# Analysis of National Highway Bridge Inventory Database and Model Building

## Introduction

The aim of this project is to demonstrate the implementation of various machine learning algorithms on a given dataset. The dataset used is the popular NBI bridge dataset provided by the Federal Highway Administration and it can be accessed from the famous UC Irvine dataset repository. The bridge data has no specified target variable thus making it suitable for unsupervised learning problems such as clustering. In this project, however, clustering is first conducted and the cluster labels, which are 1 and 2 in this case, were used to assign labels (class 1 deterioration and class 2 deterioration) to the data. Once this was done, supervised learning was performed. These include logistic regression, KNN and Deep Neural Network. Similarly, Principal Component Analysis was conducted to reduce the variable space and the principal components were used as variables for another logistic regression model. ROC curves were plotted for these models and their performances were compared using the accuracy and F1-score metrics.

## Dataset & Data Preparation

The dataset is a ".csv" file consisting of 22 sheets, each representing the data for a given year between 1995 and 2013. Each sheet has 8 data tables where each table has 52 rows representing all the states in the USA including the District of Colombia and Puerto Rico, and 23 columns representing the measured total area of several bridge components. In addition to the total area of the components, the condition states which could be either structural deficient, functionally deficient or combination of both were also presented separately. Of all the 23 components, 13 were considered because the remaining seem to have several missing data points or zero measured area, indicating that some states do not have those types of bridges. For simplicity, the total area of structures and the area of deficient structures were considered. A new data table for structural deficiency ratio was obtained by dividing the total area of deficient structures by the total area of structures. Following this operation, missing data "NAN" due to the division operation where the total area is zero were replaced by 0 for completeness. This preprocessing was conducted by writing a python script that automatically extracts data from all the sheets to capture information from all the years. The resulting dataset from this process has a dimension of 1144 by 13.

## Exploratory Data Analysis

Basic exploratory analysis was conducted using correlation plot that shows the relationship between all the variables, as depicted in Figure 1. In Figure 1, Slab, Multibeam, GirderFloorBeamSystem, TeaBeam, and Frame seem to be the most correlated set of variables. Also, Frame is correlated with ArchDeck, even though ArchDeck seems to be uncorrelated with most of the variables. Other variables that appear to be uncorrelated with most of the variables include BoxBeamMultple, Culvert, TrussThru and TrussDeck as they all have lower correlations, with some close to zero. It is worth mentioning that since the correlations shown in the plot indicates linear relationships between variables, there may be some nonlinear relationship between the variables which are not reflected in the correlation plot. As such for this exercise,

these variables were left as they were expected to contribute some explanations to the variance in the dataset, however small that may be.

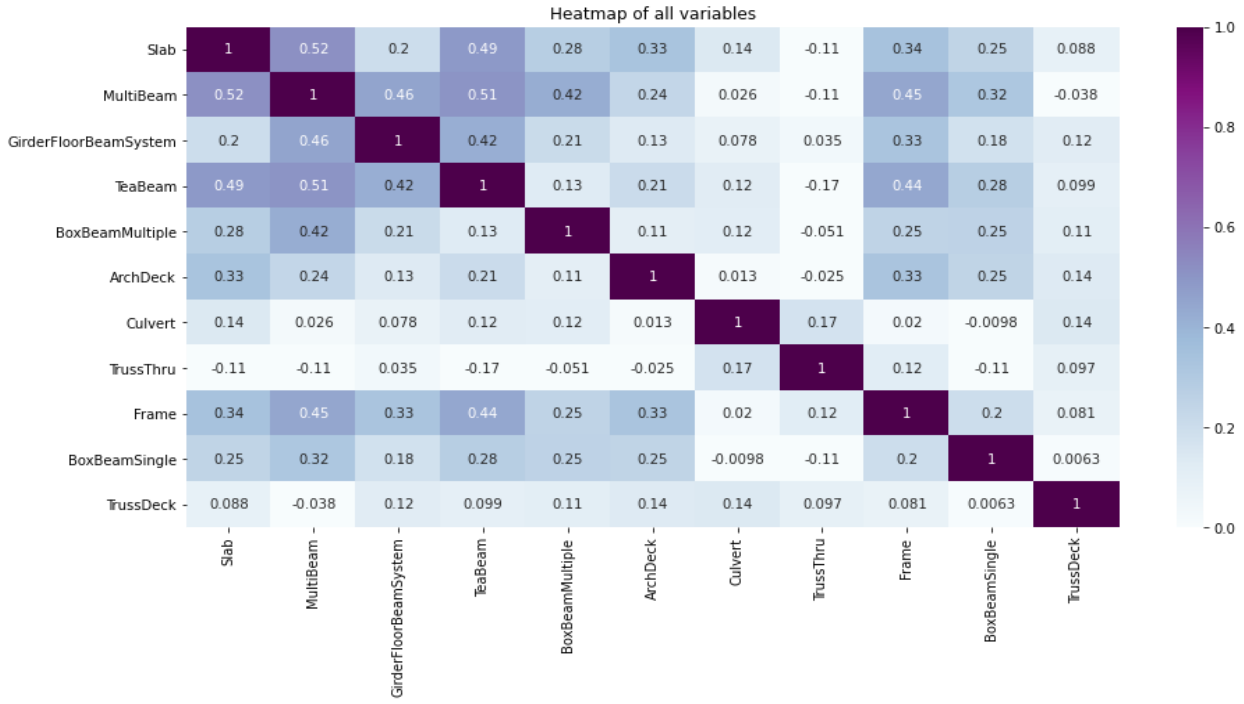



**Figure 1. Correlation plot of all the variables**

## Modeling

### Hierarchical Clustering

The first step of this analysis of the data is hierarchical clustering. As an unsupervised learning approach, hierarchical clustering is used to segment the observations in the dataset into distinct groups without having to specify the number of K clusters unlike in K-Means clustering where this parameter must be specified and tuned. In this project, hierarchical clustering is first performed using the data for years 2013 and 2012. As shown in Figures 2 and 3, the states can be clustered into three and four groups respectively. However, when the combined multi-year data was considered, a cluster of two groups was generated as shown in Figure 4. Using this information, each data point was labelled as either class 1 deterioration or class 2 deterioration. Both classes have enough observations (519 for class 1, 625 for class 2) indicating that the data is not imbalanced. While this two-class nomenclature may not hold any valid meaning in bridge deterioration studies, it is so adopted in this exercise to demonstrate that it is possible to convert an unsupervised problem to a supervised learning problem by adding labels from the former to an originally unlabeled data. Subsequent analyses were conducted using this labeled dataset.

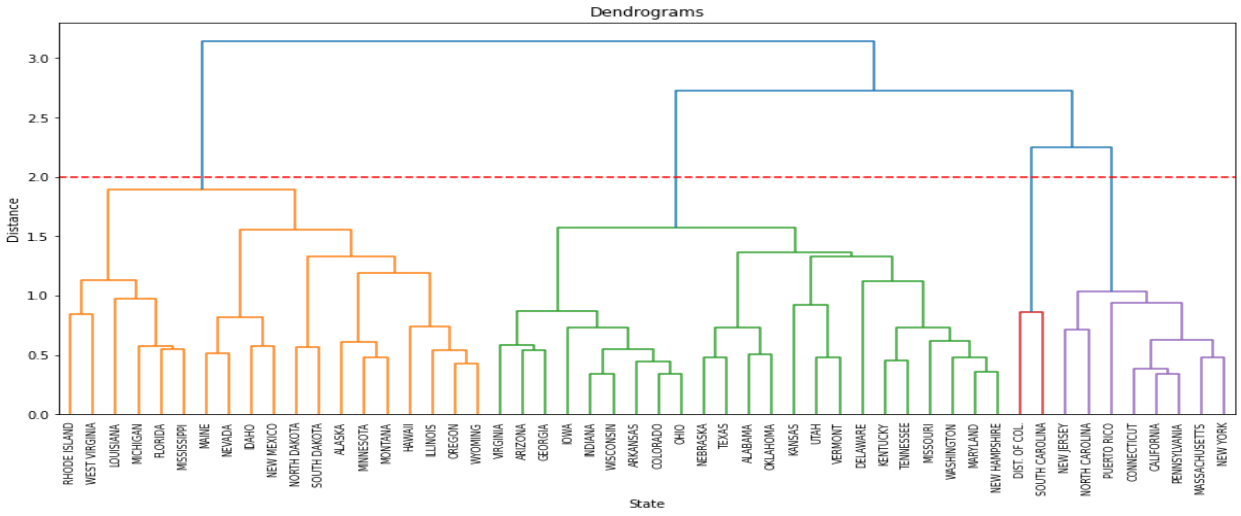



**Figure 2. Dendrogram of Hierarchical clustering for a single year data considering year 2013.**

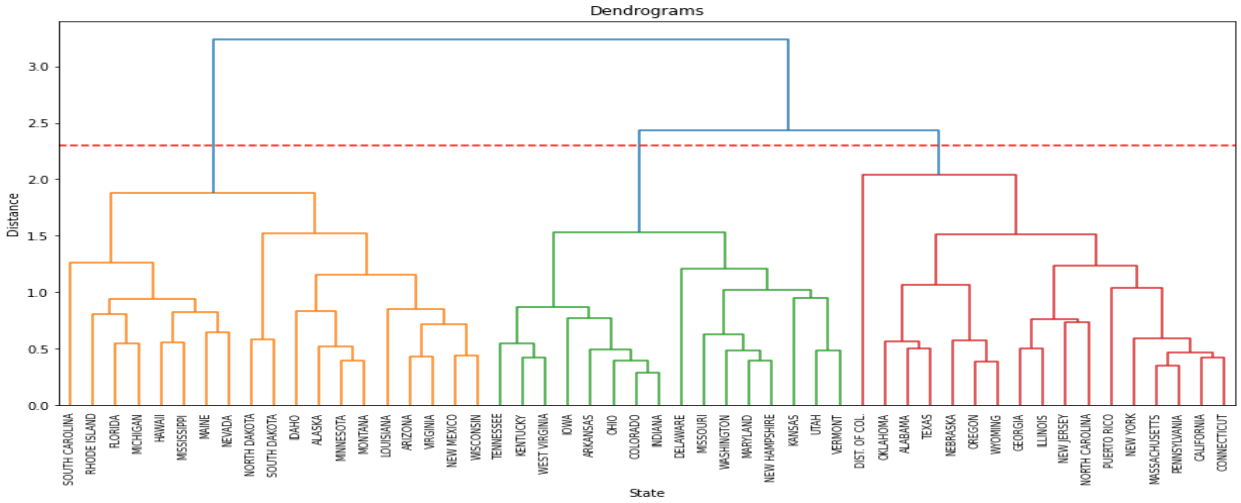



**Figure 3. Dendrogram of Hierarchical clustering for a single year data considering year 2012.**

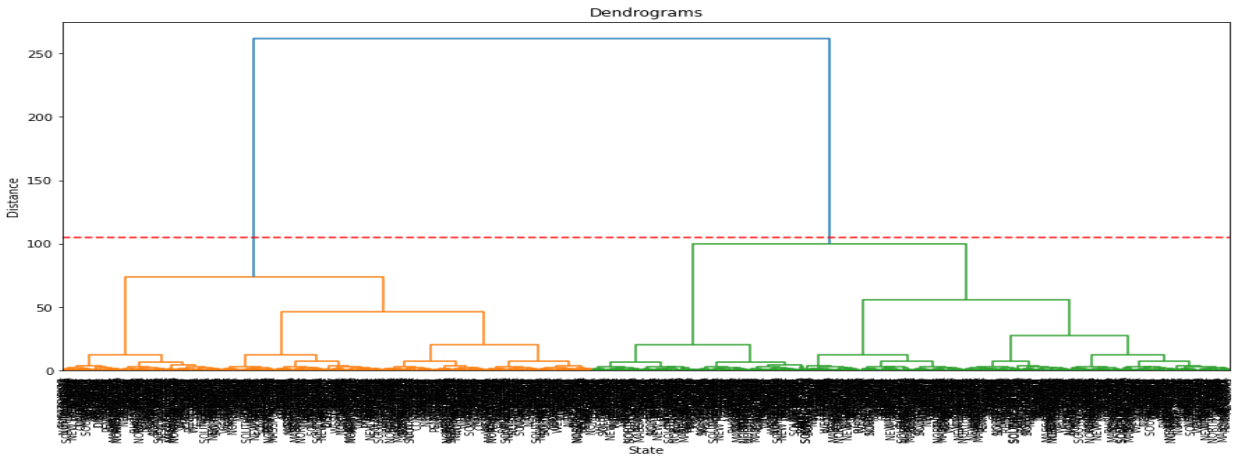
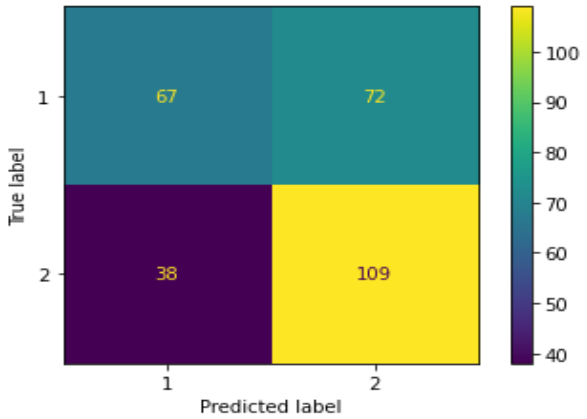



**Figure 4. Dendrogram of Hierarchical clustering for the multiyear data (1995 – 2013)**

**Logistic Regression**

Once clustering was performed, the labeled dataset was used for conducting logistic regression analysis. Logistic regression analysis is a supervised learning classification approach in which the target variable is a categorical type such as a case where the problem involves binary classification as is this case in this project.

To implement logistic regression, the variables "state" and "year" in the data frame were dropped, thus resulting in new dimension of 1144 x 11. The data is then split into train and test sets at a ratio of 75:25. Python's scikit-learn library was used to fit a logistic regression on the training set and the model was validated using the holdout (test) set. The model was evaluated by measuring its accuracy, F1 score and AUC. Figures 5a and 5b show a confusion matrix of the model classification and the ROC curve respectively.



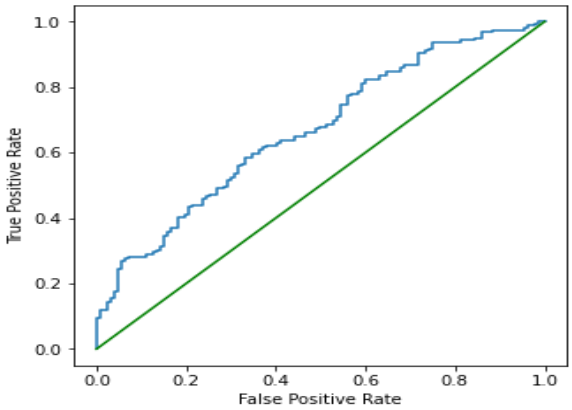




**Figure 5a. Confusion Matrix** **Figure 5b. ROC curve**

In Figure 5a, it can be observed that out of the 286 test observations, 67 were correctly classified as class 1 deterioration, 109 were correctly classified as class 2 deterioration, while the remaining 110 are misclassified predictions. The accuracy score for this model is 61.54% while the F1 score is 54.9% indicating that the model barely classifies at an average performance.

In Figure 5b, it is shown that there is indeed a deviation between the actual test set and the predictions. The area under the curve, AUC, when measured is 62.1% which just like the accuracy and F1 score is not good enough.

In an attempt to improve the model, the logistic regression analysis was conducted with two different cross validation methods, namely K-fold cross validation and stratified K-fold validation. Although both approaches involve dividing the data into folds where one of the folds is used as test in turn by replacements while the remaining are considered as train set, the difference between both methods is that while K-fold performs random sampling on the data, stratified K-fold performs stratified sampling. Both methods improved the model albeit slightly with an accuracy of 62.11% and 61.71% respectively. Details of these two cross validation processes, including their confusion matrices and ROC curves, are contained in the attached Jupyter notebook.

## K Nearest Neighbor

Another supervised learning method considered in this exercise is K Nearest Neighbor. It is a classification algorithm in which a label is assigned to an originally unlabeled data by determining its distance from the existing data points whose labels are known. The process involves first calculating the distance from the observations in the training set. Then a pre-specified number of nearest neighbors, K are selected based on their distances to the new unlabeled dataset and the label of the closest data point is assigned to the unlabeled data point. Alternatively, by using a majority vote approach, the label that appear the most among the nearest neighbors is assigned to the new data point. To successfully implement KNN, the number of neighbors K must be tuned. Also, the distance metric which can be either Euclidean, Manhattan or Minkowski etc., should be selected. In this project, the GridSearchCV module from the scikit-learn library was used to tune and select the best parameters. The result of this tuning suggests the following parameters: distance metric: Manhattan; number of neighbors K: 2; and labeling metric: distance. Using these parameters, a KNN model was fitted. The resulting accuracy of the model is 92.7% while the F1-score is 91.5%.

## Principal Component Analysis

Principal component analysis (PCA) is a method used to reduce a large data set of variables to a smaller set of variables, called principal components, through a linear combination of the original variables. The principal components capture unique proportion of the variance in the dataset. This way, the subsequent modeling, such as prediction or classification, is simplified as there are now fewer variables to build the model with. In this project, PCA was conducted on the original dataset and the proportion of explained variance captured by the principal components is presented in Figure 6a. Figure 6b presents the cumulative explained variance captured by all the principal components.

| | percent explained variance |
|---|---|
| PC1 | 27.144242 |
| PC2 | 21.165122 |
| PC3 | 12.035578 |
| PC4 | 9.167196 |
| PC5 | 8.169006 |
| PC6 | 5.835628 |
| PC7 | 5.139621 |
| PC8 | 4.793356 |
| PC9 | 2.824355 |
| PC10 | 2.349003 |
| PC11 | 1.376894 |

Figure 6a. Percent explained variance by principal components

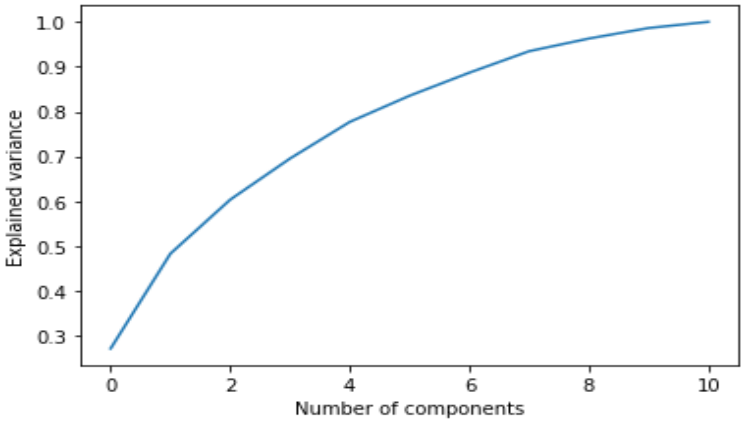



Figure 6b. Cumulative explained variance against number of principal components

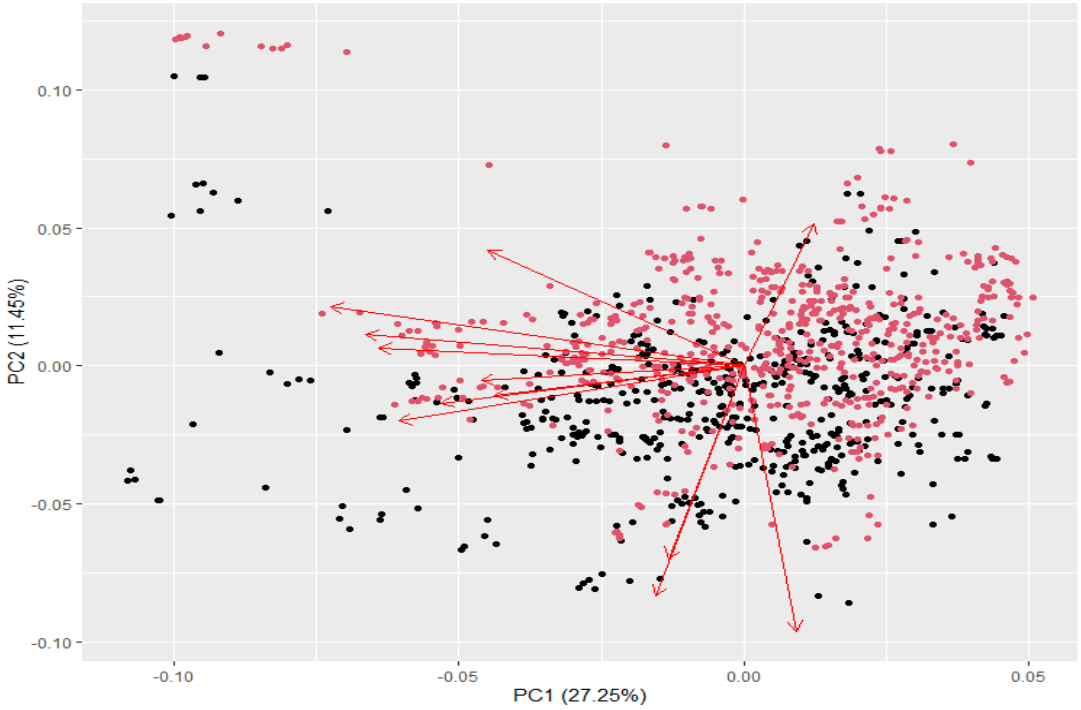



Figure 6c. Biplot of PC2 against PC1

In Figure 6a, it is seen that the first principal component captures only 27% of the variance in the dataset, second principal component captures 21%, third principal component captures 12% and so on and so forth. In Figure 6b, it is observed that the first 6 principal components capture 85% of the variance and the addition of the seventh principal component result in only 5% more explained variance. As such, from this analysis, it may suffice to use only the first six principal components for developing a logistic regression model. Figure 6c is the biplot of principal component 2 versus 1.

To build a classification model with the principal components, the coefficients of the principal components are mapped to both the training and test sets and the same method of building logistic regression discussed previously, is implemented. The resulting model yielded an accuracy of the model is 57.69% and a F1-score is 61.09%.

**Deep Learning**

Deep learning involves the use neural network architectures which have more than three hidden layers for predictions. In this project, deep learning was built to classify the deterioration types as either class 1 or class 2. TensorFlow was used in this project for building and compiling the network as well training and validating its performance. The network is a simple one with five hidden layers, an input layer and output layer. The input layer has number of units (i.e., neurons) corresponding to the number of features (11 variables) in the dataset. Notice that passing in the input this way is similar to how the flatten layer is applied to images to reshape the image array to a vector. The output layer has two output units that provides the probability of being class 1

and class 2. The first three hidden layers have 12 neurons and the last two have 6 neurons each, resulting in total of 590 parameters to be trained. In the feedforward process, each of the hidden layers are followed by drop out layers which randomly penalize some of the weight thus serving a regularization procedure. The hidden layers were activated using the ReLU activation function while the output layer uses SoftMax function to restrict the output to a value between 0 and 1, which is the probability of being class 1. Figure 7 presents the architecture of this model.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 12)                144

 dropout (Dropout)           (None, 12)                0

 dense_1 (Dense)             (None, 12)                156

 dropout_1 (Dropout)         (None, 12)                0

 dense_2 (Dense)             (None, 12)                156

 dropout_2 (Dropout)         (None, 12)                0

 dense_3 (Dense)             (None, 6)                 78

 dropout_3 (Dropout)         (None, 6)                 0

 dense_4 (Dense)             (None, 6)                 42

 dropout_4 (Dropout)         (None, 6)                 0

 dense_5 (Dense)             (None, 2)                 14

=================================================================
Total params: 590
Trainable params: 590
Non-trainable params: 0
_________________________________________________________________
```

**Figure 7. Architecture of the neural network model**

To estimate the weights, stochastic gradient descent (SGD) optimization algorithm was used to optimize the loss function over the parameters of the network. The loss function used was binary cross-entropy as it is desired for a binary classification problem such as in this project. Several iterations of the feedforward process were performed with the gradients of the weights updated during every iteration. These updates were performed by backpropagating the error from the feedforward process over the parameters. For improved performance, the training was conducted in mini batches with batch size of 32. The number of epochs of training the model is 1000. The performance of this model on the validation set is 39.51% accuracy and 28.57% f1-score.

### Summary of Results

The results of all the classification models trained in this project are summarized in Table 1. Accuracy and F1-score are the metrics considered for comparing the performance of all the models as they are popularly regarded as the most suitable for classification models. Other

metrics are Recall and Precision but these two are captured by the F1-score. In Table 1, it is seen that most of the models performed near average on the validation data sets. Deep learning model seemed to perform worst. This was expected as the dataset is unlikely suitable for deep learning. PCA also performed near average. This may be because only 80% of the variance in the dataset is captured by the six principal components used for the classification model suggesting that the remaining variance not captured may have been some important information that could contribute to the improvement of the model. Overall, K Nearest neighbor yielded the most outstanding performance with both accuracy and f1-score above 90%. This is likely due to the robust parameter tuning that was conducted with the GridSearchCV module that produced the best parameters for the model. It could also be because K nearest neighbor algorithm involves a measure of the distances between the observations in the data just like clustering, and since the labels for this dataset were generated from the clustering process performed initially, it makes sense for there to be minimal misclassification in the KNN model.

**Table 1. Summary of model performance evaluations**

| Model | Accuracy (%) | F1-Score (%) |
|---|---|---|
| LogisticRegression (basic hold-out, no CV) | 61.54 | 54.9 |
| LogisticRegression (K-fold cross validation) | 62.11 | 54.92 |
| LogisticRegression (stratified k-fold cross validation) | 61.71 | 58.46 |
| K Nearest Neighbor | 92.70 | 91.50 |
| PCA with LogisticRegression | 57.69 | 61.09 |
| Deep Learning | 39.51 | 65.41 |

## Conclusion

In this project, the application of different machine learning algorithms on the famous NBI bridge dataset has been demonstrated. Simple data wrangling and exploratory data analysis were conducted to identify the variables of interest and to show the interactions between these variables. By performing clustering, the observations in the data set were grouped into two classes and these classes were used to label the dataset thus converting the dataset into one that could be used for supervised learning. It is important to mention, however, that this nomenclature from the clustering, which are referred to as class 1 deterioration and class 2 deterioration in this exercise, is just an attempt to demonstrate that it is possible to convert unlabeled data to labeled data and may not hold any true importance in bridge deterioration studies. Logistic regression, K Nearest Neighbor (KNN), Principal Component Analysis (PCA) and Deep Learning were all performed on the dataset. The logistic regression had three variants, including one with K-fold validation, stratified K-fold validation, and a basic hold-out validation. Logistic regression was also conducted with the principal components from the PCA. Of all these models, KNN was found to perform best with accuracy and f1-score above 90%. All codes used for the analysis can be found in the attached jupyter notebooks as appendices.