CS699
Sicheng Yi
Amina Bauyrzhan

Intermediate Project Report

We worked with the dataset project_dataset.csv, which initially was preprocessed by Professor and had 4947 tuples and 204 attributes. We performed further preprocessing.

We performed various data preprocessing steps, feature selection, outlier detection, data standardization, handling imbalanced class distribution, dataset splitting, and the construction and evaluation of multiple classification models. Below is a detailed description of each step performed in the code:

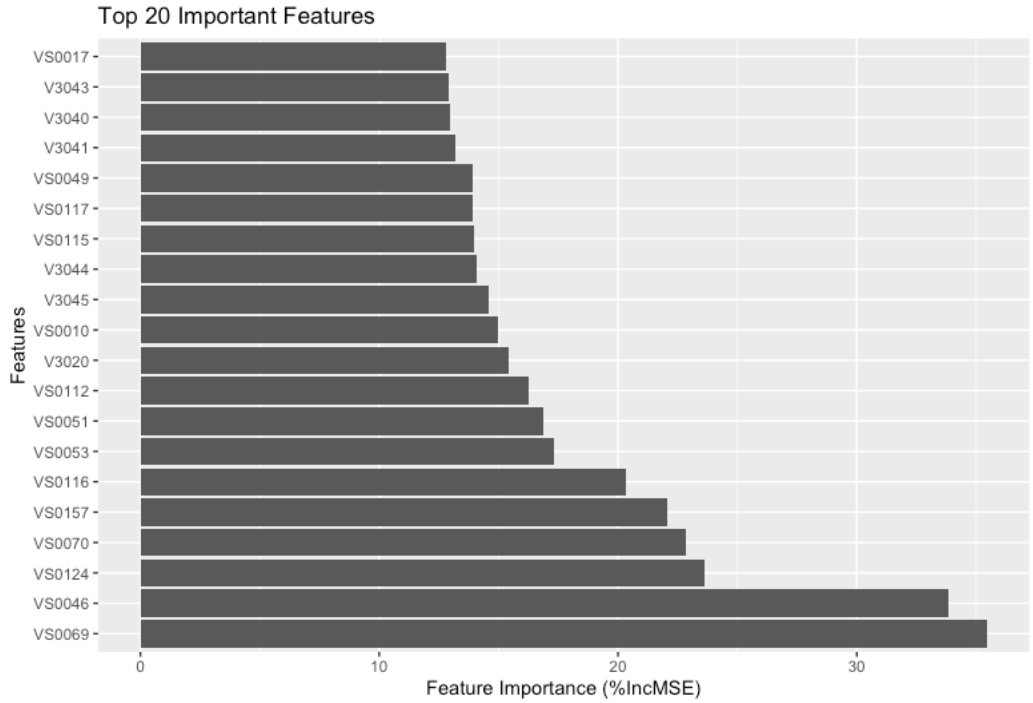
Feature Selection:
In this section, we identify the most important features in the dataset that are likely to influence the prediction of bullying victims. We use a random forest model to assess the importance of each feature and select the top 20 features based on their contribution to the model's predictive performance. The importance scores were calculated to determine which features are most relevant for modeling. The table provided displays the top 20 important features, along with their respective importance scores based on two metrics: %IncMSE (Increase in Mean Squared Error) and IncNodePurity (Increase in Node Purity). %IncMSE (Increase in Mean Squared Error): This metric measures how much the mean squared error increases when a particular feature is randomly permuted. Higher values indicate that the feature has a greater impact on reducing the model's prediction accuracy when perturbed. In simpler terms, higher values suggest that the feature is more important for the model's performance.

IncNodePurity (Increase in Node Purity): This metric indicates how much the purity of decision tree nodes (in the random forest) increases when a particular feature is used for splitting. Higher values suggest that the feature leads to more homogeneous groups of data points in the decision tree nodes, making it more important for classification.

Table: Top 20 Important Features

	%IncMSE	IncNodePurity
VS0069	35.44806	23.382939
VS0046	33.83066	49.177869
VS0124	23.63488	21.872517
VS0070	22.86432	18.084740
VS0157	22.08349	8.015085

VS0116	20.32947	3.806795
VS0053	17.34362	7.655380
VS0051	16.87100	11.570485
VS0112	16.25110	15.664938
V3020	15.39340	12.589236
VS0010	14.99305	10.510261
V3045	14.55608	1.510556
V3044	14.05781	1.549421
VS0115	13.98901	3.633011
VS0117	13.90720	2.802826
VS0049	13.90102	6.682048
V3041	13.19670	2.577352
V3040	12.98298	2.263087
V3043	12.90533	3.100565
VS0017	12.79016	11.477576



Data Integrity.

We checked for missing values, but there were no missing values.

We checked for outliers in the top 20 important features of our dataset. Outliers are extreme data points that deviate significantly from the majority. Detecting outliers is crucial because they can indicate data errors, reveal valuable insights, or impact the reliability of statistical analyses. We used a function to identify outliers in each of the top 20 features, storing the results in outliers_detected. We determined if any outliers were present for each feature using

outliers_present. Then we created boxplots for features with outliers to visually explore their data distribution and the presence of extreme values.

Data Standardization:
After selecting the important features, we standardize the data. Standardization is the process of scaling numerical features to have a similar range. This step ensures that all features have a common scale, preventing any particular feature from dominating the model due to its larger magnitude.

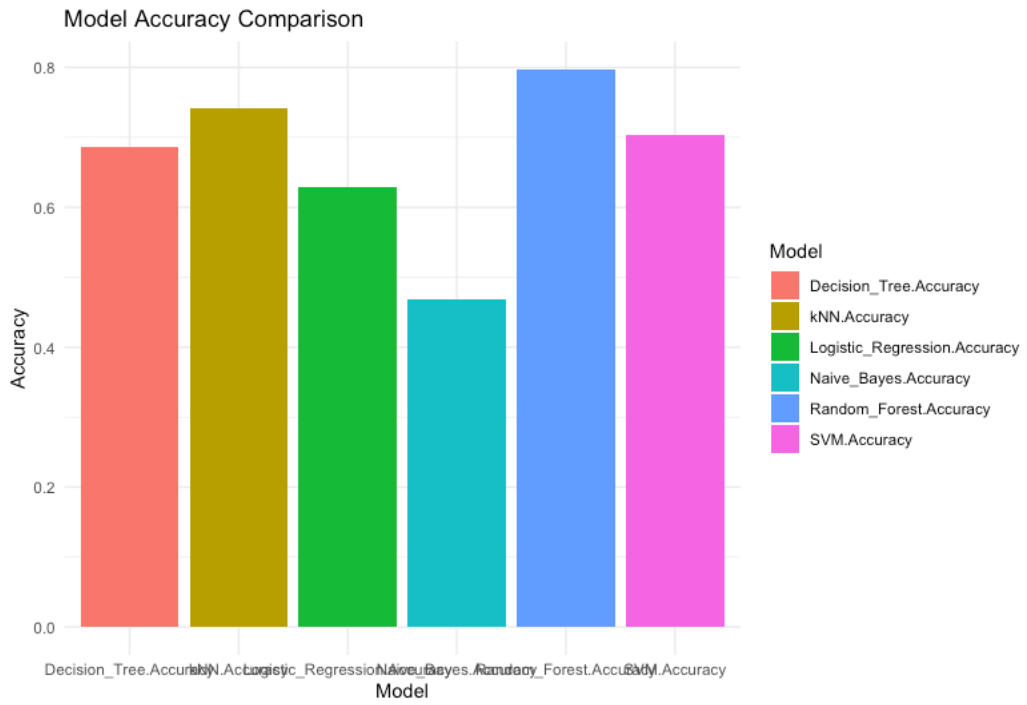
Handling Imbalanced Class Distribution:
Here, we address the issue of imbalanced class distribution. The class distribution refers to the proportion of bullying victims (class 1) and non-victims (class 0). We notice that one class is significantly larger than the other. To balance the classes, we oversample the minority class (class 1) to make its count equal to half of the majority class (class 0).

Stratified Split of the Dataset:
We split the dataset into two subsets: a training set and a test set. It's important to ensure that the class distribution is preserved in both sets, so we perform stratified splitting. This means that both the training and test sets have a similar proportion of bullying victims and non-victims.

Building Classification Models:
In this section, we create and train various classification models. These models include Logistic Regression, Decision Tree, Random Forest, Support Vector Machine (SVM), k-Nearest Neighbors (k-NN), and Naive Bayes. For each model:

We verify that the levels of the target variable (o_bullied) match between the training and test datasets.
We build the model using the respective algorithm.
We make predictions using the model, and we convert predicted probabilities into binary classes based on a threshold (usually 0.5).
Model Evaluation:
After creating and predicting with each classification model, we evaluate their performance. This is done by calculating confusion matrices, which provide information about the number of true positives, true negatives, false positives, and false negatives for each model.

Determining the Best Model:
To determine the best-performing model, we compute the accuracy of each model. The accuracy indicates the proportion of correctly predicted instances. The model with the highest accuracy is selected as the best model for predicting bullying victims.



Data Integrity.

We checked for missing values, but there were no missing values.

We checked for outliers in the top 20 important features of our dataset. Outliers are extreme data points that deviate significantly from the majority. Detecting outliers is crucial because they can indicate data errors, reveal valuable insights, or impact the reliability of statistical analyses.

We used a function called detect_outliers_in_vector to check for outliers in each of these 20 features. The result of this process was a list called outliers_detected, where each element of the list corresponds to one of the top features and contains information about any detected outliers.

Next, we checked if any outliers were detected for each of the top 20 features. We created a logical vector called outliers_present, which indicated whether outliers were found for each feature. We did this by verifying if the corresponding element in outliers_detected was not empty (indicating the presence of outliers) and had a length greater than zero (to confirm that there were outliers).

Reporting Outliers: If we found any outliers (at least one feature with outliers), we printed a message stating that outliers were detected. We also created a list called outliers_names, which contained the names of the features with detected outliers, and printed those feature names.

Boxplot Visualization: Finally, for each feature that had outliers, we created a boxplot to visualize the distribution of data and the outliers. We used the ggplot library to create these boxplots. Each boxplot represented a single feature, and we customized the plot title and axis labels to make it informative.

Outliers detected in the following top features:
[1] "VS0069" "VS0046" "VS0124" "VS0070" "VS0157" "VS0116" "VS0053" "VS0051" "VS0112" "V3020" "V3045" "V3044"
[13] "VS0115" "VS0117" "VS0049" "V3041" "V3040" "V3043" "VS0017"