Applying Singular Value Decomposition and Principal Component Analysis to Classify Earthquake Damage

Niki Yoon

March 16, 2025

Abstract

This project applies machine learning techniques to analyze the 2015 Gorkha earthquake dataset. Singular Value Decomposition (SVD) and Principal Component Analysis (PCA) are utilized to reduce dimensionality, and a predictive model is developed to classify structural damage levels.

1 Introduction

The 2015 Gorkha earthquake, also known as the Nepal earthquake, struck on April 25, 2015, with a magnitude of 7.8. It was one of the most devastating natural disasters in Nepal's history, causing over 8,000 deaths and injuring more than 21,000 people. The earthquake triggered massive landslides and aftershocks, further worsening the damage. Infrastructure was severely affected, with nearly 600,000 buildings destroyed and an additional 285,000 damaged. Rural villages, particularly in the mountainous regions, suffered the worst destruction, leaving thousands homeless. The economic impact was estimated at over $10 billion, nearly half of Nepal's GDP. Rebuilding efforts have been slow due to logistical and financial constraints, emphasizing the need for better predictive models to assess and mitigate earthquake damage.

Notably, many buildings in the area lacked more modern earthquake-resistant designs ("earthquake resistant" being relative), especially in rural areas where buildings were made of mud mortar, stone, or reinforced brick, all of which are vulnerable to earthquakes. Furthermore, before the disaster, there was a lack of standardized building codes and many homes were self built, with vertical expansion (without foundation reinforcement). I aim to apply the techniques learned in this class to analyze the post-earthquake damage on individual buildings, and factors that contributed to such damage.

(a) Collapsed building after the earthquake.

(b) Unreinforced stone masonry

Figure 1: Examples of earthquake-damaged buildings.

2 Methodology

2.1 Data and Preprocessing

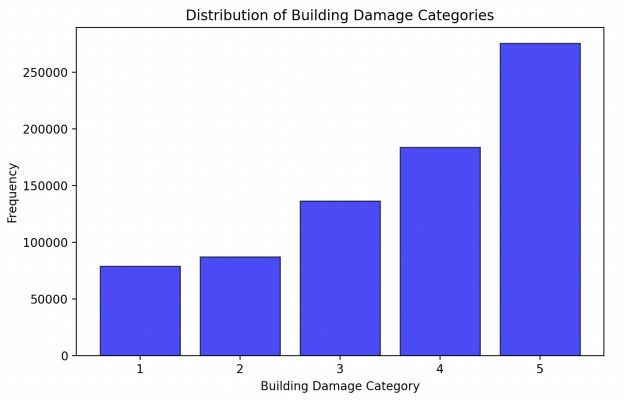
The National Planning Commission (NPC), an advisory body for the Nepal Government, as well as KLL (a civic tech company) and the Central Bureau of Statistics undertook an Open Mapping project of creating one of the largest disaster datasets ever assem-

bled. The dataset has over 762,000 rows (buildings), 30 columns (features) and includes information such as the house conditions, structural dimensions, and most importantly the impact that the earthquake had on the building, on a "damage-scale" from 1-5, 1 meaning little to no damage, and 5 meaning complete destruction. Each building has a unique ID, some of these many categories include:

```
Foundation type (mud, mortar, stone, etc)
Roof_type (mud, stone, bamboo, etc)
Ground_floor_type
Land_surface_condition
Height
Area_sq_ft
Age_building
```

Many of these features are categorical, so during the preprocessing one-hot encoding was applied (to change different categorical features into boolean values). After preprocessing, the data set was 762,000 rows and 147 columns (some columns with close to 0 variance were removed). The data was then split into 80% training and 20% testing, meaning around 610,000 rows and 148 columns of data were going to be used to train the model. I also scaled the data to have a mean of 0 and standard deviation of 1, as well as min-max scaling.

Click to View Dataset Portal



(a) Basic analysis of building damage (based on the damage grade feature)

2.2 Dimensionality Reduction

As the preprocessed data has 147 columns, I apply dimensionality reduction (SVD and PCA) to shrink the matrix to have only 20 columns! (We will see that the accuracy of the model remains high). Given the one hot encoding, there is significant colinearity between features, making the data extremely redundant before I apply dimensionality reduction. I apply both SVD and PCA, and then train a classification machine learning model on both.

2.2.1 Singular Value Decomposition (SVD)

Singular Value Decomposition (SVD) is a factorization of a matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$ into three matrices:

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{1}$$

where:

- $\mathbf{U} \in \mathbb{R}^{m \times m}$ is an orthogonal matrix containing the left singular vectors. In our cases, $\mathbf{U}$ represents the how much each **building** contributes to the structure of the dataset.
- $\mathbf{\Sigma} \in \mathbb{R}^{m \times n}$ is a diagonal matrix containing singular values σ_i in decreasing order. The singular values indicate the importance of each singular vector. Higher values mean the vectors in $\mathbf{V}^T$ contribute to more of the dataset.
- $\mathbf{V}^T \in \mathbb{R}^{n \times n}$ is an orthogonal matrix containing the right singular vectors. In this case, it describes how each feature (columns) contribute to each latent pattern. These rows are the principal directions.

To reduce dimensionality, we approximate $\mathbf{X}$ using the top k singular values:

$$\mathbf{X}_k = \mathbf{U}_k\mathbf{\Sigma}_k\mathbf{V}_k^T \tag{2}$$

where $\mathbf{U}_k$, $\mathbf{\Sigma}_k$, and $\mathbf{V}_k$ retain only the top k components. This allows us to capture most of the variance while reducing the data size. I choose $k = 20$.

2.2.2 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a method for reducing data dimensionality while preserving the most significant variance. Given a dataset $\mathbf{X} \in \mathbb{R}^{m \times n}$, we first center it by subtracting the mean:

$$\mathbf{X}_{\text{centered}} = \mathbf{X} - \bar{\mathbf{X}} \tag{3}$$

Then, we compute the covariance matrix:

$$\mathbf{C} = \frac{1}{m} \mathbf{X}_{\text{centered}}^T \mathbf{X}_{\text{centered}} \tag{4}$$

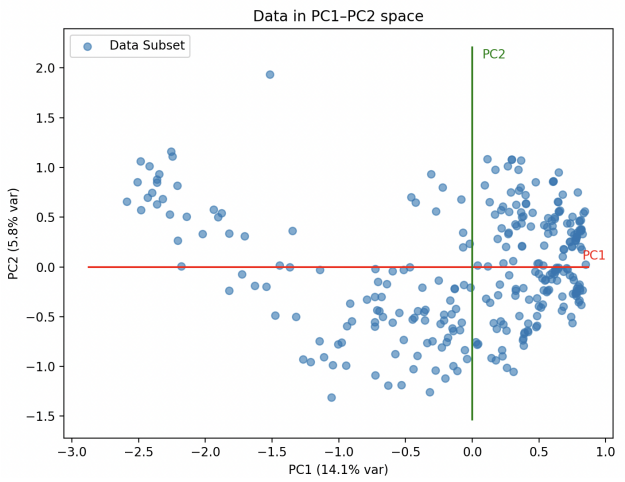
PCA finds the eigenvectors and eigenvalues of $\mathbf{C}$:

$$\mathbf{C}\mathbf{w} = \lambda \mathbf{w} \tag{5}$$

where λ are the eigenvalues and $\mathbf{w}$ are the corresponding eigenvectors. The top k eigenvectors form the transformation matrix $\mathbf{W}_k$:

$$\mathbf{X}_k = \mathbf{X}_{\text{centered}} \mathbf{W}_k \tag{6}$$

which projects the data onto a lower-dimensional space while retaining the most important variance.



(a) A random sample of the data plotted against the first two principal components. We see that PC1 and PC2 account for around 20% of the variance in the total data set.

Using the first two PCA components, we see that the most important features were:

```
PCA1:
has_superstructure_mud_mortar_stone
foundation_type_Mud mortar-Stone/Brick
other_floor_type_Timber/Bamboo-Mud
ground_floor_type_Mud

PCA2:
has_superstructure_timber
other_floor_type_TImber/Bamboo-Mud
other_floor_type_Timber-Planck
other_floor_type_TImber/Bamboo-Mud
```

2.3 Machine Learning Model

I trained a basic logistic regression model to classify each building into the damage grade, 1-5. I trained a logistic regression model on both the SVD as well as PCA components. For the SVD, I used a matrix of rank 20, and I also used the first 20 PCA components. Then, I also used the scikit library to train a random forest model, for the same purpose.

3 Results

Model	Accuracy	F1 Score
SVD-based Logistic Regression	86.4%	0.76
PCA-based Model	87.1%	0.85
SVD-based RF	88.31%	.88
PCA-based RF	88.40%	.88

Table 1: Performance comparison of models

4 Discussion

I was able to achieve around $87 - 88\%$ accuracy on all models. Again, I believe this was achievable with dimensionality reduction because there is a high amount of co-linearity between the dataset with the one hot encoding. For example, if a structure has a mud floor, of course any other floor type is irrelevant (although it will still be noted in a whole separate

feature).

It's known that PCA can be derived from SVD if the data is mean-centered. Mathematically, PCA involves computing the eigendecomposition of the covariance matrix, which in practice can be accomplished literally with the same steps as SVD. Therefore, both methods produce comparable latent feature spaces—leading to similar classification accuracies. Indeed, when subtracting the column means from the original dataset before SVD, the resulting lower-dimensional embeddings are essentially the same as those from PCA. This explains why the two approaches yielded similar performance in this project.

One important factor I realized throughout this project was data preprocessing: standardizing or mean-centering the data prior to SVD is crucial for a fair comparison with PCA, since PCA inherently centers the dataset by subtracting each feature's mean. Any mismatch in centering or scaling steps between the two pipelines could have produced differences. In this project, consistent preprocessing ensured the spaces were genuinely comparable. I struggled getting good scores on my model before applying proper scaling.

5 Conclusion

I was able to achieve high accuracy on various models using PCA and SVD techniques. The original dataset was so large, that my computer continually crashed (or took an impractical amount of time) when training on the original dataset. This project reiterates the effectiveness of dimensionality reduction in machine learning applications.

This project was conducted as part of Math 104 at Stanford University, under the guidance of Professor Gene Kim. The analysis of PCA and SVD in classification aligns with the course's focus on linear algebra applications in real-world data processing.

6 References

References

[1] Kim, Gene. Math 104: Applied Linear Algebra. Stanford University, 2024-2025.