Workflow Document: Semi-Supervised Binary Classification on Software Vulnerability Dataset (BigVul)

# Approach 1: Label Propagation Algorithm (Sklearn) on Undersampled Data

## 1. Process Overview

Goal: To classify software vulnerabilities using a semi-supervised learning approach where only 10% of the data is labeled and 90% is unlabeled.  
Algorithm: The model is built using the Label Propagation algorithm from Scikit-learn, which propagates labels through a graph constructed with K-nearest neighbors.

## 2. Data Preparation

1. Loading Data:   
 - The embeddings for both the labeled and unlabeled sets were pre-generated and loaded from external files.  
 - The dataset contains both labeled and unlabeled samples:  
 - Labeled Set Size: 2,180 samples.  
 - Unlabeled Set Size: 19,620 samples.  
 - The embeddings for both labeled and unlabeled data were combined into a single dataset for processing.  
   
2. Dimensionality Reduction:  
 - The original feature set contains 3,840 dimensions. To optimize performance, PCA (Principal Component Analysis) was used to reduce the feature space to 256 components. This step helps manage computational resources while retaining critical variance.  
  
3. Normalization:  
 - After PCA, L2 normalization was applied to the embeddings to standardize the feature vectors for cosine similarity during KNN-based graph construction.  
 - Normalization ensures that each vector has a unit length, which improves the performance of KNN in terms of similarity calculation.

## 3. Preprocessing

Semi-supervised Labeling:   
 - The model marks 10% of the dataset as labeled and the rest as unlabeled. The labeled points have known target labels, while the unlabeled points are assigned -1. This allows the Label Propagation algorithm to propagate labels from the labeled points to the unlabeled data.

## 4. Model Explanation

Label Propagation:  
 - The Label Propagation model was initialized using the K-Nearest Neighbors (KNN) kernel. The KNN kernel constructs a similarity graph by connecting each point to its 10 nearest neighbors (K=10).  
 - The algorithm iteratively propagates the labels from labeled samples to their nearest neighbors through the graph. This label propagation continues until convergence, which was set to a maximum of 1000 iterations.  
 - After the process completes, the majority of the samples are labeled, and the model can then evaluate performance on the previously unlabeled set.

## 5. Performance Evaluation

1. Unlabeled Set Evaluation:  
 - The following metrics were used to evaluate the performance of the model on the previously unlabeled samples:  
   
 Unlabeled Accuracy:  
 - Accuracy: 99.77% (0.9977), indicating that the model was highly effective in classifying the previously unlabeled samples.  
  
 Confusion Matrix:  
 [[9805 0]  
 [ 46 9769]]  
  
 Classification Report:  
 precision recall f1-score support  
 0 1.00 1.00 1.00 9805  
 1 1.00 1.00 1.00 9815  
 accuracy 1.00 19620  
 macro avg 1.00 1.00 1.00 19620  
 weighted avg 1.00 1.00 1.00 19620  
   
2. ROC-AUC Curve:  
 - ROC-AUC (Receiver Operating Characteristic - Area Under Curve) scores were calculated for both classes, with AUC values close to 1.00 for both. This confirms the model’s strong ability to distinguish between the two classes.

## 6. Insights and Discussion

High Accuracy: The model’s accuracy of 99.77% on the unlabeled set demonstrates that label propagation is highly effective for this task.  
Handling Class Imbalance: The confusion matrix and classification report show balanced performance across both classes, confirming that the undersampling approach successfully addressed class imbalance.  
Key Parameters:  
 - K=10: This value defines the number of neighbors each sample influences. A moderate K value (10) ensures that each sample interacts with a balanced number of neighbors, preventing overfitting or under-smoothing.  
 - 1000 iterations: The high number of iterations ensures that the labels fully propagate through the graph, allowing the model to reach convergence.