Consensus Clustering based Undersampling Approach to Imbalanced Learning

A dissertation submitted to the Jawaharlal Nehru Technological University, Hyderabad in partial fulfillment of the requirement for the award of degree of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING

Submitted
by

Shaik Abdul Khadeer (20B81A05C1)

Mohammed Afreed(20B81A05C3)

Medari Uday Kiran (20B81A05H7)

Under the Guidance of
Dr. D. Sandhya Rani
Professor

Department of Computer Science and Engineering

CVR COLLEGE OF ENGINEERING

(An UGC Autonomous Institution, Affiliated to JNTUH, Accredited by NBA, and NAAC)
Vastunagar, Mangalpalli (V), Ibrahimpatnam (M),
Ranga Reddy (Dist.) - 501510, Telangana State

2023-24

CVR COLLEGE OF ENGINEERING

(An UGC Autonomous Institution, Affiliated to JNTUH,
Accredited by NBA, and NAAC)
Vastunagar, Mangalpalli (V), Ibrahimpatnam (M),
Ranga Reddy (Dist.) - 501510, Telangana State.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project work entitled **"Consensus Clustering Based Undersampling Approach to Imbalanced Learning**" is being submitted by **Shaik Abdul Khadeer (20B81A05C1), Mohammed Afreed (20B81A05C3), and Medari Uday Kiran (20B81A05H7)** in partial fulfillment of the requirement for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering,** during the academic year 2023-2024.

Project Guide
Dr. D. Sandhya Rani
Professor

Professor-in-charge projects
Dr. V. Dattatreya

External Examiner

Professor and Head, CSE
Dr. A. Vani Vasthala

DECLARATION

I hereby declare that this project report titled "**Consensus Clustering Based Undersampling Approach to Imbalanced Learning**" submitted to the Department of Computer Science and Engineering, CVR College of Engineering, is a record of original work done by me under the guidance of **Dr**. **D. Sandhya Rani**. The information and data given in the report is authentic to the best of my knowledge. This project report is not submitted to any other university or institution for the award of any degree or diploma or published at any time before.

Shaik Abdul Khadeer (20B81A05C1)

Mohammed Afreed (20B81A05C3)

Medari Uday Kiran (20B81A05H7)

Date:

Place:

ACKNOWLEDGEMENT

We wish a deep sense of gratitude and heartfelt thanks to **Dr. Rama Mohan Reddy,** Principal, **and the Management** for providing excellent lab facilities and tools. Finally, we thank all those guidance helpful to us in this regard.

We thank our Vice-Principal **Prof. L. C. Siva Reddy** for providing excellent computing facilities and a disciplined atmosphere for doing our work.

We would like to express heartful thanks to **Dr. A Vani Vasthala,** Professor & Head of the CSE Department, for providing us an opportunity to do this project and extending support and guidance.

We thank **Dr. N Subhash Chandra**, Project Coordinator **Dr. V. Dattatreya,** Professor Incharge, **Dr. M. Swami Das,** Associate Professor and **Ms. K Sravani**, Sr. Asst. Professor for their valuable guidance and support which helped us to complete the project work successfully.

We respect and thank our internal guide, **Dr. D. Sandhya Rani,** Professor, Department of CSE, for giving us all the support and guidance, which made us complete the project duly.

We are thankful for and fortunate enough to get constant encouragement, support, and guidance from all **Teaching staff of CSE Department** which helped us in successfully completing this project work.

Finally, I would like to thank my family, without their special suggestions it was not easy to complete this project.

ABSTRACT

Class imbalance poses a significant challenge in machine learning, with the minority class and majority class. This issue is encountered in real-world applications such as medical diagnosis, malware detection, anomaly identification, bankruptcy prediction, and spam filtering. To address this issue, we propose a consensus clustering-based undersampling approach to handle imbalanced learning. It involves strategically undersampling the instances in the majority class using a consensus clustering-based approach. In the consensus clustering schemes, five clustering algorithms (namely, k-means, k-modes, k-means++, mini-batch k-means, and Fuzzy k-means) and their combinations were taken into consideration. In the classification phase, four supervised learning methods (namely, naïve Bayes, logistic regression, support vector machines, random k-nearest neighbor algorithm) and two ensemble learner methods (namely, AdaBoost and random subspace algorithm) were utilized.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVATIONS:

ML	Machine Learning
SMOTE	Synthetic Minority Over-sampling Technique
KEEL	Knowledge Extraction based on Evolutionary Learning
AUC	Area Under Curve
FCM	Fuzzy C-Means
VS code	Visual Studio Code

CHAPTER 1

INTRODUCTION

CLASS IMBALANCE IN MACHINE LEARNING

The Class imbalance is an important problem in machine learning, where the extremely small proportion of instances belonging to one class referred as, the minority class, whereas the extremely high proportion of instances of the other class or classes referred as, the majority class. Imbalanced datasets pose several challenges to conventional supervised learning methods. Some of the conventional supervised learning methods are support vector machines and decision trees can build viable classification models for balanced datasets. Since the imbalanced datasets suffer from majority and minority class, this skewed distribution of instances in labeled dataset may lead to degradation of performance of the model. The supervised learning process is based on the use of global evaluation measures such as classification accuracy. Hence, learning from imbalanced datasets can be biased towards the majority class, and classification models may tend to misclassify the instances of minority class. Supervised learning algorithms may regard the instances of minority class as noise or outlier, and noisy data and outlier may be regarded as the instances of minority class.

Imbalanced datasets can be encountered in several real-world problems and applications, including software fault identification, medical diagnosis, malware detection, anomaly identification, bankruptcy prediction, and spam filtering. For data mining problems mentioned in advance, the number of instances for minority class is scarce. However, the identification of the instances of minority class may be more critical. For instance, the misclassification of cancerous (malignant) tumors as noncancerous (benign) in medical diagnosis can have severe effects. Similarly, the number of instances for fraudulent transactions can be scarce. However, it is critical to build prediction models that can identify fraudulent transactions in finance. Hence, handling imbalanced datasets properly is an important research problem in machine learning.

To deal efficiently with the datasets with imbalanced distribution and to build robust and efficient classification schemes, data preprocessing methods have been utilized in conjunction with machine learning algorithms. The methods utilized to tackle with class imbalance problem can be mainly divided into four categories as algorithm level approaches, data-level approaches, cost-sensitive approaches, and ensemble learning-based approaches. Algorithm level approaches seek to adapt supervised learning algorithms to bias learning towards the instances of minority class. Data-level approaches seek to rebalance the instances of the imbalanced dataset so that the effects of skewed distributions can be eliminated in the learning process.

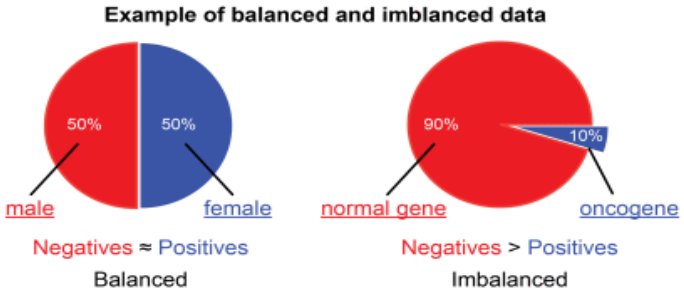

Fig-1.1 Example of Balanced and Imbalanced Data

In this project, we introduce an undersampling technique for imbalanced learning utilizing a consensus clustering-based approach. This method involves reducing the number of instances in the majority class through a consensus clustering strategy. While numerous clustering algorithms exist in the literature, no single algorithm universally provides optimal clustering results, as indicated by the no free lunch theorem. Consequently, the proposed approach seeks to amalgamate the outputs of diverse clustering algorithms to surpass the constraints of individual methods and attain more resilient and effective clustering outcomes. The consensus clustering schemes, five clustering algorithms (namely, *k*-means, *k*-modes, *k*-means++, Fuzzy *k*-mean, and mini-batch *k*-mean and their combinations were taken into consideration. In the classification phase, four supervised learning methods (namely, naïve Bayes, logistic regression, support vector machines and *k*-nearest neighbor algorithm) and two ensemble learner methods (namely random forests, AdaBoost algorithm) were utilized.

1.1MOTIVATION

The motivation of this project stems from the pervasive issue of class imbalance in machine learning, which poses significant challenges in various real-world applications such as medical diagnosis, fraud detection, and anomaly identification. In these contexts, accurately identifying minority class instances is often paramount, as errors can have severe consequences like misdiagnosing cancerous tumors or failing to detect fraudulent transactions.

Conventional supervised learning methods struggle to address imbalanced datasets, as they tend to favor the majority class, leading to poor predictive performance and misclassification of minority class instances. Additionally, the overlapping nature of minority class instances with other classes further complicates the learning process.

To tackle these challenges, various techniques have been developed, including algorithm-level approaches, data-level approaches, cost-sensitive approaches, and ensemble learning-based approaches. This project focuses specifically on data-level approaches, particularly undersampling, which involves reducing the number of majority class instances to balance the dataset. However, traditional undersampling methods may inadvertently discard important information from the majority class.

1.2 PROBLEM STATEMENT

The problem addressed in this project is the challenge posed by class imbalance in machine learning. Class imbalance occurs when one class (the minority class) is significantly underrepresented compared to others (the majority class or classes). Imbalanced datasets are common in various real-world applications such as medical diagnosis, fraud detection, and anomaly identification.

Conventional supervised learning methods often struggle with imbalanced datasets, as they tend to bias their predictions towards the majority class, leading to poor predictive performance and misclassification of minority class instances. Additionally, the overlapping nature of minority class instances with other classes further complicates the learning process.

To address these challenges, various techniques have been developed, including algorithm-level approaches, data-level approaches, cost-sensitive approaches, and ensemble learning-based approaches. This project focuses specifically on data-level approaches, particularly undersampling, which involves reducing the number of majority class instances to balance the dataset. However, traditional undersampling methods may inadvertently discard important information from the majority class.

The project aims to propose a novel solution to this problem: a consensus clustering-based undersampling approach. By leveraging the diversity of multiple clustering algorithms and combining their decisions, this approach seeks to identify better representative instances of the majority class while mitigating the risk of information loss. The goal is to enhance predictive performance on imbalanced datasets and address the limitations of existing undersampling techniques.

The effectiveness of the proposed approach will be evaluated through empirical analysis, comparing its performance against existing methods. The project seeks to contribute valuable insights to the field of imbalanced learning methodologies and provide a novel solution that holds promise for improving classification accuracy on imbalanced datasets.

1.3 PROJECT OBJCTIVE

The proposed solution in this project is a consensus clustering-based undersampling approach. By leveraging the diversity of multiple clustering algorithms and combining their decisions, this approach aims to identify better representative instances of the majority class while mitigating the risk of information loss. This innovative technique offers a promising solution to enhance predictive performance on imbalanced datasets.

The empirical analysis conducted in this project aims to validate the effectiveness of the proposed approach compared to existing methods. By demonstrating superior predictive performance through empirical evidence, this project contributes valuable insights to the field of imbalanced learning methodologies.

Overall, this project addresses a pressing research problem in machine learning and offers a novel solution that holds significant potential for improving classification accuracy on imbalanced datasets. By presenting empirical evidence of its effectiveness, this work contributes valuable insights and opens new avenues for future research in imbalanced learning methodologies.

1.4 PROJECT REPORT ORGANIZATION

The project addresses the pervasive challenge of class imbalance in real-world applications, where skewed class distributions often lead to biased models and subpar predictive performance. By leveraging consensus clustering-based undersampling, the proposed approach aims to rebalance class distributions by selecting a representative subset of majority class instances while preserving minority class instances. Through an extensive literature review, existing techniques are scrutinized, uncovering their limitations such as information loss and lack of robustness. The system architecture is delineated, highlighting the integration of undersampling with machine learning models for training and evaluation. Experimental results substantiate the effectiveness of the approach, demonstrating enhanced model performance across various datasets. The discussion section critically examines the approach's strengths and weaknesses, exploring its applicability in domains like fraud detection and medical diagnosis. Future research directions are identified to refine and extend the proposed method, emphasizing the need for ongoing efforts to tackle class imbalance in machine learning applications. Overall, the project underscores the significance of consensus clustering-based undersampling in fostering more reliable and unbiased models, with implications for both research and practical implementations in diverse real-world contexts.

CHAPTER-2
LITERATURE REVIEW

Anand et al. [1] proposed an approach that combines weighting and undersampling techniques. Weighting assigns higher importance to minority class instances, while undersampling reduces the number of majority class instances to balance the dataset. Kumar et al. [2] introduced an undersampling technique based on K-Means clustering, aiming to create a more balanced dataset by selecting representative samples from each cluster.

Lin et al. [3] presented a technique that utilizes clustering algorithms for undersampling imbalanced datasets. Instead of randomly selecting instances for undersampling, they group similar instances using clustering and select representative samples from each cluster. Shobana and Battula [4] extended this approach by diversifying the distribution of undersampled instances through K-Means clustering to better capture the minority class characteristics.

Guo and Wei [5] introduced a logistic regression approach for imbalanced learning, leveraging clustering techniques to capture data relationships, especially in imbalanced scenarios. Douzas et al. [6] proposed an oversampling technique combining K-Means clustering with SMOTE to refine the oversampling process.

Han et al. [7] presented a distribution-sensitive oversampling technique tailored for medical diagnosis applications. Tsai et al. [8] proposed a method combining clustering analysis and instance selection for undersampling imbalanced datasets, aiming to create a more balanced dataset suitable for classification tasks.

Li et al. [9] introduced an ensemble method for imbalanced datasets using Support Vector Machines (SVMs) combined with segmentation techniques. Barua et al. [10] proposed MWMOTE, a weighted oversampling technique for learning from imbalanced datasets.

D'Addabbo and Maglietta [11] introduced a parallel selective sampling method for classification tasks involving imbalanced and large datasets. Ha and Lee [12] proposed a novel under-sampling method based on genetic algorithms (GA) for imbalanced data classification. Sun et al. [13] presented a novel ensemble method for classifying imbalanced data.

2.1 EXISTING WORK

Several methodologies have been proposed to address the challenges of imbalanced data in classification tasks. Shobana and Battula proposed an undersampled k-means approach, aiming to handle imbalanced data by diversifying distribution. Their focus was on enhancing the performance of k-means clustering specifically for imbalanced datasets. Guo and Wei introduced a logistic regression approach for imbalanced learning, incorporating clustering analysis to improve predictive performance. Their research aimed to enhance the effectiveness of logistic regression by leveraging insights from clustering.

Douzas presented a heuristic oversampling method combining k-means clustering with SMOTE to improve imbalanced learning. Their study focused on generating synthetic instances for the minority class through clustering and oversampling techniques. Han's work proposed a distribution-sensitive oversampling method for unbalanced data, particularly in the context of medical diagnosis. By integrating clustering analysis with oversampling, their approach aimed to address class imbalance effectively in medical datasets.

Tsai's research introduced an undersampling approach that combined clustering analysis with instance selection to handle class imbalance. Their study aimed to enhance the performance of undersampling methods by considering the distribution of instances within the majority class. These methodologies collectively contribute to the ongoing efforts to tackle imbalanced data challenges in classification tasks, offering various strategies that leverage clustering and oversampling techniques.

2.2 LIMITATIONS OF EXISTING WORK

The existing work on consensus clustering-based undersampling for imbalanced learning confronts several significant challenges. The scarcity of minority class instances within imbalanced datasets poses a fundamental problem, hindering the effectiveness of data-level approaches aimed at balancing the dataset.

This scarcity undermines the potential of such methods to address class imbalance adequately. Accurate identification of minority class instances is paramount, particularly in critical domains such as medical diagnosis, where the consequences of misclassification can be severe.

However, existing methods may fall short in precisely identifying and managing these minority class instances, raising concerns regarding their reliability in practical applications. While the presented approach incorporates five clustering algorithms, there remains a limitation in exploring the full spectrum of available clustering options.

This shortfall suggests a potential missed opportunity for leveraging other clustering algorithms, including conventional and swarm-based ones, which could potentially enhance predictive performance in imbalanced learning scenarios. Recent suggestions highlight the potential for higher predictive performance through the combination of instance selection and clustering methods. However, the current scheme may not fully capitalize on this opportunity, potentially overlooking avenues for improving performance.

There is a clear need for further extension and consideration in the field of imbalanced learning. Issues such as exploring the effectiveness of consensus clustering-based undersampling alongside traditional instance selection methods require deeper investigation to enhance the overall understanding and effectiveness of the proposed methodologies. Addressing these challenges is crucial for advancing the field and improving the practical applicability of imbalanced learning techniques.

CHAPTER 3

REQUIREMENT ANALYSIS

	Hardware	Software
Developers	1. 4 GB RAM 2. 256 GB Storage 3. Intel i5 5th Gen + Processor	1. OS – Windows 7/8/10 2. Python3 3. Jupyter Notebook 4. Visual Studio Code

Table-3.1 System Requirements

3.1 SOFTWARE REQUIREMENTS

3.1.1 PYTHON

Python was the language of selection for this project. This was a straightforward call for many reasons.

1. Python as a language has a vast community behind it. Any problems which may be faced is simply resolved with a visit to Stack Overflow. Python is among the foremost standard language on the positioning that makes it very likely there will be straight answer to any question.
2. Python has an abundance of powerful tools prepared for scientific computing Packages like NumPy, Pandas and SciPy area unit freely available and well documented. Packages like these will dramatically scale back, and change the code required to write a given program. This makes iteration fast.
3. Python is scripting language most widely used for Machine Learning algorithms.
4. Python as a language is forgiving and permits for program that appear as if pseudo code. This can be helpful once pseudo code given in tutorial papers must be enforced and tested. Using python this step is sometimes trivial. However, Python is not without its errors. The language is dynamically written, and packages are area unit infamous for Duck writing. This may be frustrating once a package technique returns one thing that, for instance, looks like an array instead of being an actual array. Plus, the actual fact that standard Python documentation

does not clearly state the return type of a method, this can lead to a lot of trials and error testing that will not otherwise happen in a powerfully written language. This is a problem that produces learning to use a replacement Python package or library more difficult than it otherwise may be.

3.1.2 JUPYTER

Jupyter is an open-source web application that enables users to create and share documents containing live code, equations, visualizations, and narrative text.

1. It supports multiple programming languages such as Python, R, Julia, and Scala, making it versatile for various data science and scientific computing tasks.

2. Jupyter provides an interactive computing environment through its user-friendly interface, combining code execution, text, and visualizations in a single document.

3. Collaboration is facilitated through features like version control integration with Git, allowing multiple users to work on the same notebook simultaneously.

4. Jupyter notebooks can be easily shared via email, GitHub, or the Jupyter Notebook Viewer, promoting reproducible research and knowledge dissemination.

5. It integrates seamlessly with popular data science libraries and frameworks, enabling data exploration, analysis, and machine learning.

6. Jupyter is widely used in education for teaching programming, data science, and computational science, providing an interactive environment for students to experiment with code.

7. It is also prevalent in academic research for prototyping algorithms, conducting experiments, analyzing data, and sharing results with collaborators and the broader scientific community.

8. The Jupyter community is vibrant and diverse, contributing to the project's development, documentation, and outreach efforts.

3.1.3 VISUAL STUDIO CODE

Visual Studio Code (VSCode) is a free, open-source source code editor developed by Microsoft. It's a lightweight and highly customizable code editor that supports a wide range of programming languages.

Some key **features** of **Visual Studio Code**:

Cross-Platform: VSCode is available for Windows, macOS, and Linux, making it a versatile choice for developers using different operating systems.

Extensions: One of the standout features of VSCode is its extensive support for extensions. There is a rich marketplace where developers can find and install extensions for various languages, themes, debuggers, and more, allowing users to tailor their development environment.

Intelligent Code Completion: VSCode provides intelligent code completion suggestions as you type, helping developers write code more efficiently.

Integrated Git Support: The editor comes with built-in Git support, enabling version control operations without needing an external tool.

Debugging: VSCode supports debugging for a variety of languages and provides a visual debugger interface, making it easier to identify and fix issues in the code.

Built-in Terminal: It includes an integrated terminal, allowing developers to run command-line tools and scripts without leaving the editor.

Syntax Highlighting and IntelliSense: VSCode provides syntax highlighting for a wide range of languages and offers IntelliSense, which provides context-aware code suggestions.

Customizable Themes: Users can customize the appearance of VSCode by choosing from a variety of themes available in the marketplace.

Task Automation: It supports task automation and includes a task runner for common build systems.

Collaboration: VSCode has features to support collaborative development, such as Live Share, allowing multiple developers to work on the same codebase simultaneously.

VSCode has gained widespread popularity in the developer community due to its performance, versatility, and the active support and updates provided by Microsoft. It's particularly well-suited for web development, but it supports a wide array of languages and frameworks, making it a popular choice for various types of development projects.

3.2 HARDWARE REQUIREMENTS

Processor: Intel i2 or more.

RAM: 4 GB or more.

Cache: 512 KB.

Hard Disk: 16 GB hard disk recommended.

Disk Drive:1.44MB Floppy Disk Drive.

CHAPTER 4

SYSTEM DESIGN

4.0 Proposed System architecture

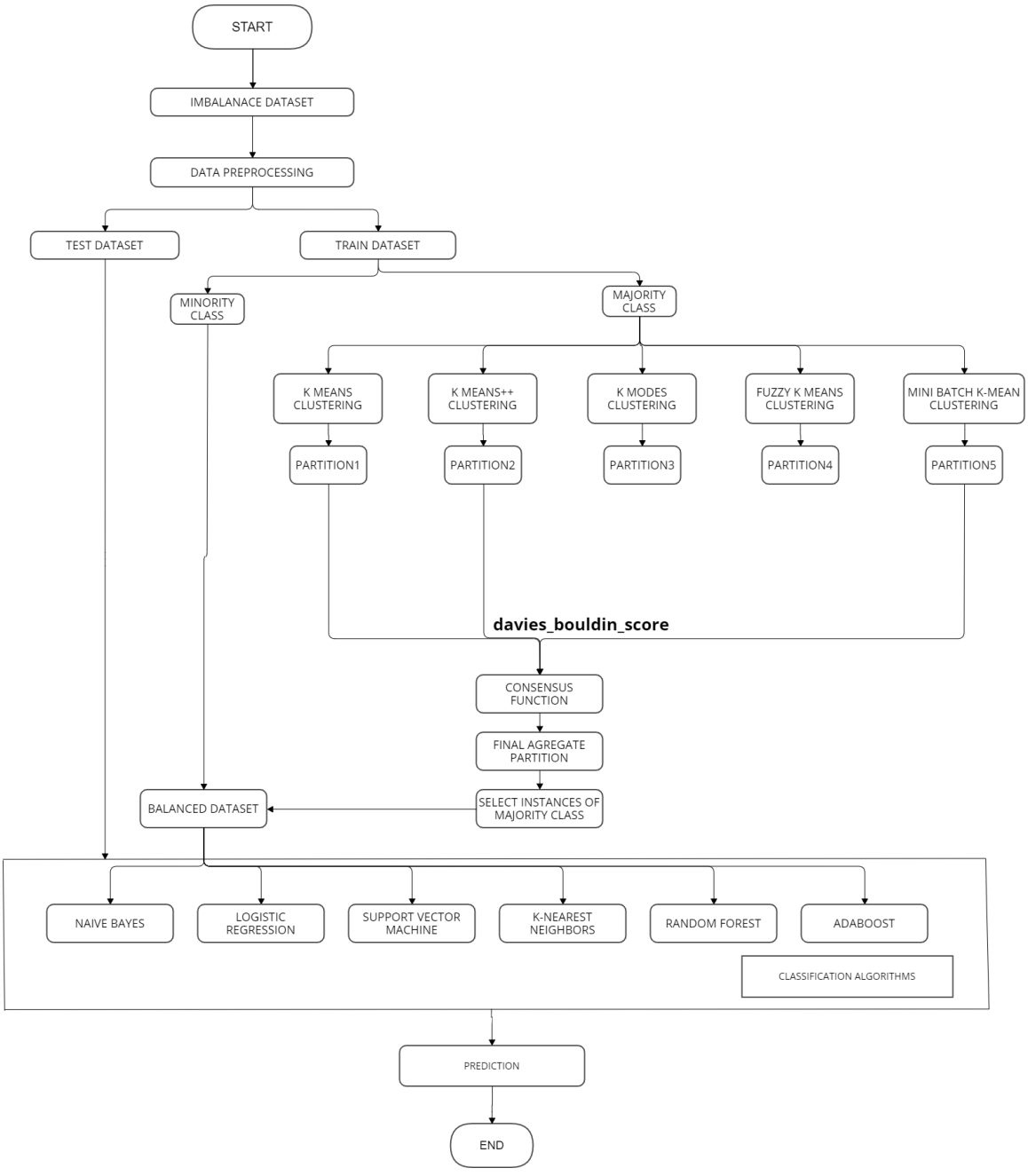

Fig-4.0 Proposed System Architecture

4.1Proposed methods

The proposed system architecture leverages consensus clustering-based under sampling to address imbalanced learning. By integrating five clustering algorithms, it aims to enhance predictive performance by combining diverse perspectives on the data structure. This approach could potentially lead to a more effective model, particularly in scenarios where traditional methods struggle due to imbalanced class distributions. The success of this architecture would likely depend on the characteristics of the specific dataset and the nature of the problem being addressed.

1. Identify Imbalanced Dataset:

The first step is to identify the imbalanced dataset. This means recognizing that the dataset has a majority class and a minority class, and that the majority class has significantly more instances than the minority class.

2. Split Data into Training and Testing Sets:

The next step is to split the imbalanced dataset into training and testing sets. The training set will be used to train the classification model, while the testing set will be used to evaluate the model's performance.

3. Under sample Majority Class using Consensus Clustering:

Applying Multiple Clustering Algorithms:

The algorithm applies multiple clustering algorithms to the majority class data. These algorithms partition the data into different clusters.

The Base clustering algorithms are

3.1. k-means:

It is a classic partitioning algorithm that employs an iterative process to cluster data points into predefined clusters. It minimizes the within-cluster variance by assigning each data point to the cluster with the closest centroid, which is the mean of the feature values. The algorithm starts with random initial centroids and iteratively refines them until convergence. It involves the following steps: assigning data points to the nearest

cluster centroid, recalculating centroids, and repeating until convergence. The result is clusters with centroids that represent the mean feature values of the assigned data points.

3.2. k-means++:

It is an enhancement to the standard k-means algorithm, specifically addressing the issue of initial centroid selection. It aims to improve convergence by initializing cluster centroids in a way that increases their separation, reducing the risk of convergence to local optima. The k-means++ initialization involves choosing the first centroid randomly and subsequent centroids based on their distance from existing centroids. This initialization process contributes to a more robust start for the k-means algorithm, potentially leading to better final clusters.

3.3. k-modes:

It is designed for clustering categorical data, providing an alternative to k-means when features are not continuous. It uses the mode (most frequent value) as the cluster center. The algorithm iteratively updates cluster centroids by selecting the mode of each categorical feature within the cluster. This mode-based approach is well-suited for datasets where features are discrete and categorical.

3.4. Fuzzy k-means:

It is a soft clustering algorithm allowing data points to belong to multiple clusters simultaneously, with varying degrees of membership. Unlike traditional k-means, which assigns each point to a single cluster, It introduces membership values that represent the degree of belonging to each cluster. These membership values are iteratively updated to minimize the overall variance.

3.5. Mini-batch k-means:

It is a variant designed for efficiency on large datasets by processing data in smaller batches rather than the entire dataset at once. Instead of updating centroids using the entire dataset, mini-batch k-means updates them based on randomly sampled subsets (batches) of the data. This approach makes the algorithm faster and more scalable,

making it suitable for large datasets where memory constraints may be an issue. However, it may sacrifice a bit of accuracy compared to standard k-means.

Combining Partitions using Consensus Function: The partitions generated by the different clustering algorithms are then combined using a consensus function. This function identifies a final, more robust partition of the majority **Simple voting:** Each algorithm's cluster assignment for a data point is counted, and the most frequent cluster wins.

Centroid-based selection: Select instances closest to the cluster centroid.

4. Train Ensemble Classification Scheme:

The balanced dataset obtained from undersampling is then used to train an ensemble classification scheme. This means that multiple classification learners are trained on the data, and their predictions are combined to improve accuracy.

5. Evaluate Performance:

Evaluate the performance of the final ensemble model on the held-out testing set. Consider metrics like accuracy, precision, recall, and F1-score depending on your classification task.

4.2 Use case Diagrams

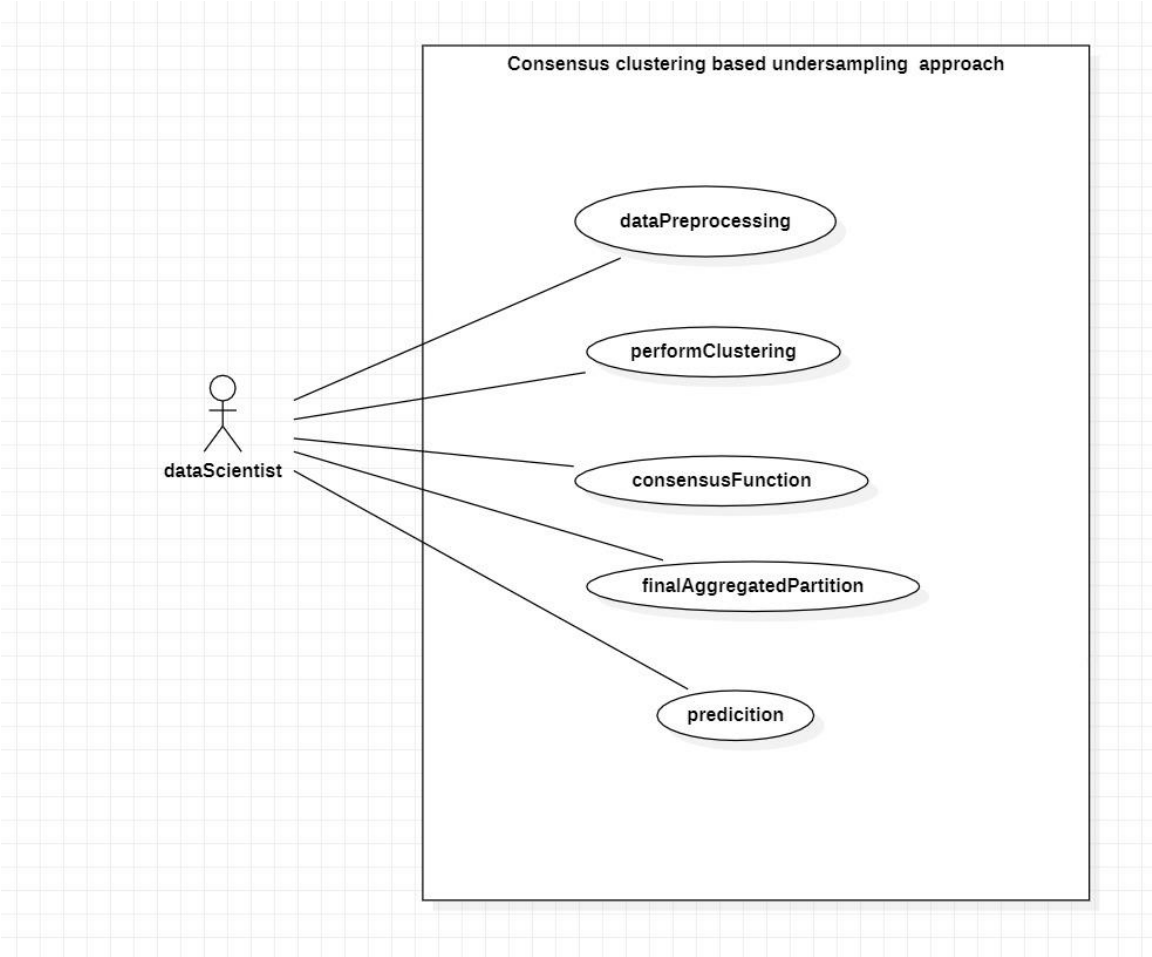

Fig-4.2 Use case diagram

The above use case diagram illustrates a comprehensive data analysis and machine learning workflow, encompassing key components such as data preprocessing, clustering, consensus function application, and final aggregation. Data preprocessing involves cleaning and organizing raw data to make it suitable for analysis, while clustering groups similar data points together based on specific features. The consensus function plays a crucial role in combining results from multiple clustering algorithms or runs to enhance clustering robustness. Finally, the final aggregation step synthesizes clustering outcomes into a cohesive solution, facilitating the extraction of meaningful insights or patterns.

4.3 Datasets and Technology stack

Datasets

- We have taken two datasets one is from the base paper and other dataset from Kaggle.
- The datasets mentioned in the reference paper are from KEEL (Knowledge Extraction based on Evolutionary Learning)-dataset repository. This repository contains preprocessed and relabeled datasets which helps in performing and evaluating various projects.

Car Evaluation Dataset:

This a car evaluation dataset taken from the Kaggle dataset source. These are features and of this car evaluation dataset.

	buying	maint	doors	persons	lug_boot	safety	class
0	vhigh	vhigh	2	2	small	low	Bad
1	vhigh	vhigh	2	2	small	med	Bad
2	vhigh	vhigh	2	2	small	high	Bad
3	vhigh	vhigh	2	2	med	low	Bad
4	vhigh	vhigh	2	2	med	med	Bad

Yeast Dataset:

This is a yeast dataset taken from Keel-dataset repository mainly for the class imbalance problem.This dataset is preprocessed and relabeled with class negative and positive.

	Mcg	Gvh	Alm	Mit	Erl	Pox	Vac	Nuc	Class
0	0.58	0.61	0.47	0.13	0.5	0.0	0.48	0.22	negative
1	0.43	0.67	0.48	0.27	0.5	0.0	0.53	0.22	negative
2	0.58	0.44	0.57	0.13	0.5	0.0	0.54	0.22	negative
3	0.42	0.44	0.48	0.54	0.5	0.0	0.48	0.22	negative
4	0.51	0.40	0.56	0.17	0.5	0.5	0.49	0.22	negative

Pima Dataset:

Pima Dataset is Taken form keel-dataset repository for the class imbalance problem. This dataset is preprocessed.

	Preg	Plas	Pres	Skin	Insu	Mass	Pedi	Age	Class
0	14.0	175.0	62.0	30.0	0.0	33.6	0.212	38.0	positive
1	4.0	146.0	78.0	0.0	0.0	38.5	0.520	67.0	positive
2	15.0	136.0	70.0	32.0	110.0	37.1	0.153	43.0	positive
3	5.0	116.0	74.0	29.0	0.0	32.3	0.660	35.0	positive
4	6.0	0.0	68.0	41.0	0.0	39.0	0.727	41.0	positive

Technologies Used

4.3.1 Python

Python was the language of selection for this project. This was a straightforward call for many reasons.

1. Python as a language has a vast community behind it. Any problems which may be faced is simply resolved with a visit to Stack Overflow. Python is among the foremost standard language on the positioning that makes it very likely there will be straight answer to any question.
2. Python has an abundance of powerful tools prepared for scientific computing Packages like NumPy, Pandas and SciPy area unit freely available and well documented. Packages like these will dramatically scale back, and change the code required to write a given program. This makes iteration fast.
3. Python as a language is forgiving and permits for program that appear as if pseudo code. This can be helpful once pseudo code given in tutorial papers must be enforced and tested. Using python this step is sometimes trivial. However, Python is not without its errors. The language is dynamically written, and packages are area unit infamous for Duck writing. This may be frustrating once a package technique returns one thing that, for instance, looks like an array instead of being an actual array. Plus, the fact that standard Python documentation does not clearly state the return type of a method, this can lead to a lot of trials and error testing that will not otherwise happen in a powerfully written language. This is a problem that produces learning to use a replacement Python package or library more difficult than it otherwise may be.

4.3.2 Visual Studio Code

Visual Studio Code (VSCode) is a free, open-source source code editor developed by Microsoft. It's a lightweight and highly customizable code editor that supports a wide range of programming languages.

Here are some key features of Visual Studio Code:

Cross-Platform: VSCode is available for Windows, macOS, and Linux, making it a versatile choice for developers using different operating systems.

Extensions: One of the standout features of VSCode is its extensive support for extensions. There is a rich marketplace where developers can find and install extensions for various languages, themes, debuggers, and more, allowing users to tailor their development environment.

Intelligent Code Completion: VSCode provides intelligent code completion suggestions as you type, helping developers write code more efficiently.

Integrated Git Support: The editor comes with built-in Git support, enabling version control operations without needing an external tool.

Debugging: VSCode supports debugging for a variety of languages and provides a visual debugger interface, making it easier to identify and fix issues in the code.

Built-in Terminal: It includes an integrated terminal, allowing developers to run command-line tools and scripts without leaving the editor.

Syntax Highlighting and IntelliSense: VSCode provides syntax highlighting for a wide range of languages and offers IntelliSense, which provides context-aware code suggestions.

Customizable Themes: Users can customize the appearance of VSCode by choosing from a variety of themes available in the marketplace.

Task Automation: It supports task automation and includes a task runner for common build systems.

Collaboration: VSCode has features to support collaborative development, such as Live Share, allowing multiple developers to work on the same codebase simultaneously.

CHAPTER 5

IMPLEMENTATION

5.1 SCREENSHOTS

5.1.1 Generating Clustering methods:

Five clustering algorithms are performed on majority class instances of the dataset

1. K-mean:

```
kmeans=KMeans(n_clusters=7,init='random',max_iter=500).fit(class_majority_df)
clustering_result['Kmeans']=kmeans.labels_
```

2. K-mean++:

```
kmeans=KMeans(n_clusters=num_of_clusters,init='k-means++',max_iter=500).fit(class_majority_df)
clustering_result['Kmeans++']=kmeans.labels_
```

3. K-mode:

```
from kmodes.kmodes import KModes
km = KModes(n_clusters=num_of_clusters, init='Huang', n_init=5, verbose=1)
clusters = km.fit_predict(class_majority_df)
df_0_centroids=km.cluster_centroids_
clusters
```

4. Mini Batch K-mean:

```
from sklearn.cluster import MiniBatchKMeans
mbk = MiniBatchKMeans(init='k-means++', n_clusters=num_of_clusters, batch_size=40, n_init=10, max_
mbk.fit(class_majority_df)
clustering_result['Minibatch']=mbk.labels_
```

5. Fuzzy K-mean:

```
fcm = FCM(n_clusters=num_of_clusters)
fcm.fit(class_majority_df.values)
# outputs
fcm_centers = fcm.centers
fcm_labels = fcm.predict(class_majority_df.values)
fcm_labels
```

5.1.2 Davies Bound Score

```
Kmeans (partition 1):  0.5128109791005412
Kmeans++ (partition 2):  0.5145712285906867
KModes (partition 3):  55.514283051714415
Minibatch Kmeans (partition 4):  0.5146585683464967
Fuzzy K-means(partition 5):  0.515174152307507
```

5.1.3 Consensus Function

```
#Majority Voting

C = []
for i in range(len(new_clustering_result)):
  l = new_clustering_result.iloc[i]
  l = list(l[1:])
  majority_label = max(set(l), key = l.count)
  C.append(majority_label)

new_clustering_result['C'] = C
new_clustering_result
```

5.2 RESULTS

5.2.1 Results of Car Evaluation Dataset

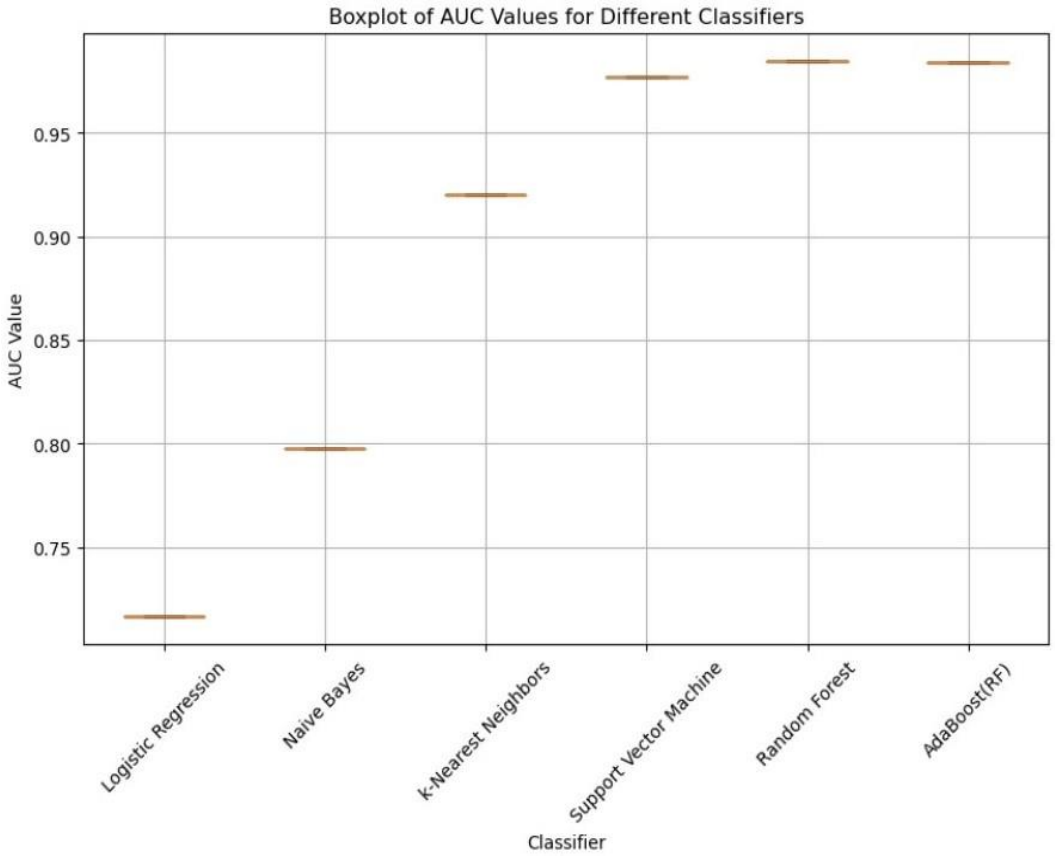

Fig-5.2.1 AUC boxplots of Car Evaluation dataset over different classifiers

The x-axis lists the different classifiers used in the experiment. These could be algorithms like Logistic Regression, Naive Bayes, k-Nearest Neighbors, Support Vector Machine, Random Forest, and AdaBoost (applied to Random Forest). And y-axis represents the AUC score achieved by each classifier. A higher AUC value indicates better performance, with a perfect score of 1 corresponding to a model that flawlessly separates positive from negative instances. Among the AUC values for different classifiers used in Car Evaluation Dataset, both Random Forest and AdaBoost (RF) has the Highest AUC value and the least AUC value among all classifiers is for Logistic Regression.

5.2.2 Results of Yeast Dataset

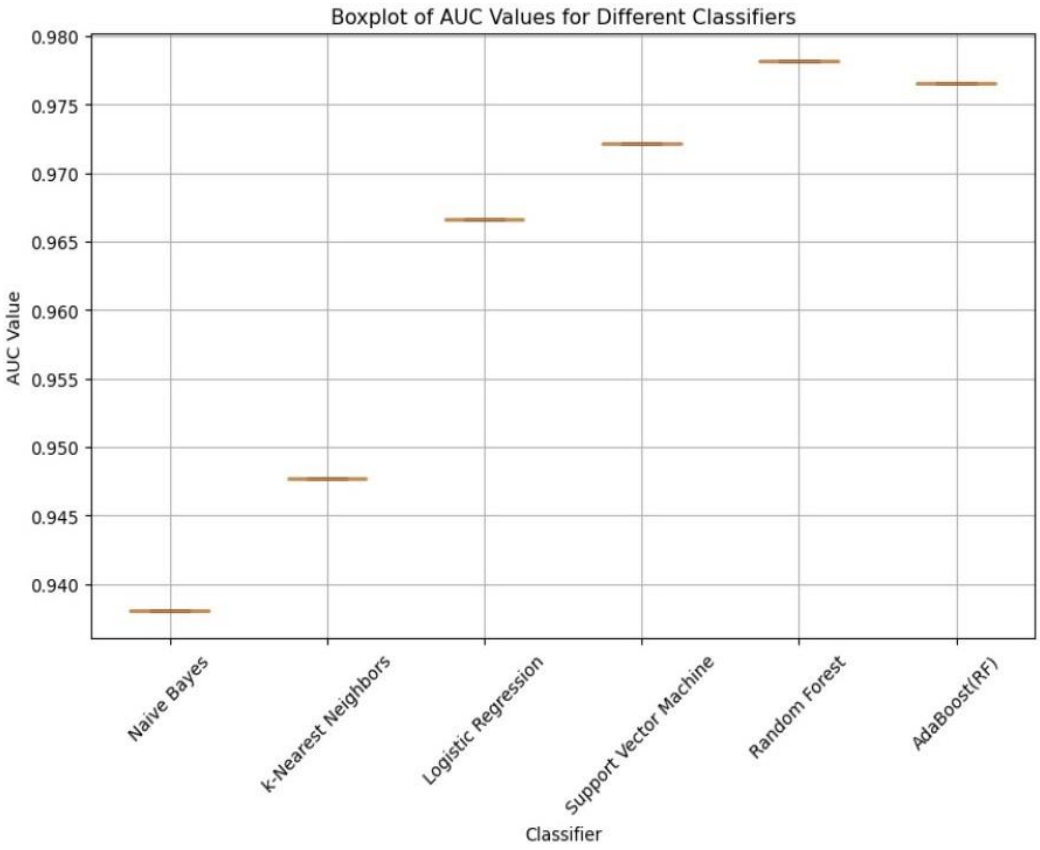

Fig-5.2.2 AUC boxplots of Yeast dataset over different classifiers

The Boxplot of AUC Values for Different Classifiers on the Yeast Dataset where on x-axis there are different classifiers like Naïve Bayes, K-Nearest Neighbors, Logistic Regression, Support Vector Machine, random Forest and AdaBoost (RF) and on the y-axis, there are AUC scores of those classifiers. The AUC score for the Random Forest is Highest and least for the Naïve Bayes. The curve looks like a logistic growth curve from naïve bayes to AdaBoost.

5.2.3 Results of Pima Dataset

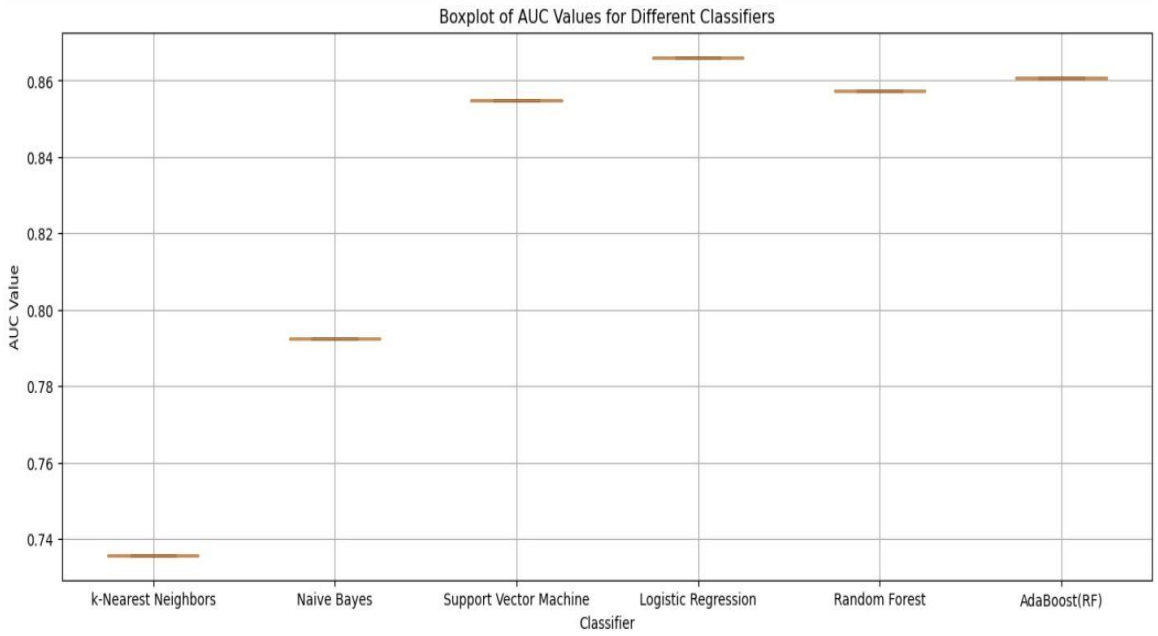

Fig-5.2.3 AUC boxplots of Pima dataset over different classifiers

The x-axis lists the different classifiers used in the experiment. These could be algorithms like Logistic Regression, Naive Bayes, k-Nearest Neighbors, Support Vector Machine, Random Forest, and AdaBoost (applied to Random Forest). The y-axis represents the AUC score achieved by each classifier. A higher AUC value indicates better performance, with a perfect score of 1 corresponding to a model that flawlessly separates positive from negative instances. The AUC score for the Logistic Regression is Highest With 0.866 and least with 0.735 for the K-Nearest Neighbors in Pima Dataset. We have a both logistic regression and AdaBoost with closest values 0.86.

5.2.4 Results of Datasets

With Consensus Clustering based Undersampling Approach							Without Consensus Clustering based Undersampling Approach
Dataset	Navies Bayes	K-nearest Neighbour	Logistic Regression	Support Vector Machine	Random Forest	Adaboost	C4.5 (Decision Tree)
Yeast	0.93801	0.94765	0.96659	0.9721	**0.9814**	0.97658	0.883
Pima	0.79237	0.73572	**0.86613**	0.85496	0.85735	0.86063	0.701
Car Evaluation	0.79694	0.91504	0.71242	0.97555	**0.98127**	0.98524	0.823

Table-5.2.4 Results of Dataset

The above table consists of results on Yeast Dataset and Car Evaluation Dataset before consensus Clustering based Undersampling Approach and after consensus clustering based Undersampling approach method where the Average AUC values before applying the consensus function is less than the average AUC values after applying the consensus function. The average AUC values of Random forest on yeast Dataset is greater after applying consensus function i.e. 0.9814, The AUC value before applying the consensus function was 0.883. The average AUC values of Logistic Regression on Pima Dataset is greater after applying consensus function (i.e. 0.86613) when compared to the AUC values on Logistic Regression on Pima Dataset before applying the consensus function (i.e. 0.701). The average AUC value of AdaBoost on Car Evaluation Dataset is greater i.e. 0.98524.

CHAPTER 6

CONCLUSION

In conclusion, this project emphasizes the significance of addressing class imbalance in machine learning and proposes consensus clustering-based undersampling schemes as a promising solution. Empirical analysis indicates that these schemes outperform traditional preprocessing methods for imbalanced datasets. Future research should explore the integration of different clustering algorithms, consensus functions, and instance selection techniques to further enhance predictive performance. Additionally, evaluating these methods on larger datasets and real-world applications would provide valuable insights into their effectiveness and scalability. Overall, consensus clustering-based undersampling shows potential for improving classification performance in imbalanced learning scenarios.

6.1 Future Scope

This project encompasses several avenues for further exploration and enhancement. Firstly, extending the analysis to include additional clustering algorithms, such as ant clustering, particle swarm-based clustering, and firefly clustering, would provide a broader understanding of their effectiveness in addressing class imbalance. Moreover, investigating alternative consensus functions beyond simple voting-based and incremental voting functions could offer insights into optimizing the aggregation of clustering results. Secondly, integrating consensus clustering-based undersampling schemes with instance selection methods holds promise for improving predictive performance further. Exploring the synergies between these approaches could lead to more robust solutions for imbalanced learning tasks. Additionally, scaling up the evaluation to encompass larger datasets and real-world applications would validate the scalability and practical relevance of the proposed techniques.

Finally, considering emerging advancements in the field, such as deep learning-based approaches for imbalanced data, and exploring their integration with consensus clustering-based undersampling could open up new avenues for research and innovation in addressing class imbalance effectively. Overall, the future scope of this project involves refining existing methodologies, exploring novel techniques, and validating their applicability in diverse contexts to advance the state-of-the-art in imbalanced learning.

REFERENCES

[1] A. Anand, G. Pugalenthi, G. B. Fogel, and P. N. Suganthan, “An approach for classification of highly imbalanced data using weighting and undersampling,” Amino Acids, vol. 39, no. 5, pp. 1385–1391, 2010.

[2] N. S. Kumar, K. N. Rao, A. Govardhan, K. S. Reddy, and A. M. Mahmood, “Undersampled K-means approach for handling imbalanced distributed data,” Progress in Artificial Intelligence, vol. 3, no. 1, pp. 29–38, 2014.

[3] W.-C. Lin, C.-F. Tsai, Y.-H. Hu, and J.-S. Jhang, “Clusteringbased undersampling in class-imbalanced data,” Information Sciences, vol. 409-410, pp. 17–26, 2017.

[4] G. Shobana and B. P. Battula, “An under sampled k-means approach for handling imbalanced data using diversified distribution,” International Journal of Engineering and Technology (UAE), vol. 7, no. 1.8, pp. 113–117, 2018.

[5] H. Guo and T. Wei, “Logistic regression for imbalanced learning based on clustering,” International Journal of Computational Science and Engineering, vol. 18, no. 1, pp. 54–64, 2019.

[6] G. Douzas, F. Bacao, and F. Last, “Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE,” Information Sciences, vol. 465, pp. 1–20, 2018.

[7] W. Han, Z. Huang, S. Li, and Y. Jia, “Distribution-sensitive unbalanced data oversampling method for medical diagnosis,” Journal of medical Systems, vol. 43, no. 2, p. 39, 2019.

[8] C.-F. Tsai, W.-C. Lin, Y.-H. Hu, and G.-T. Yao, “Undersampling class imbalanced datasets by combining clustering analysis and instance selection,” Information Sciences, vol. 477, pp. 47–54, 2019.

[9] Q. Li, B. Yang, Y. Li, N. Deng, and L. Jing, “Constructingsupport vector machine ensemble with segmentation for imbalanced datasets,” Neural Computing and Applications, Vol. 22, no. S1, pp. 249–256, 2013.

[10] S. Barua, M. M. Islam, X. Yao, and K. Murase, “MWMOTE—majority weighted minority oversampling technique for imbalanced data set learning,” IEEE Transactions on Knowledge and Data Engineering, vol. 26, no. 2, pp. 405–425, 2014.

[11] A. D’Addabbo and R. Maglietta, “Parallel selective sampling method for imbalanced and large data classification,” Pattern Recognition Letters, vol. 62, pp. 61–67, 2015.

[12] J. Ha and J. S. Lee, “A new under-sampling method using genetic algorithm for imbalanced data classification,” in Proceedings of the 10th International Conference on Ubiquitous Information Management and Communication, p. 95, January 2016.

[13] Z. Sun, Q. Song, X. Zhu, H. Sun, B. Xu, and Y. Zhou, “A novel ensemble method for classifying imbalanced data,” Pattern Recognition, vol. 48, no. 5, pp. 1623–1637, 2015.