICATION METHODS

MULTI-CLASS SVM

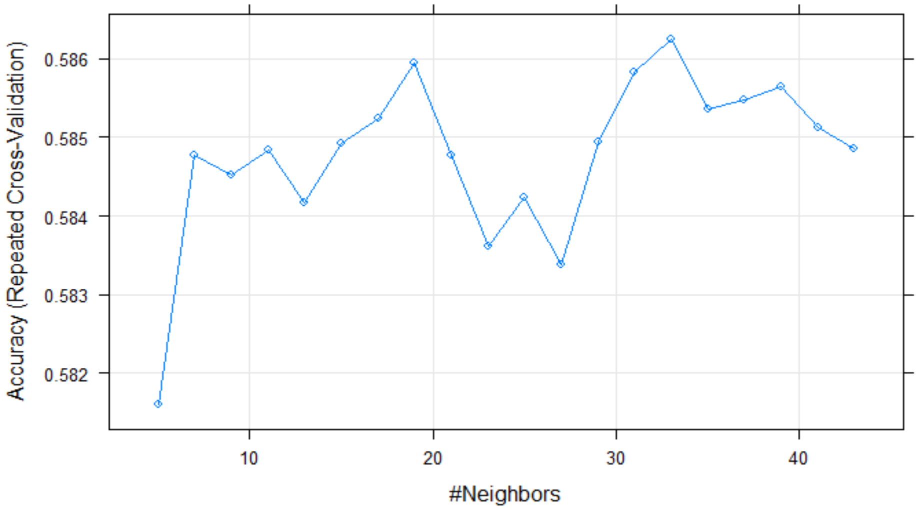
SVM() in the package "e1071" is used for actual classification and it handles variable scaling inside the function. We use a classification method using the radial kernel and kept the gamma value to be 0.1 to avoid any overfitting in the model.one of the drawbacks of using SVM over large dataset is that it takes higher training time. SVM doesn't directly provide probability estimates, these are calculated using an expensive five-fold cross-validation.

By running the model over the test set accuracy of 64.69% was achieved with all the predictor variable. To improve the model the results of MRMR feature selection was applied to the model resulting only slight change in accuracy of 65.26% and not improving the model significantly.

K-NEAREST NEIGHBORS

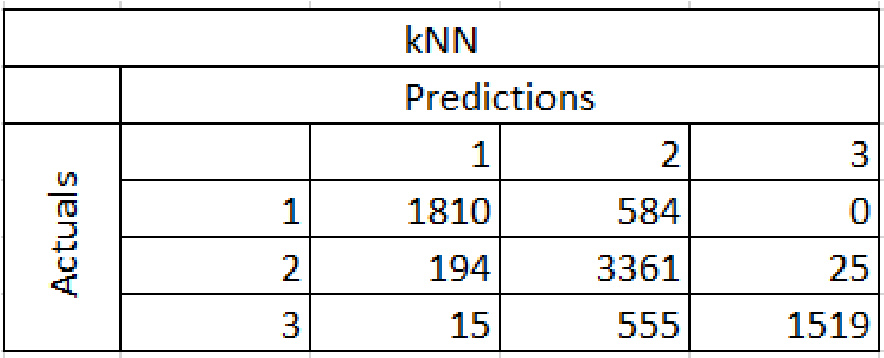
K-nearest neighbors also know as lazy learner, in an algorithm where all the cases are stored, and it will classify new cases. We decided to use this technique because it is non-parametric, which means that it will not make any assumptions on the underlying data distributions, i.e. the model structure is determined from the data. We used cross-validation to determine a good K value, by using an independent dataset to validate the K value.

Graph 4:



Our model gave the value for k=33 (Refer to Graph 4).

Table 1: kNN accuracy



kNN				
	Predictions			
Actuals		1	2	3
	1	1810	584	0
	2	194	3361	25
	3	15	555	1519

This is assumed to be a good value as it reduces the overall noise. This resulted in 85% of accuracy (Table 1).

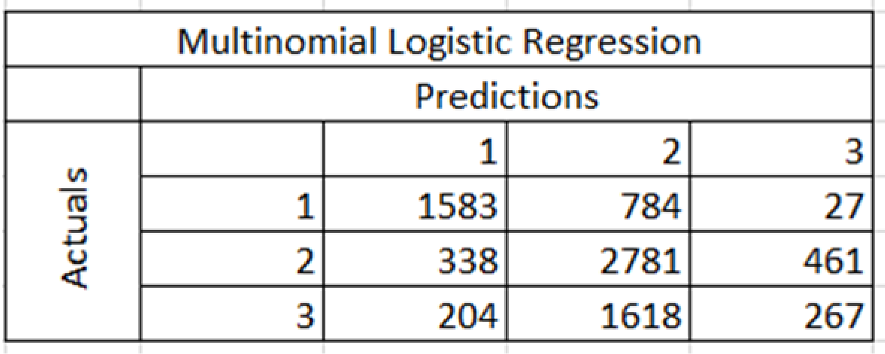
MULTINOMIAL LOGISTICS REGRESSION

Multinomial logistic regression is the regression analysis used when the dependent variable is nominal with more than two levels. Similar to logistic regression, multinomial logistic regression has advantages such as being easy to interpret, robust to noise, feature correlation and variance in feature scale. MLG also oftentimes takes less training time than SVM and kNN models.

For this problem, we tried MLG with both the full dataset (58 features) and the reduced dataset after MRMR feature selection (45 features left). The final result is 59% of accuracy for both methods. This means that feature selection doesn't affect MLG.

For the reduced dataset, MLG's performance is given as follows:

Table 2: Multinomial Logistic Regression Accuracy



Multinomial Logistic Regression				
	Predictions			
Actuals		1	2	3
	1	1583	784	27
	2	338	2781	461
	3	204	1618	267

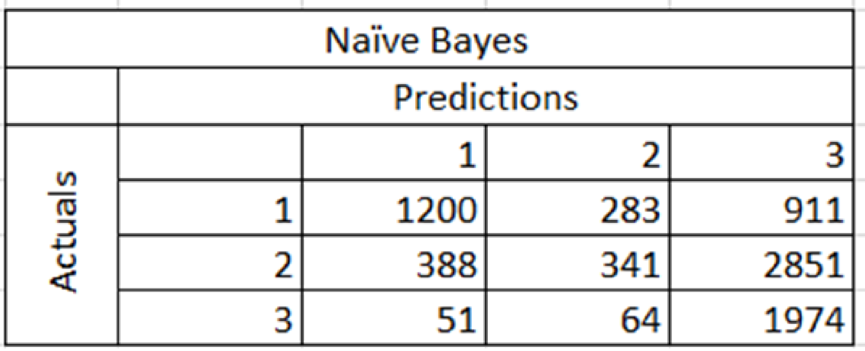
Compared with other models, MLR cannot be considered a good classifier, but if we use the majority class (damage level 2) for a baseline model, which would score 44% accuracy, MLR is still performing much better than guessing on a frequent value.

NAÏVE BAYES

Naive Bayes is a simple model that could be useful on small dataset with short training time. Naive Bayes assumes that features are independent from each other, in which case the algorithm could beat Logistic Regression with regard to performance. However, this is rare in practice.

Another point to consider about Naive Bayes is that it usually performs better on categorical data rather than continuous data. For this project, Naive Bayes is not an effective approach with a very high error rate (0.44 for the full dataset and 0.45 for reduced dataset after feature selection.

Table 3: Naïve Bayes Accuracy

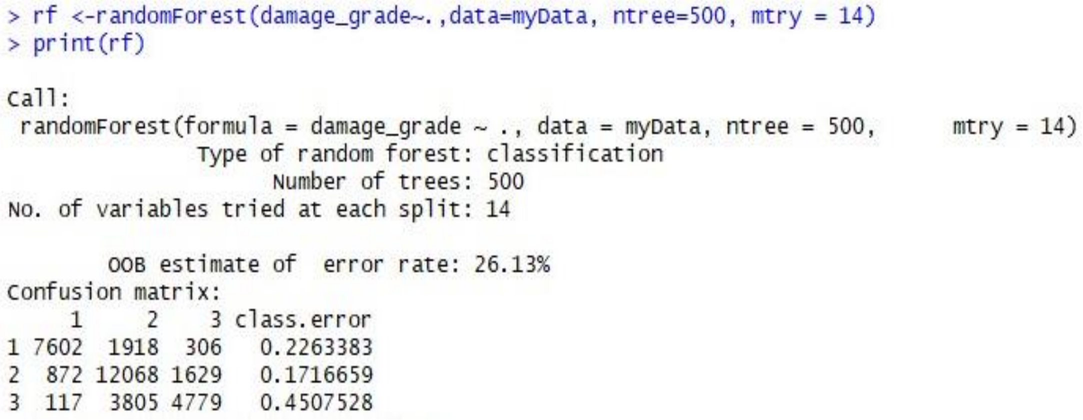


Naïve Bayes				
	Predictions			
Actuals		1	2	3
	1	1200	283	911
	2	388	341	2851
	3	51	64	1974

RANDOM FOREST

In Random Forests the idea is to decorrelate the several trees which are generated by the different bootstrapped samples from training Data. And then we simply reduce the Variance in the Trees by averaging them. This helps us to reduce the variance and also improve the Performance of Decision Trees on Test Set and eventually avoid Overfitting. The idea is to build lots of Trees in such a way to make the Correlation between the Trees smaller.

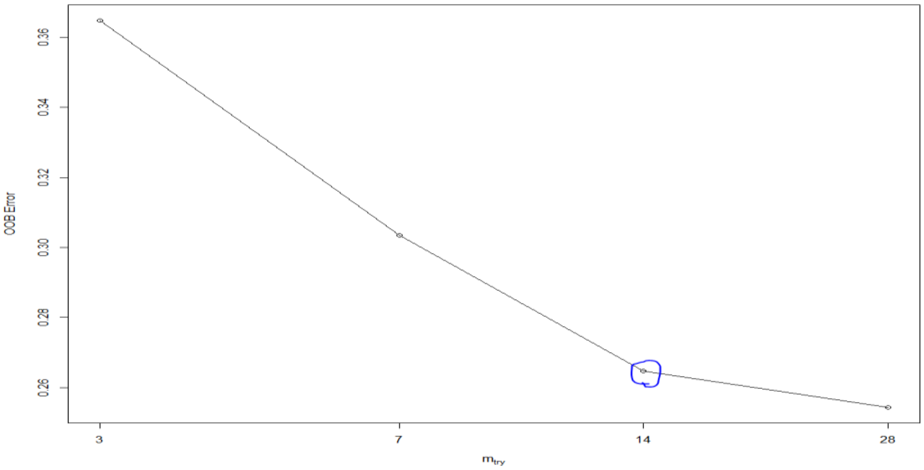
Image 3:



```
> rf <-randomForest(damage_grade~.,data=myData, ntree=500, mtry = 14)
> print(rf)

Call:
 randomForest(formula = damage_grade ~ ., data = myData, ntree = 500,      mtry = 14)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 14

        OOB estimate of  error rate: 26.13%
Confusion matrix:
     1     2    3 class.error
1 7602  1918  306   0.2263383
2  872 12068 1629   0.1716659
3  117  3805 4779   0.4507528
```

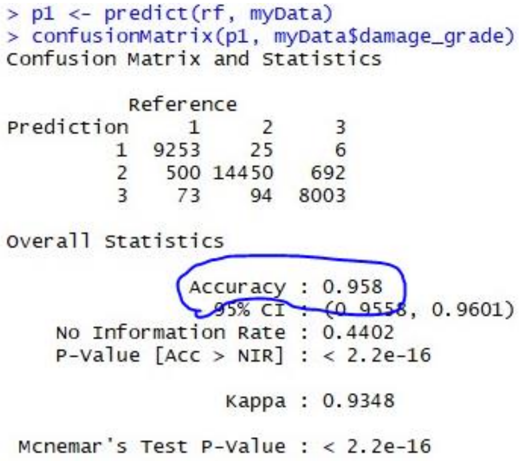
The number of variables randomly selected at each split is 14.

Graph 4: Error graph



This plot shows the Error and the Number of Trees. We can easily notice that how the Error is dropping as we keep on adding more and more trees and average them.

Image 4:

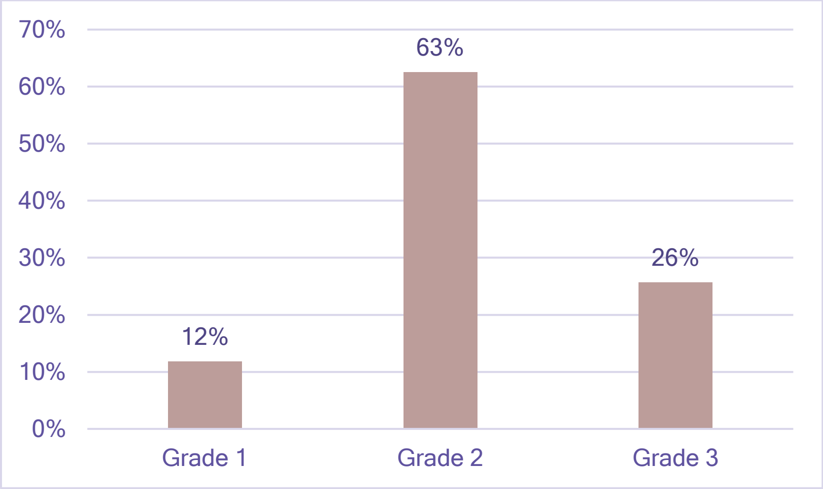


```
> p1 <- predict(rf, myData)
> confusionMatrix(p1, myData$damage_grade)
Confusion Matrix and Statistics

          Reference
Prediction     1     2     3
         1  9253    25     6
         2   500 14450   692
         3    73    94  8003

Overall Statistics

               Accuracy : 0.958
                 95% CI : (0.9558, 0.9601)
    No Information Rate : 0.4402
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9348

 Mcnemar's Test P-Value : < 2.2e-16
```

CLASSIFICATION RESULTS

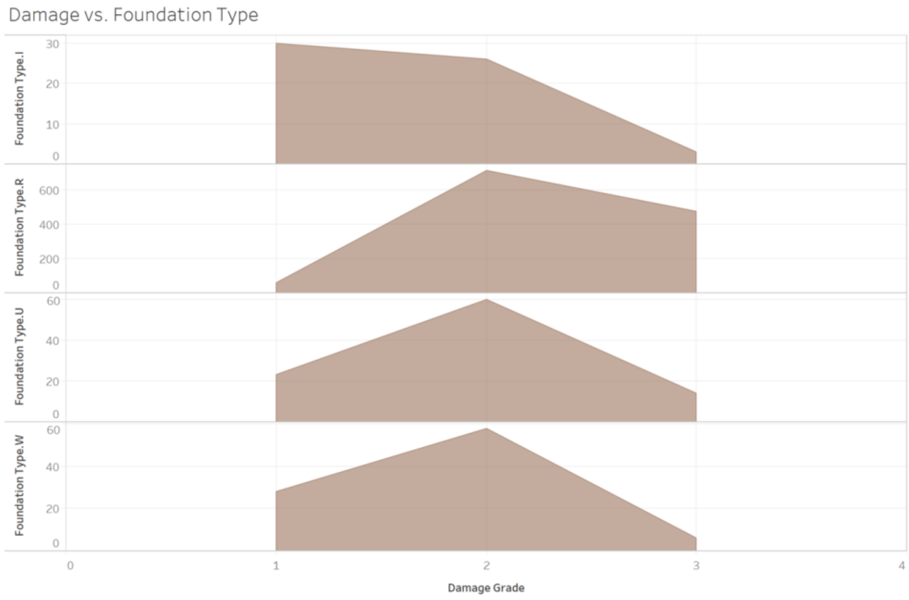
The damage grade was classified as 1 (Low damage) ,2 (Medium damage), and 3 (Almost complete destruction). Out of the 11 most affected districts, districts from the remote rural areas were affected the most, namely Nuwakot and Dolakha. Most of the damaged buildings around 63% were classified as grade 2 damage, i.e. medium damage, fullered by grade 3 damage i.e. complete destruction with 26% of the buildings and lastly 12% of the buildings were classified as grade 1 damage i.e. low damage.

Graph 5: Building damage Classification



The top two variables that contributed the most towards the damage were Foundation type and Ground Floor type.

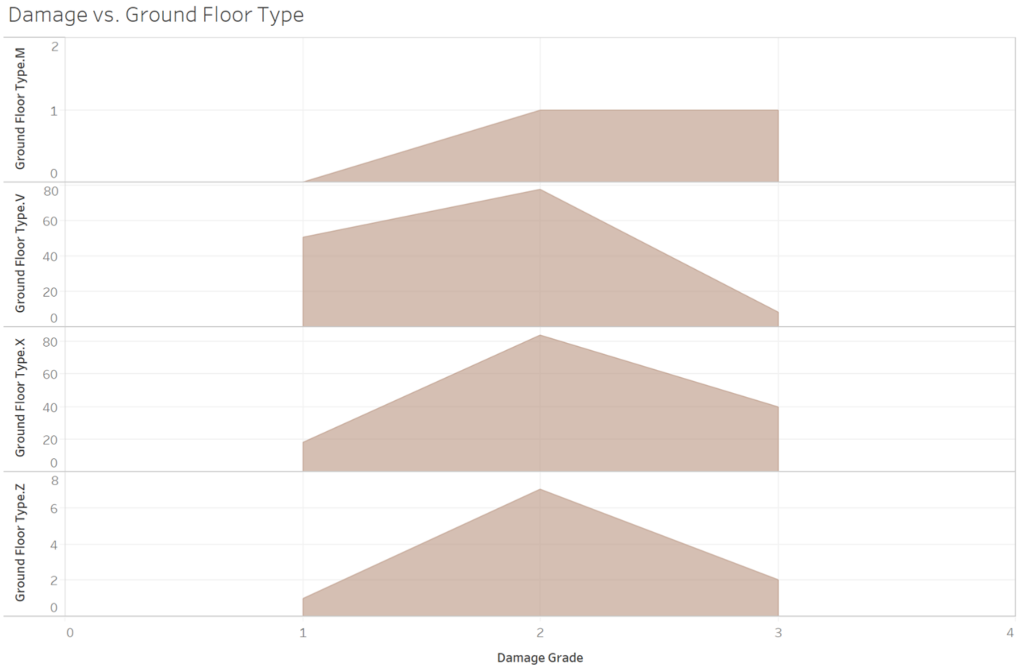
Graph 6: Damage Grade Type vs. Foundation type



The plots of sum of Foundation Type.I, sum of Foundation Type.R, sum of Foundation Type.U and sum of Foundation Type.W for Damage Grade.

Foundation type was divided into 4 categories, namely type W: Pillared, type U: Wood/Bamboo, type R: Mud mortar, and type I: Other. Most of the buildings built with type R: Mud mortar were the most damaged, mainly grade 2 and 3 damage.

Graph 7: Damage Grade Type vs. Ground Floor Type

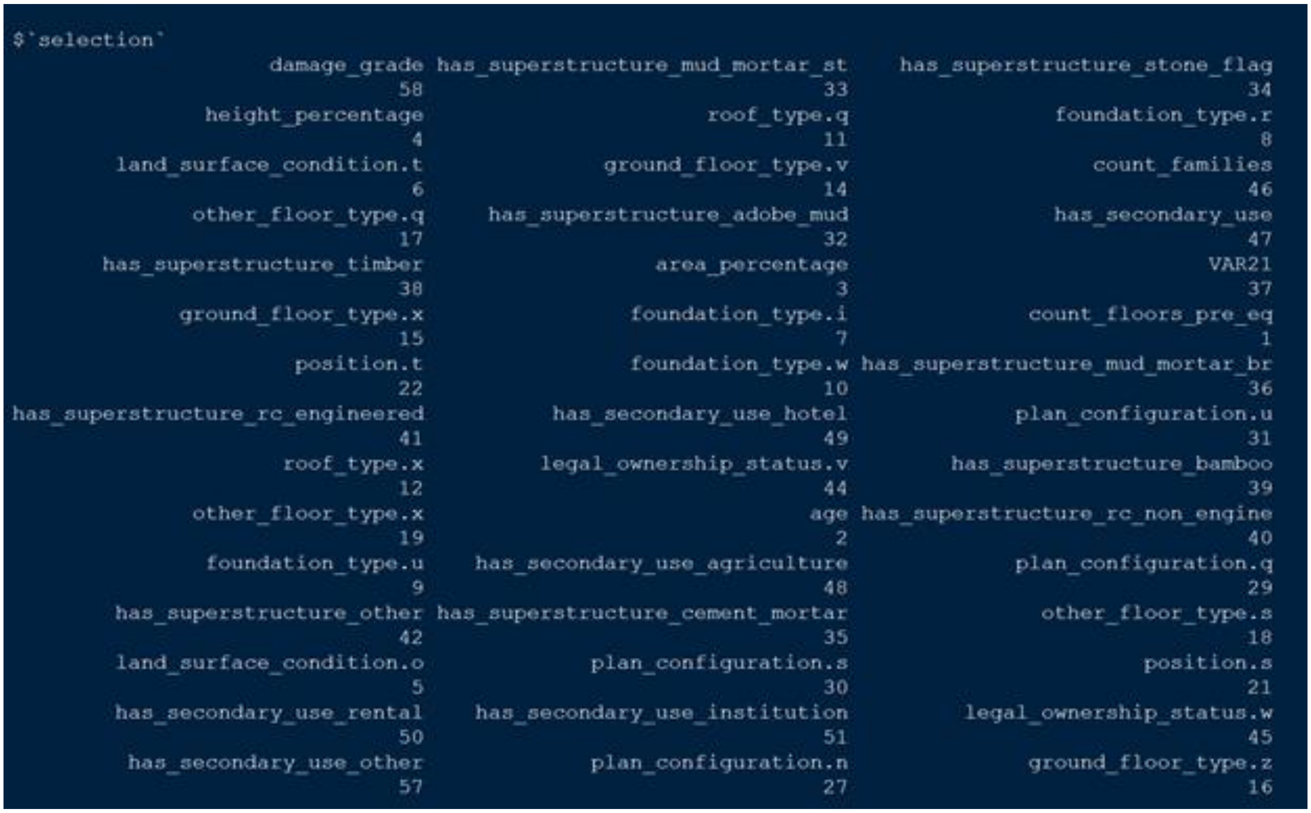


The plots of sum of Ground Floor Type.M, sum of Ground Floor Type.V, sum of Ground Floor Type.X and sum of Ground Floor Type.Z for Damage Grade.

Ground Floor type was categorized into 4 types, namely type X: Concrete, type V: Wood, type M: Brick / Stone, type Z: Other. Surprisingly, buildings built on concrete faced more damage than buildings built on wood. Such points can be noted for and by future architects when reconstructing Nepal or buildings in countries similar to Nepal in terms of the land surface.

APPENDIX I:

A. Variable Selection: *Maximum Relevance Minimum Redundancy*



```
$`selection`
                    damage_grade has_superstructure_mud_mortar_st    has_superstructure_stone_flag
                              58                               33                               34
               height_percentage                      roof_type.q                foundation_type.r
                               4                               11                                8
        land_surface_condition.t              ground_floor_type.v                   count_families
                               6                               14                               46
              other_floor_type.q     has_superstructure_adobe_mud                has_secondary_use
                              17                               32                               47
       has_superstructure_timber                  area_percentage                            VAR21
                              38                                3                               37
             ground_floor_type.x                foundation_type.i              count_floors_pre_eq
                              15                                7                                1
                      position.t                foundation_type.w has_superstructure_mud_mortar_br
                              22                               10                               36
has_superstructure_rc_engineered          has_secondary_use_hotel             plan_configuration.u
                              41                               49                               31
                     roof_type.x         legal_ownership_status.v        has_superstructure_bamboo
                              12                               44                               39
              other_floor_type.x                              age has_superstructure_rc_non_engine
                              19                                2                               40
               foundation_type.u    has_secondary_use_agriculture             plan_configuration.q
                               9                               48                               29
        has_superstructure_other has_superstructure_cement_mortar               other_floor_type.s
                              42                               35                               18
        land_surface_condition.o             plan_configuration.s                       position.s
                               5                               30                               21
        has_secondary_use_rental    has_secondary_use_institution         legal_ownership_status.w
                              50                               51                               45
         has_secondary_use_other             plan_configuration.n              ground_floor_type.z
                              57                               27                               16
```

B. Variable Selection: Boruta

The significant variables in our case are:
count_floors_pre_eq, age+area_percentage, height_percentage, has_superstructure_adobe_mud, has_superstructure_mud_mortar_stone, has_superstructure_stone_flag, has_superstructure_cement_mortar_stone, has_superstructure_mud_mortar_brick, has_superstructure_cement_mortar_brick, has_superstructure_timber, has_superstructure_bamboo, has_superstructure_rc_engineered, count_families, has_secondary_use, has_secondary_use_agriculture, land_surface_condition.t, foundation_type.i, foundation_type.r, foundation_type.u, foundation_type.w, roof_type.q, roof_type.x, ground_floor_type.v, other_floor_type.q, other_floor_type.s, other_floor_type.x, position.o+position.s, position.t, plan_configuration.d, plan_configuration.q, plan_configuration.u, legal_ownership_status.v, legal_ownership_status.w

C. Variable Selection: Stepwise

The 33 significant variables are:

```
[1] "(Intercept)"
[2] "X"
[3] "count_floors_pre_eq"
```

```
 [4] "has_superstructure_adobe_mud"
 [5] "has_superstructure_mud_mortar_stone"
 [6] "has_superstructure_stone_flag"
 [7] "has_superstructure_cement_mortar_stone"
 [8] "has_superstructure_mud_mortar_brick"
 [9] "has_superstructure_cement_mortar_brick"
[10] "has_superstructure_timber"
[11] "has_superstructure_rc_engineered"
[12] "has_secondary_use"
[13] "has_secondary_use_agriculture"
[14] "has_secondary_use_hotel"
[15] "has_secondary_use_institution"
[16] "has_secondary_use_industry"
[17] "has_secondary_use_gov_office"
[18] "has_secondary_use_use_police"
[19] "land_surface_condition.o"
[20] "land_surface_condition.t"
[21] "foundation_type.r"
[22] "foundation_type.u"
[23] "roof_type.q"
[24] "ground_floor_type.v"
[25] "ground_floor_type.x"
[26] "position.s"
[27] "position.t"
[28] "plan_configuration.c"
[29] "plan_configuration.d"
[30] "plan_configuration.f"
[31] "plan_configuration.m"
[32] "plan_configuration.n"
[33] "plan_configuration.q"
[34] "plan_configuration.s"
[35] "plan_configuration.u"
```

CONTRIBUTION

Apurva Tawde: Data collection, cleaning, Feature Selection, Modelling, Classification results, Presentation and Report formatting

Michelle Do: Data collection, cleaning, Feature Selection, Modelling, Classification results, Presentation and Report formatting

Yagna Ganesh Easwaran: Data collection, cleaning, Feature Selection, Modelling, Classification results, Presentation and Report formatting

Khanjan Patel: Data collection, cleaning, Feature Selection, Modelling, Classification results, Presentation and Report formatting

Jinesha Jain: Data collection, cleaning, Feature Selection, Modelling, Classification results, Presentation and Report formatting