Emmet Houghton
Professor Alex Wong
CPSC 381/581: Machine Learning
Final Project Report

Forecasting Catastrophe Risk: FEMA National Flood Insurance Program

I. Introduction

The central aim of this project is the application of machine learning techniques to forecasting insurance claims against policies issued by the Federal Emergency Management Agency's (FEMA) National Flood Insurance Program (NFIP). In developing models for predicting the costs of catastrophic weather events, this work aims to support FEMA's efforts in managing and mitigating the effects of flood disasters on a national scale.

The models presented here were trained on a synthesis of numerous publicly available datasets containing information about more than 2.6 million individual flood insurance policies which filed for claims between 1977 and 2024, as well as geographic information about the insured properties and meteorological information about the storms which caused flooding. Due to the overwhelming volume of the complete dataset, the models presented here focus on significant tropical storms of category 1 or greater which affected properties in Florida, a state which is prone to catastrophic weather events and storm surges. This subset of the data constitutes roughly 200,000 samples. Other than the size of the data, other challenges faced included the integration of disparate data sources as well as a pronounced skew in the labels representing claim amounts. An array of statistical techniques including principal component analysis and stochastic sampling were employed in overcoming these technical challenges during model development.

The forecasting of flood insurance claims is a task of critical importance to the government. Since its inception in 1968, the NFIP has grown to protect over 22,648 communities across the United States, forming an essential component of the national strategy to diminish the adverse economic impacts of flooding. Despite its significant role, the NFIP has struggled with financial sustainability, largely due to the accumulation of debt in the wake of major disasters such as Hurricane Katrina in 2005. As of October 2023, the program faced a daunting debt load of $20.5 billion, necessitating FEMA to manage over $300 million in annual interest payments and severely restricting the program's capacity to serve its beneficiaries effectively. A detailed understanding of national flood risk empowers FEMA to allocate resources more strategically for damage mitigation and to negotiate better terms in the reinsurance and catastrophe bond markets, reducing the cost of transferring catastrophic weather risks to investors.

Ultimately, for the NFIP to continue its mission of reducing the financial impact of floods on individuals and communities, it is imperative that FEMA employs accurate and comprehensive methods for quantifying risk exposure. This project's advanced analytical approach aims to refine these methods, enhancing the financial integrity of the program and ensuring it can fulfill its promise of protection and recovery support to those who depend on it.

II. Data and Codebase

Data for this project was aggregated from a number of sources to provide a holistic overview of the risk a property poses to the National Flood Insurance Program in the face of a catastrophic storm. First, the OpenFEMA Dataset: FIMA NFIP Redacted Claims (Version 2), which can be accessed directly from the agency's website, provides detailed information about the 2.6 million properties which sustained flood damage and made claims between 1977 and 2024.

Elevation information is missing for most properties in the FEMA Redacted Claims dataset. Fortunately, however, the dataset does contain approximate coordinates in latitude and longitude for each property, which allowed for the querying of topological information from the US Geological Survey (USGS) API through their Bulk Point Query service.

Finally, information about the magnitude of storms which caused the flooding of each property was obtained from the Emergency Events Database (EM-DAT), which was established under the support of the World Health Organization (WHO) and the Belgian Government in 1988. Since 1999, EM-DAT has been supported by the Bureau for Humanitarian Assistance within the United States Agency for International Development.

All datasets can be retrieved from the following endpoints:

1. OpenFEMA Dataset: FIMA NFIP Redacted Claims (Version 2) https://www.fema.gov/openfema-data-page/fima-nfip-redacted-claims-v2.

2. US Geological Survey Elevation API: https://apps.nationalmap.gov/epqs/

3. Emergency Events Database (EM-DAT) Public Table: https://doc.emdat.be/docs/data-structure-and-content/emdat-public-table/

The codebase for this project is available on GitHub: https://github.com/emmethoughton/ml-fema-final-proj/tree/main

III. Methodology

Various approaches were experimented with for fitting the data. Here, three of these modeling schemes and their associated labeling methods are presented. The relative efficacy of each approach will be discussed in section V.

III.I. Dense Neural Network

A significant challenge to fitting the dataset is the extreme skew of the number of dollars paid to policyholders. While most policyholders received very small quantities of relief from the NFIP, others received more than $300,000 to repair the flood damage on their properties. Multiple transformations were applied to these values in order to maximize their meaning for machine learning. For training the neural network, each insurance policy was labeled with the proportion of the total coverage on the property which was paid to the policyholder:

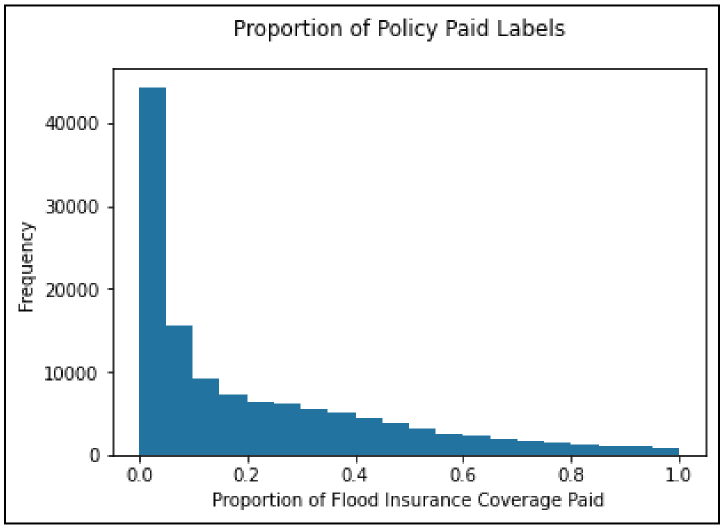


Figure 1. *Labeling scheme defined by proportion of policy paid to policyholders in Florida.*

The PyTorch library was used for the construction of the neural network. Various network architectures were experimented with, but the final network was instantiated with 6 hidden layers, the first four of which contain 16 nodes, followed by layers with 8 and 4 nodes respectively. All nodes were activated with the sigmoid function, as the network is estimating a proportion between zero and 1. Choices of hyperparameters are discussed in greater detail in section IV.II.

The network was trained for 35 epochs in batches of 32 samples with mean squared error loss and PyTorch's Stochastic Gradient Descent optimizer, and the model with the best validation MSE over the training cycles was restored at the end of training.

III.II. Polynomial Expansion, PCA, and Linear Regression

The neural network struggled to escape local minima during training, suggesting that a model of less complexity might perform better for fitting the dataset. As such, a Linear Regression model was next employed for forecasting losses.

For the linear regression model, the strongest results were obtained using labels defined by the standardized logarithm of the amount paid to policyholders. The effectiveness of this scheme is sensible, given the logarithm adjusts for the extreme skew of the labels while standardization meaningfully centers these values around zero.

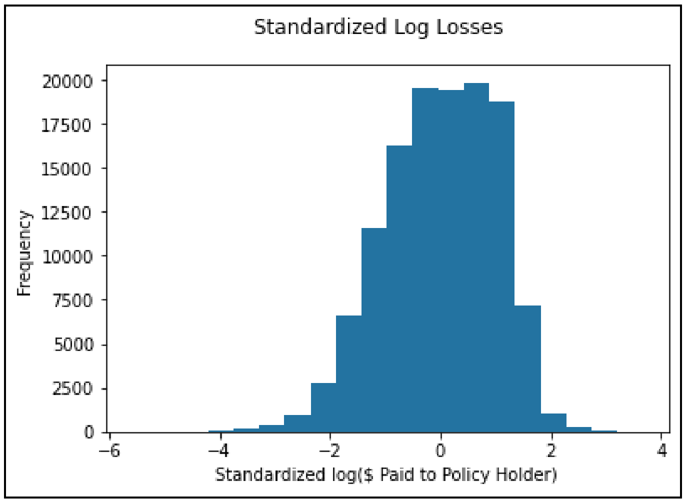


Figure 2. *Labeling Scheme defined by standardized logarithm of amount paid to policyholders in Florida.*

Applying the normal equation directly to the dataset to find optimal weights would not capture non-linear relationships between features. Thus, a degree 2 polynomial expansion was first applied on the dataset using the Sci-Kit Learn library's PolynomialFeatures transformer. Polynomial features with no variance were dropped and the remaining were standardized.

Due to the dimensionality of the original dataset, however, this polynomial transformation process generated a matrix with 2485 columns. To improve the portability of the dataset, principal component analysis (PCA) was performed on the standardized columns to identify the basis vectors which explained the greatest variance of the dataset after polynomial expansion. The principal component analysis procedure returned a set of eigenvectors with the following eigenvalues:

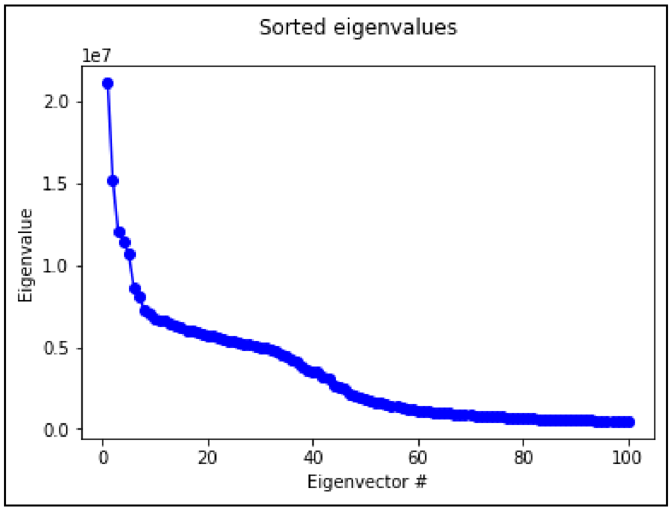


Figure 3. *Sorted eigenvalues of polynomially-expanded dataset obtained from principal component analysis after standardization.*

In order to decide the number of eigenvectors onto which to project the expanded dataset, reconstruction loss in mean squared error was calculated for the top *d* eigenvectors between 5 and 95. These errors are plotted below:

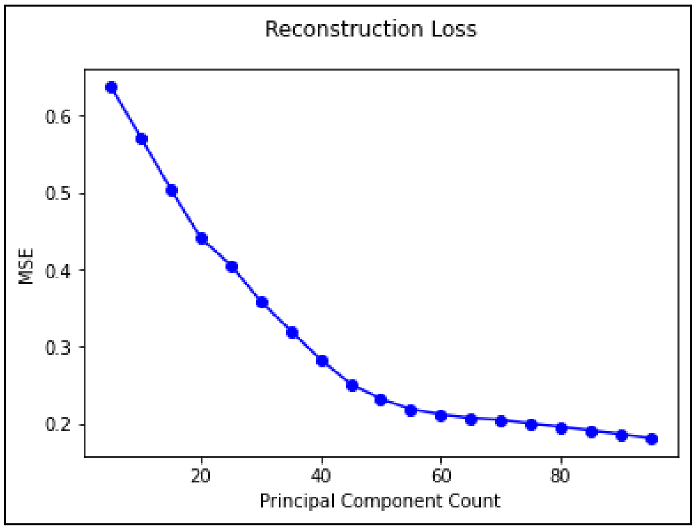


Figure 4. *Reconstruction loss after projecting dataset onto top* d *eigenvectors obtained from PCA.*

We observe diminishing marginal returns to dimensionality beyond approximately 60 eigenvectors. As such, the polynomially expanded dataset was projected onto the 60 eigenvectors with the greatest eigenvalues in preparation for linear regression. SciKit-Learn's LinearRegressor was fit to this resulting projection of the dataset.

III.III. Gradient Boosting Regressor

Finally, in order to capture non-linear relationships in the data more generally, a gradient boosting regressor was fit to the redacted claims. The gradient boosting regressor is a collection of weak decision tree models which collectively form a strong predictive model. Each tree is trained to predict and reduce the residuals, or errors, left by previous trees, and the decision thresholds for splitting the nodes in these trees are chosen to minimize impurity, an information theory metric, across the resulting buckets. The algorithm iteratively adds new trees, each tailored to correct the hardest-to-predict residuals from the prior iteration. Significant transformation of the dataset was not required for training the gradient booster because the algorithm, which simply applies thresholds to the features, is largely unaffected by monotonic transformations.

Several measures were taken in order to prevent overfitting. The maximum depth of each decision tree was limited to 3 and the minimum number of samples in each leaf node was set to 15, preventing the trees from adjusting their predictions at an extremely granular level. Furthermore, the total number of trees sequentially trained was set to 12.

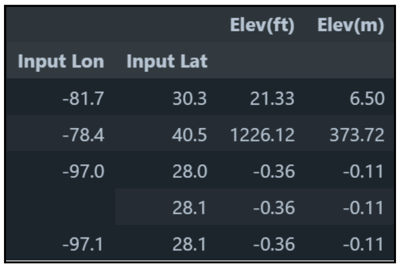
Fitting the gradient boosting regressor proved challenging due to memory constraints. The training dataset had to be reduced to 25% of the redacted claims in the Florida dataset to avoid kernel crashes.

IV Implementation Details

IV.I Data Preprocessing

Data preprocessing proved to be the most computationally expensive step of model development due to the scale of the FEMA redacted claims dataset. The first challenge in data preparation was aggregating data from the FEMA dataset, the US Geographical Survey, and the EM-DAT public table into a single data frame.

The USGS Elevation API was queried to obtain approximate elevation information about each property. The bulk point query service only allowed for 500 coordinate pairs to be submitted at a time through the manual upload of .csv files, so distinct pairs of coordinates were organized and separated into 34 batches to be requested to the API. The results of these queries are included in the GitHub repository for this project in the coordinates directory.

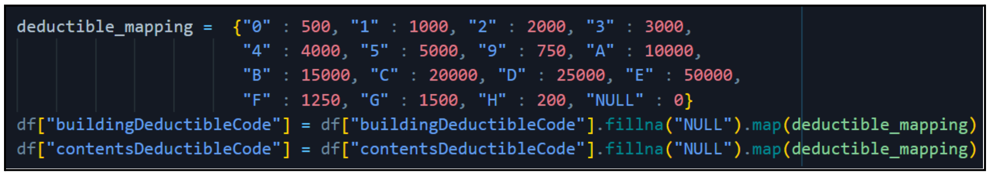


		Elev(ft)	Elev(m)
Input Lon	Input Lat		
-81.7	30.3	21.33	6.50
-78.4	40.5	1226.12	373.72
-97.0	28.0	-0.36	-0.11
	28.1	-0.36	-0.11
-97.1	28.1	-0.36	-0.11

Figure 5. *Header of coordinate-indexed elevation table obtained through USGS.*

The matching of insurance claims to storm magnitudes in the EM-DAT public table was the most expensive step to data preprocessing. Dates were the primary tool used for constructing a robust mapping, but significant padding was necessary to consistently capture the storm of interest, as some claims only provided approximate time information to the nearest month and some storms spanned multiple days. If multiple catastrophic weather events made landfall in the US on a given date, then the storm selected was the storm for which the name provided by FEMA was a substring or an encompassing string of the name provided by EM-DAT. In the case of no results, the magnitude of the storm with the timestamp closest to the timestamp of the flood loss was selected.

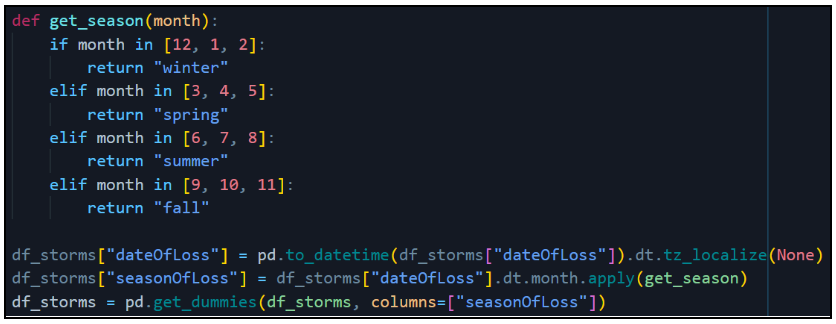
After combining data from these sources, categorical features were transformed into meaningful columns. Exact numerical representations for categorical features such as the building and contents deductible codes were obtained from the FEMA Redacted Claims dataset documentation, while others were converted into indicator columns.



```python
deductible_mapping =  {"0" : 500, "1" : 1000, "2" : 2000, "3" : 3000,
                       "4" : 4000, "5" : 5000, "9" : 750, "A" : 10000,
                       "B" : 15000, "C" : 20000, "D" : 25000, "E" : 50000,
                       "F" : 1250, "G" : 1500, "H" : 200, "NULL" : 0}
df["buildingDeductibleCode"] = df["buildingDeductibleCode"].fillna("NULL").map(deductible_mapping)
df["contentsDeductibleCode"] = df["contentsDeductibleCode"].fillna("NULL").map(deductible_mapping)
```

Figure 6. *Transformation of categorical features using FEMA documentation.*

The dataset was also augmented with a number of features which seemed potentially valuable to learning algorithms. For example, dummy variables representing the season in which the tropical storm occurred were appended to the dataset:



```python
def get_season(month):
    if month in [12, 1, 2]:
        return "winter"
    elif month in [3, 4, 5]:
        return "spring"
    elif month in [6, 7, 8]:
        return "summer"
    elif month in [9, 10, 11]:
        return "fall"

df_storms["dateOfLoss"] = pd.to_datetime(df_storms["dateOfLoss"]).dt.tz_localize(None)
df_storms["seasonOfLoss"] = df_storms["dateOfLoss"].dt.month.apply(get_season)
df_storms = pd.get_dummies(df_storms, columns=["seasonOfLoss"])
```

Figure 7. Augmentation of dataset with seasonal information.

IV.II Hyperparameter Selection

Hyperparameters were tuned through various iterations of training and evaluation to reduce overfitting and maximize model performance. This section summarizes the final hyperparameters and briefly justifies the selection of each.

The neural network was the most difficult model to tune. First, a baseline neural network was trained with various labeling schemes and corresponding activation functions to choose the most robust form for the predictor. For example, untransformed labels were trained with ReLU activation, while the final set of labels representing the proportion of each policy that was paid was trained with sigmoid activation. Network structure was then adjusted. Larger hidden layers of 16 nodes were placed in the first hidden layers, and layers of non-increasing size were placed in subsequent layers in order to bridge the gap between the 69 nodes of the input layer and the single node in the output layer. The network was trained with both mean squared error loss (MSE) and L1 loss as well as stochastic gradient descent (SGD) and Adam optimizers, but in the end MSE loss and the SGD optimizer were chosen because these hyperparameters produced more consistent results across training cycles. Finally, the learning rate was tuned. Larger learning rates ($\sim10^{-3}$) were most successful with the SGD optimizer, while smaller ones ($\sim10^{-5}$) proved most effective for the Adam optimizer. A small batch size of 32 was selected to assist the network in managing the overwhelming scale of the dataset. All trained networks were fully-connected.

Dense Neural Network Hyperparameters:

Shape: (69, 16, 16, 16, 16, 8, 4, 1)
Activation: sigmoid applied to all nodes
Learning Rate: 0.001
Loss: Mean Squared Error
Optimizer: Stochastic Gradient Descent
Batch Size: 32

The gradient boosting regressor was initially trained with 25 estimators and a max depth of 4, which led to significant overfitting. Through tuning, the model size was reduced to ultimately train only 12 decision tree estimators with a max depth of 3. Each leaf node was also restricted to contain at least 15 samples:

Gradient Boosting Regressor Hyperparameters

n_estimators: 12
max_depth: 3
min_samples_leaf: 15

These hyperparameters collectively prevent the gradient booster from creating jagged, granular decision boundaries across the training data and in turn help the model generalize to the validation and testing sets, which was especially critical in this context because memory constraints only permitted the use of 25% of the dataset for training the Gradient Boosting Regressor.

V. Results

Finally, we compare the results of the trained models. After 35 training epochs, the neural network was able to achieve a mean squared error of 0.0597 on the testing set for forecasting the proportion of the total value of the insurance policy which FEMA would ultimately pay to the policyholder due to flood damage caused by a catastrophic storm in Florida. Note that this result is relatively unimpressive, however, as the square root of this MSE is 0.244 and the labels are contained in the range (0, 1]. A mean squared error closer to 0.03 would have been a desirable result. Stronger predictive power was observed in the other trained models.

The linear regression model trained on the quadratically expanded dataset reduced using principal component analysis performed significantly better than the neural network. Recall that labels chosen for this model were the standardized logarithm of the amount paid to policyholders in Florida after catastrophic weather events. On the training set, the linear regressor achieved an MSE of 0.5618, and on the validation and testing sets it reached an MSE of 0.5567. Below, I plot the predictions of the linear regressor against the labels of the dataset. The blue line is a least-squares trendline fit to the two sets:

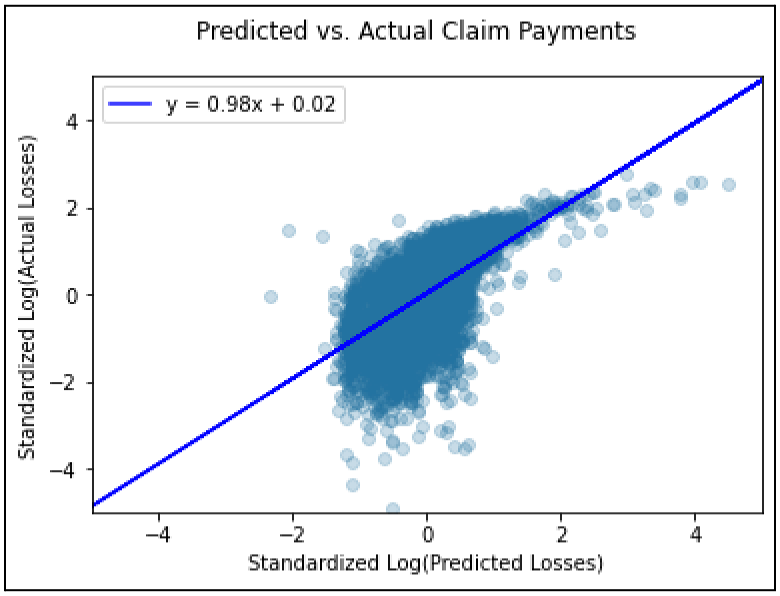


Figure 8. *Linear regressor's predicted vs. actual standardized log of flood insurance payouts after catastrophic weather events in Florida.*

Interestingly, it seems that while the regressor was unable to distinguish with certainty between policies which were paid smaller amounts of money, the model made a number of extremely high-confidence predictions for policies which received significant quantities in claim payments.

The gradient boosting regressor exhibited the strongest performance in fitting the augmented FEMA redacted claims dataset for catastrophic storms in Florida. On the training dataset, the gradient booster achieved an MSE score of $4.41*10^9$, and on the validation and testing datasets it achieved an MSE of $3.29*10^9$. These scores are notably strong given that labels were not transformed before training. In fact, the predictions explain about 66% of the variance in the test set labels, and the two sets exhibit a 0.958 linear correlation. Below, I plot the gradient booster's predictions against actual quantities paid out by the National Flood Insurance Program to policyholders in Florida after flooding events:

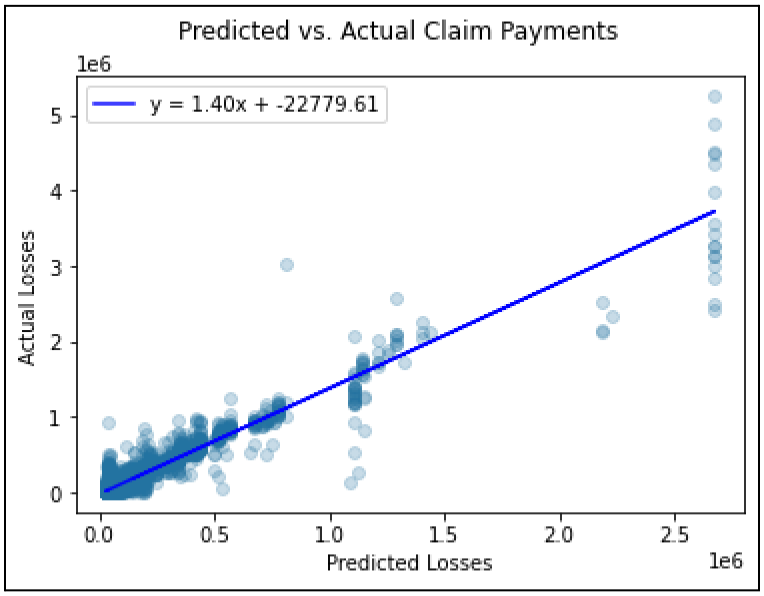


Figure 9. *Gradient boosting regressor's predictions vs. actual flood insurance payouts after catastrophic weather events in Florida.*

The scores achieved by the gradient boosting regressor exceeded the goals for predictive power outlined in the project proposal and would provide a robust foundation for pricing the insurance policies issued by FEMA in the wake of catastrophic weather events in Florida.

VI. Conclusions

The successful application of diverse modeling techniques in this study underscores the potential of machine learning in transforming risk assessment practices within FEMA's National Flood Insurance Program and the broader reinsurance industry. The deployment of dense neural networks, linear regression after polynomial expansion and principal component analysis, and gradient boosting regression demonstrated varied levels of efficacy, with the gradient boosting regressor emerging as the most robust in predicting flood insurance payouts to policyholders in Florida after catastrophic weather events. These results reinforce the notion that non-linear modeling approaches can capture the nuances of natural disaster risk better than traditional linear models, particularly in highly variable environments with disparate types of features. Throughout the project, the challenges of handling large datasets, integrating data sources, and overcoming significant skew in label distribution were addressed through extensive data processing and experimental modeling strategies.

By accurately forecasting NFIP claims with these models, FEMA can optimize resource allocation, improve financial planning, and offer better rates in catastrophe bond markets, thereby enhancing the resilience of communities to flood disasters. Moving forward, the predictive power of machine learning promises to transform the reinsurance market, but the deployment of these models in a real-world scenario would necessitate continuous monitoring and updating to adapt to the dynamic nature of climate patterns and urban development. Overall, this study succeeded in advancing the predictive capabilities of the NFIP but also set a precedent for the use of advanced machine learning techniques in federal disaster management and insurance frameworks. With continued refinement and adaptation, these models promise to significantly bolster the NFIP's ability to protect the thousands of American communities and safeguard the financial integrity of the program for years to come.