Generalized post-disaster structural assessment

Abhijit Mahapatra & Yutae Lee

Overview

- This project is aimed to provide a generalized machine learning model that predicts post-disaster building condition/damage level from 0-4 by the structure and material that was used to build the building.
- The implication of the study would be easier to notify the people who live in the area where disaster strikes, how severe the storm will impact their houses and if they need to evacuate before the wind comes.

Image courtesy: Google images

Data Description

Irma(2017)

Data v1.0 Dimensions
HI-DA.csv
Row - 2212
Column - 65
Key Features:
- Status
- Number of stories
- Age
- first floor elevation
- Wall cladding
- Roof cover

Dorian(2020)

Data v2.0 Dimensions
Dorian_PA_Final.Xlsx
Row - 1094
Column – 90
Features:
- Status
- Number of stories
- Age
- Wall cladding
- Roof material
- Roof system
- Roof cover
- Foundation type
- Wall structure
- Soffit type
- large door present

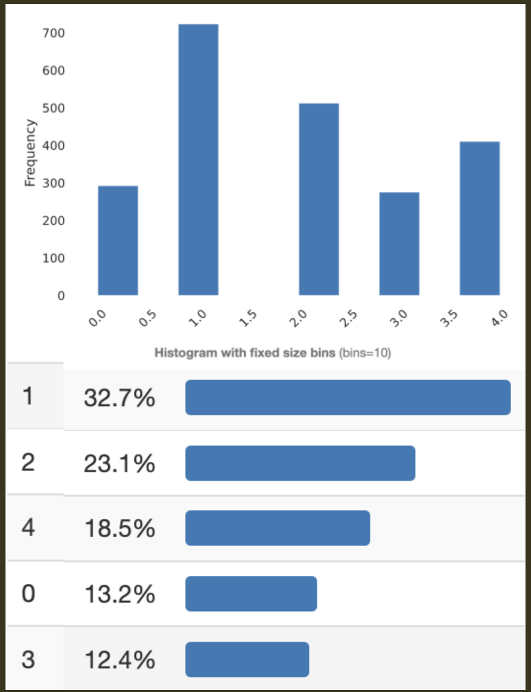
Michael(2019)

Data v2.0 Dimensions
Hurricane_Michael.csv
Row - 736
Column – 40
Features:
- Status
- Number of stories
- Age
- Wall cladding
- Roof material
- Roof system
- Roof cover
- Foundation type
- Wall structure
- Soffit type
- large door present

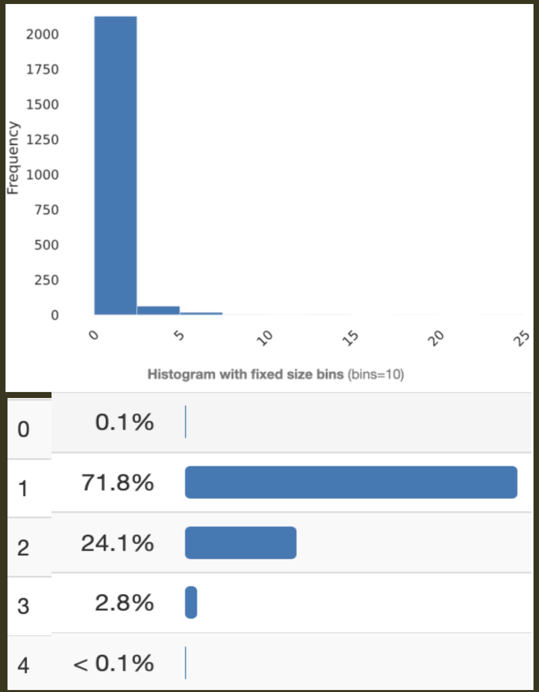
Primary Challenges

- Missing data for multiple columns
- Column discrepancy over datasets (Irma vs Dorian&Michael)
- Mislabeled data classes (-1s)
- Skewed class wise data distribution

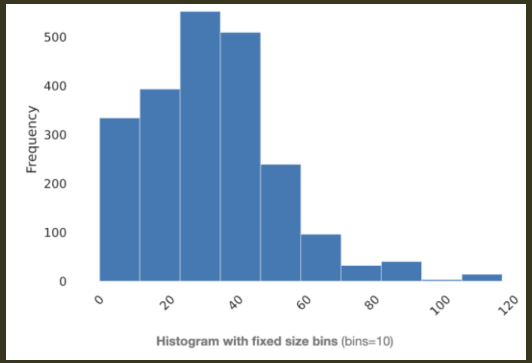
Exploratory data analysis



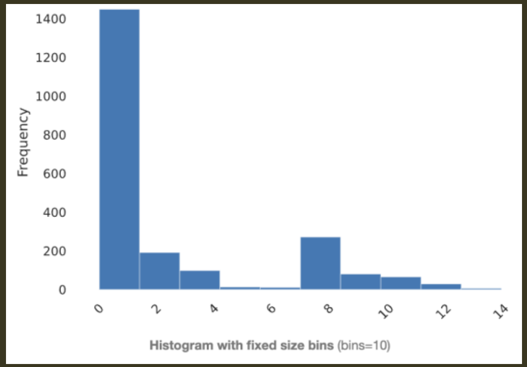
Status



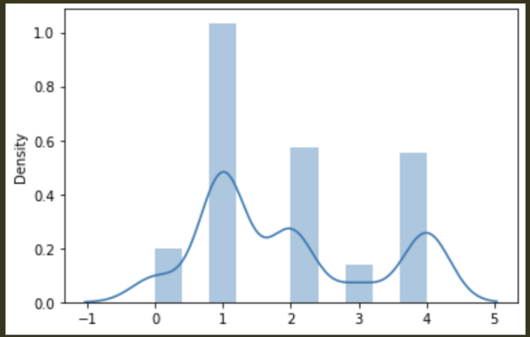
stories



Age



FF Elevation



Target variable density

Implementation

- Baseline model implementation approach – AutoML Using PyCaret Classification
- Deep learning Implementation approach – TensorFlow
- Ensemble Machine learning Implementation - XGBoost

Data Preprocessing

- In order to implement any machine learning technique, we had to clean the data for data processing. Which includes:
- *Column rename for consistency and age renumeration. (Mainly Hurricane Irma)*
- *Missing value imputation – Forward/backward fill and KNN imputation(age).*
- *Textual data to integer labelling using custom encoding*

ex: Asphalt shingles (laminated),"Metal, corrugated" - 13

"Roof Diaphragm, wood","Roof Diaphragm, steel", "Wall Diaphragm, concrete" → Bucket: Wood → Encoding 2

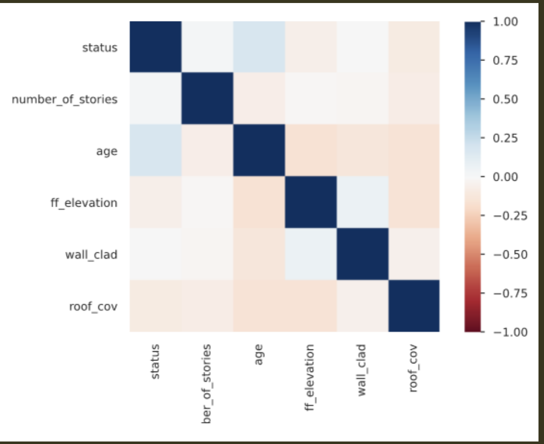
- Bucket for wall elements to reduce the class types from 101 to 5

Example - "Roof Diaphragm, wood", "Wall Diaphragm, masonry" - Wood then label encoding to 2

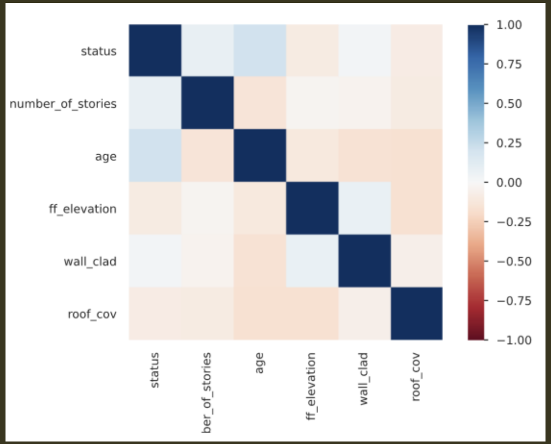
- Scaling was done to standardize the data.

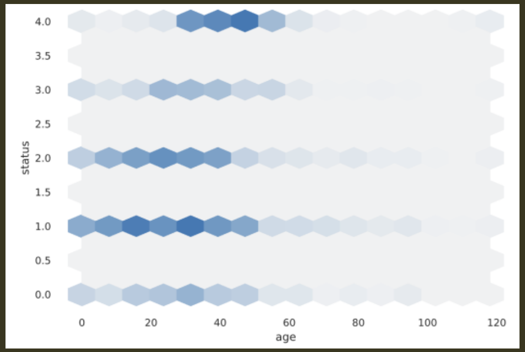
Feature selection v1.0

Pearson coefficient

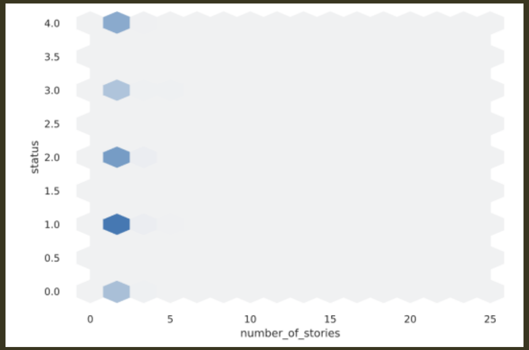


Spearman's coefficient

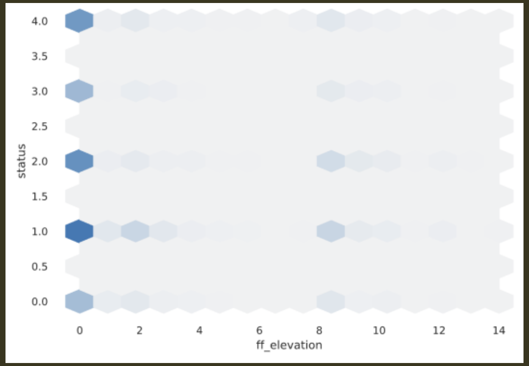




Status vs Age



Status vs # stories



Status vs FF elevation

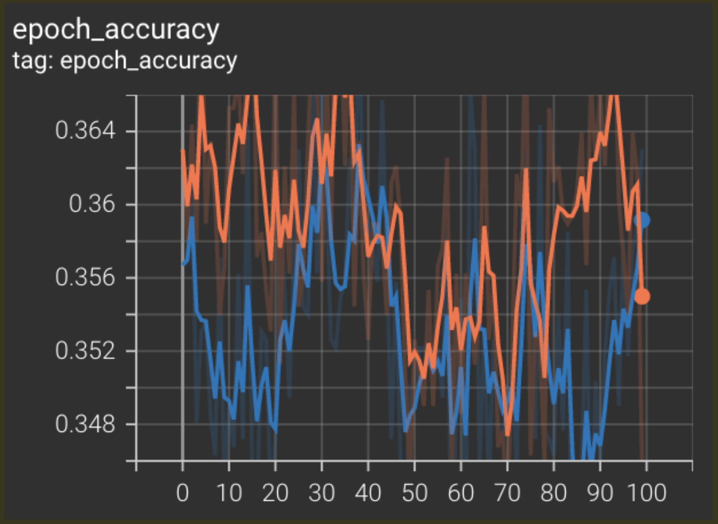
Baseline model implementation

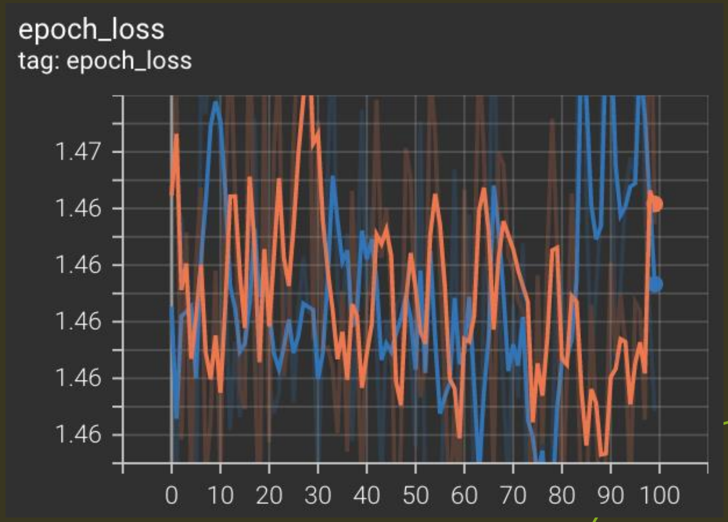
- Using AutoML from PyCaret we tried to find out the baseline performance

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
lightgbm	Light Gradient Boosting Machine	0.4322	0.7147	0.3984	0.4245	0.4238	0.2499	0.2516	0.143
rf	Random Forest Classifier	0.4212	0.7071	0.3985	0.4166	0.4148	0.2404	0.2417	0.223
et	Extra Trees Classifier	0.4186	0.6815	0.4027	0.4162	0.4138	0.2410	0.2420	0.221
gbc	Gradient Boosting Classifier	0.4148	0.6852	0.3505	0.4089	0.3819	0.2027	0.2120	0.349
knn	K Neighbors Classifier	0.4044	0.6625	0.3575	0.4020	0.3932	0.2056	0.2089	0.052

Deep learning Implementation

- We have developed a neural network classifier and trained it for a multiclass classifier.
- 8-layer network, with Adam optimizer and learning rate 0.001
- With early stopping and regularization, on validation/test set observed about 40% accuracy for the given data.





Ensemble Machine learning Implementation

- After getting unsatisfactory results in our previous attempts, we took a pivot from the direction and implemented a simple ensemble model using XGBoost as it handles multi class classification really well.
- As a result, we found better overall classification than the previous performance.
- We recorded 47.2 % accuracy and here is the confusion matrix.

	precision	recall	f1-score	support
0	0.40	0.21	0.27	58
1	0.52	0.64	0.57	152
2	0.35	0.29	0.32	100
3	0.57	0.21	0.30	58
4	0.49	0.77	0.60	75
accuracy			0.47	443
macro avg	0.46	0.42	0.41	443
weighted avg	0.47	0.47	0.45	443

Class-wise performance

```
Accuracy: 47.18%
[[12 11  6  1  0]
 [29 98 42 13  8]
 [ 9 25 29 13  8]
 [ 1  3  4 12  1]
 [ 7 15 19 19 58]]
```

PCA + XGboost

- Although implementing XGboost gave us a meaningful improvement in accuracy, we wanted to develop farther by using XGboost model after reducing dimension of the data by Principal Component Analysis.
- We reduced the dimension from 6 to 4 and got a minor improvement in accuracy, where we cannot confidently say that using PCA will help us get better accuracy.
- Overall, the model recorded 47.4% accuracy and here is the confusion matrix along with it.

```
Accuracy: 47.40%
[[16 13  4  1  1]
 [27 91 43 16 17]
 [ 7 27 37 11  7]
 [ 1  3  2 17  1]
 [ 7 18 14 13 49]]
```

Class wise:

	precision	recall	f1-score	support
0	0.46	0.28	0.34	58
1	0.47	0.60	0.53	152
2	0.42	0.37	0.39	100
3	0.71	0.29	0.41	58
4	0.49	0.65	0.56	75
accuracy			0.47	443
macro avg	0.51	0.44	0.45	443
weighted avg	0.49	0.47	0.46	443

Implementation v2.0

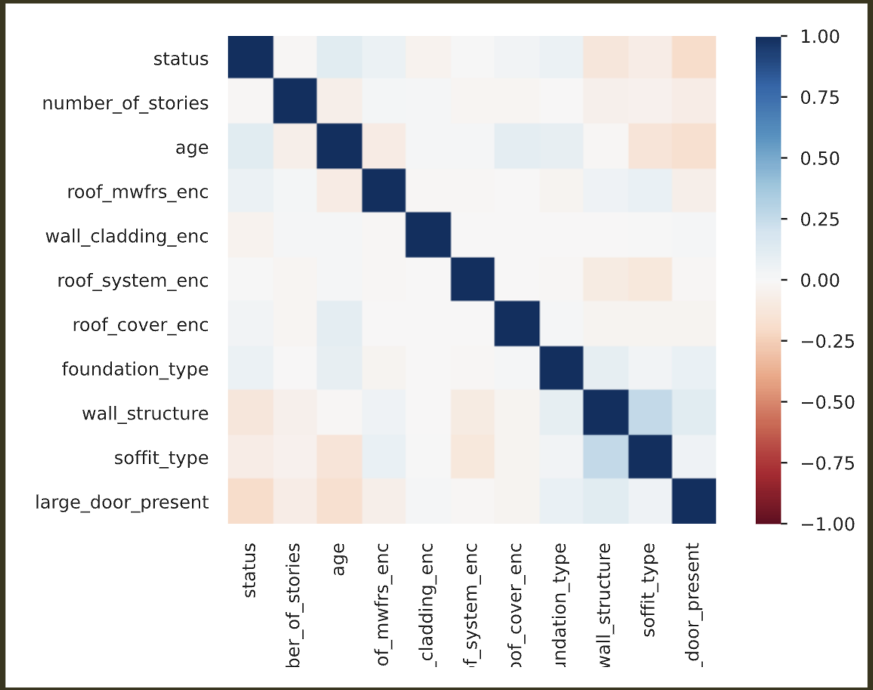
- As we learned the data collected in Irma and Dorian & Michael has significant difference in data collection, we took the later for our analysis to create a better model in terms of classification.
- So, we took another 11 features from Dorian and Michael (given on the first slide)
- We didn't get much higher accuracy but at least better than the last one, so we moved to the XGBoost classifier with tuning params:
    - N_estimator-600
    - Nthreads-3
    - Booster-"GBTREE

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
rf	Random Forest Classifier	0.4340	0.7100	0.3749	0.4289	0.4119	0.2568	0.2622	0.211
gbc	Gradient Boosting Classifier	0.4274	0.7056	0.3539	0.4048	0.3979	0.2438	0.2492	0.284
lightgbm	Light Gradient Boosting Machine	0.4208	0.7234	0.3678	0.4058	0.4074	0.2456	0.2474	0.130
lr	Logistic Regression	0.3961	0.6861	0.3262	0.3570	0.3643	0.2008	0.2052	0.530
ridge	Ridge Classifier	0.3896	0.0000	0.3161	0.3439	0.3553	0.1914	0.1961	0.008

Ensemble XGboost v2.0

- For the version 2.0 we took 11 features from Dorian and Michael dataset, for trying out any comparative model performance.
- Surprisingly, accuracy increased more than what we had earlier. Here is snippet of new confusion matrix

```
Accuracy: 52.05%
[[ 5  1  0  0  1]
 [ 8 26 17  5  3]
 [ 3 16 25 10  6]
 [ 1  1  3 15  6]
 [ 9  2  6  7 43]]
```



PCA + XGboost v2.0

- Similar to v1.0, we wanted to see if reducing the dimension using Principal Component Analysis could help us getting better model.
- This time we reduced dimension from 11 to 8. But the accuracy of the model was worse than the accuracy of the model when we are not using PCA.

```
Accuracy: 48.40%
[[ 4  0  0  0  1]
 [10 23 11  7  4]
 [ 4 11 23 12  6]
 [ 1  4  5 13  5]
 [ 7  8 12  5 43]]
```

Cross Class adjustments

- As we didn't get a very high accuracy, after consulting with sponsor and our guide, we did a cross class analysis for the prediction and ground truth.
- In short:
  - Adding +1 and -1 with the prediction class, as we observed lot of close misclassification.
  - Create a set and tried to search the prediction class is in there, if yes increase count, if not add nothing.
  - In the end calculate the (matches/total rows) x 100

By this we achieved **91.13%** accuracy.

```
(cnt/len(df))*100
```

```
91.13345521023766
```

Observation

- From our experiment, we were able to see that XGboost was the best model that turned out to have highest accuracy.
- Class 1 and 4 scored higher accuracy than other classes, perhaps this maybe caused by the ambiguity of class 2 and 3 and low support of class 0.
- Our model can successfully predict the range of true classes with high accuracy; however, it cannot predict the exact class with the same accuracy.
- Here is a snippet of our analysis on -1 and +1 prediction and how/why it helps to increase the accuracy.

```python
X = df[['status']]
Y = df[['list_plus']]

# split data into train and test sets
seed = 8
test_size = 0.2
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=test_size, random_state=seed)
print(classification_report(y_test, X_test))
```

```
              precision    recall  f1-score   support

         0.0       0.19      0.83      0.31         6
         1.0       0.59      0.63      0.61        43
         2.0       0.82      0.63      0.71        67
         3.0       0.68      0.83      0.75        30
         4.0       0.83      0.67      0.74        73

    accuracy                           0.68       219
   macro avg       0.62      0.72      0.62       219
weighted avg       0.74      0.68      0.70       219
```

```python
X = df[['status']]
Y = df[['list_minus']]

# split data into train and test sets
seed = 8
test_size = 0.2
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=test_size, random_state=seed)
print(classification_report(y_test, X_test))
```

```
              precision    recall  f1-score   support

         0.0       0.50      0.87      0.63        15
         1.0       0.91      0.63      0.74        67
         2.0       0.55      0.60      0.57        47
         3.0       0.59      0.73      0.66        30
         4.0       0.73      0.72      0.72        60

    accuracy                           0.68       219
   macro avg       0.66      0.71      0.67       219
weighted avg       0.71      0.68      0.68       219
```

Recommendations

- The status after the disaster were assessed by a human, which means that there are great possibility of bias and ambiguity in class labelling.
- Thus, to reduce bias and ambiguity, decreasing number of target class could help the model to performance of the prediction.
- Also, unifying the columns between datasets would help to increase the number of features that could studied.
- Lastly, if there were more rows and data, a more sophisticated deep learning model could be implemented.

Future work

- Time series of wind data and its dynamic impact tracking
- Image processing for community detection of hazard impacts

Thank you