Journal of Big Data

METHODOLOGY Open Access

The use of class imbalanced learning methods on ULSAM data to predict the case–control status in genome-wide association studies

R. Onur Öztornaci[1,2*], Hamzah Syed[1,3,4], Andrew P. Morris[5,6] and Bahar Taşdelen[2]

*Correspondence:
onur.oztornaci@gmail.com

[1] Koç University Research Centre for Translational Medicine, Koç University, Istanbul, Turkey
[2] Faculty of Medicine, Department of Biostatistics and Medical Informatics, Mersin University, Mersin, Turkey
[3] GOSGene, Genetics and Genomic Medicine, UCL GOS Great Ormond Street Institute of Child Health, University College London, London, UK
[4] Great Ormond Street Hospital NHS Foundation Trust and UCL Great Ormond Street Institute of Child Health, NIHR Great Ormond Street Hospital Biomedical Research Centre, London, UK
[5] Division of Musculoskeletal and Dermatological Sciences, University of Manchester, Manchester, UK
[6] Department of Health Data Science, University of Liverpool, Liverpool, UK

Abstract

Machine learning (ML) methods for uncovering single nucleotide polymorphisms (SNPs) in genome-wide association study (GWAS) data that can be used to predict disease outcomes are becoming increasingly used in genetic research. Two issues with the use of ML models are finding the correct method for dealing with imbalanced data and data training. This article compares three ML models to identify SNPs that predict type 2 diabetes (T2D) status using the Support vector machine SMOTE (SVM SMOTE), The Adaptive Synthetic Sampling Approach (ADASYN), Random under sampling (RUS) on GWAS data from elderly male participants (165 cases and 951 controls) from the Uppsala Longitudinal Study of Adult Men (ULSAM). It was also applied to SNPs selected by the SMOTE, SVM SMOTE, ADASYN, and RUS clumping method. The analysis was performed using three different ML models: (i) support vector machine (SVM), (ii) multilayer perceptron (MLP) and (iii) random forests (RF). The accuracy of the case–control classification was compared between these three methods. The best classification algorithm was a combination of MLP and SMOTE (97% accuracy). Both RF and SVM achieved good accuracy results of over 90%. Overall, methods used against unbalanced data, all three ML algorithms were found to improve prediction accuracy.

Keywords: Machine learning, Class imbalanced methods, GWAS, ULSAM study

Introduction

Genome-wide association studies (GWAS) are the most used strategy for investigating the genetic architecture of common complex diseases. GWAS methodology examines the differences in allele frequencies between individuals affected and unaffected by the disease at single nucleotide polymorphisms (SNPs) across the genome. GWAS data can be used for predicting disease status and classification of cases from controls using machine learning (ML) algorithms. The use of ML algorithms for prediction is increasingly being used over methodology such as polygenic risk scores [1, 2]. The application of ML algorithms have some advantages in this situation over classical statistical methodology such as the logistic regression model; (i) for the classification of cases and controls,

Springer Open

© The Author(s) 2023. **Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

where a large number of possibly correlated variables may be modelled together; (ii) a significance threshold is not required; (iii) gene–gene interactions can be evaluated and (iv) assumptions based on normality and homogeneity of variances are not needed [3, 4].

GWAS data are suitable for ML data mining because large amounts of phenotype and genomic data can be studied simultaneously. Large complex GWAS data summarised with data mining approaches can be interpreted quickly and explained by graphics such as decision tree plots. Following the ML procedure, the selection of an appropriate statistical method is necessary to ensure time and cost savings. Methods that use classification are at a disadvantage due to their long calculation time, and increased likelihood of over-fitting and under-fitting. Under-fitting occurs when the model cannot find the intricate patterns hidden in the data. Whereas, when the model learns from large quantities of data, over-fitting is more likely to occur. The likelihood of over-fitting is increased when the dataset is imbalanced [4, 5].

Imbalanced data is defined as the category/outcome variable used for classification having unequal classes or differing number of observations. Class imbalance is a problem that arises frequently when applying GWAS for diseases with low prevalence in large cohorts. In this case, it may be a solution to use adjusted the type-I error when using classical statistical methods such as the logistic regression model. However, ML algorithms are not immune to class imbalance. Therefore, datasets need to be balanced with statistically unbiased resampling techniques such as (Synthetic Minority Over-sampling Technique (SMOTE). [6, 7].

GWAS typically interrogate millions of SNPs, many of which are in strong linkage disequilibrium (LD) with each other, which poses considerable computational challenges in applying ML methods. LD may affect the classification performance. LD-clumping use to remove highly correlated SNPs, may be helpful for increasing classification performance on ML. Clumping is a procedure within PLINK that selects the most significant SNPs in LD blocks. By doing this, the data size and the correlation between remaining SNPs is reduced [8]. Using clumping with ML is advantageous as it reduces processing time and it is a proven and commonly used procedure for applying ML in GWAS [9, 10].

In this paper, ML approaches were applied to a GWAS of type 2 diabetes (T2D) undertaken in the Uppsala Longitudinal Study of Adult Men (ULSAM), which includes 165 cases and 951 controls of European ancestry from Sweden. The objectives of this study are to examine the accuracy, specificity, and sensitivity of three different machine learning approaches and to compare the results. Furthermore, we applied LD clumping in both situations with/without SMOTE and compared the results.

Material and methods

When performing GWAS, the entire genome is analysed, to identify SNPs that may predispose a disease of interest. While logistic regression is used for dependent variables in a binary setting, linear regression methods are used for variables in a continuous setting. The datasets used for genetic association studies are high dimensional [11]. Sample size is very influential for classical statistical approaches like the logistic regression model because a small number of individuals (n) and/or a large number of SNPs (feature, p) can be a problem. The $(X^{T}X)$ parameters w that are used for predicting beta coefficients will not be a singular matrix and the parameters in the regression model cannot be uniquely

estimated [12]. However, ML algorithms are less affected by this problem than the classical multivariate approaches because ML algorithms have tuning parameters to optimise and regulate the process. An example of this are penalized likelihood methods [13, 14]. Another issue for both classical and ML models is imbalanced data, where the data collected between groups of samples are different [15]. Most often, LD clumping has been used when applying machine learning to GWAS data [16, 17]. Using clumping has benefits such as reducing calculation time and memory usage. It can be used to eliminate unrelated SNPs that provide redundant information using a defined threshold of LD (usually r^2) [17, 18]. In this case, the majority class is more likely to be correctly classified than the minority class. In a classification problem, the class that has fewer elements than the other classes are called the minority class. Bias is the main contributor to this classification problem because ML algorithms tend to classify data according to the majority class. SMOTE is a powerful approach for avoiding bias against the minority class [18]. We applied ML methods with LD clumping comparing performance with and without SMOTE.

Synthetic minority over-sampling technique

Class imbalance occurs in categorical data when one of the classes is less frequent and is a minority class [19]. This is often a problem in population based GWAS where cases of a disease are much less frequent than controls. This situation poses a problem for machine learning models [20]. High dimensional data sets such as gene expression, GWAS, electronic medical records, and text mining are frequently facing this problem [21]. Synthetic Minority Over-sampling Technique (SMOTE) offers a solution to this problem in conjunction to applying ML. SMOTE increases the number of observations within the minority class, using k-nearest neighbours for balancing the data. SMOTE creates synthetic samples by random interpolation among the nearest neighbours in the same class as the minority class sample[22]. When using interpolation operation, the decision space for the minority class is extended to get the closest possibility to the original variables when creating a new sample in the minority class. The minority class with the new samples have the same expected value as the original minority class samples, however, their variances are not the same [23–25]. The new artificial sample X_{new},$X\prime$ randomly selected neighbor, X_i for each instance in minority class.

$$X_{new} = X_i - \left(X' - X_i\right) * \lambda \tag{1}$$

where λ is a random number in the range [0, 1] and i$\in$ [1, n], n is the number of individuals in the minority class.

X_minority: Input data containing samples from the minority class., N: Number of synthetic samples to generate., k: Number of nearest neighbors to consider.

Random Selection of Samples (Line 4): The pseudocode starts by randomly selecting a sample from the minority class data (X_minority). This is a crucial step as it ensures diversity in the generated synthetic samples. If only specific samples were selected, it could introduce bias. Finding Nearest Neighbors (Line 5): By finding the k-nearest neighbors of the randomly selected sample, SMOTE aims to create synthetic samples that are within the local feature space of the existing data. This helps maintain the underlying data distribution and ensures that the generated samples are plausible representations of

the minority class. Random Selection of Neighbor (Line 6): Randomly selecting a neighbor from the nearest neighbors ensures variability in the synthetic samples. This step is important because it introduces randomness and avoids overfitting to specific neighboring samples. Generation of Synthetic Sample (Line 7): The formula used to generate the synthetic sample combines the randomly selected sample and a scaled difference with a randomly selected neighbor. This technique helps in creating new samples that are similar to existing ones but have some variation. Appending Synthetic Sample (Line 8): After generating a synthetic sample, it is added to the list of synthetic samples (X_synthetic). This step ensures that the generated samples are stored for later use, typically to augment the training dataset. Loop Iteration (Line 3): The loop iterates N times, generating N synthetic samples. This parameter allows control over the number of synthetic samples created, which can be adjusted based on the specific needs of the application. Output (Line 9): Finally, the list of synthetic samples (X_synthetic) is returned as the output of the algorithm. These synthetic samples can be combined with the original data to create a balanced dataset for training a machine learning model (Fig. 1).

Support vector machine SMOTE

Support vector machine SMOTE (SVM SMOTE) is a technique that based on combination of extrapolation and interpolation that defining boundaries with SVM separation formulas [26], N2 [27]. Instead of the k-neighbourhood computational interpolation in the SMOTE approach, SVM SMOTE uses support vectors, using interpolation for the majority class, while using extra polarization for the minority class, thus making the sample balanced. Synthetic data will be randomly created along the lines joining each minority class support vector with several its nearest neighbours [4].

SVM Model Training (Line 1): The pseudocode begins by applying an SVM classifier to the original imbalanced dataset. This is a standard step in building a classification model and serves as the baseline for comparison with the resampled dataset. Identification of Minority Class Samples (Line 2): Identifying the minority class samples is crucial for the subsequent steps. This ensures that the resampling technique is specifically applied to the class that requires additional support. Iterating Through Minority Class Samples (Line 3): The loop iterates through each minority class sample (x_i, y_i). This is necessary

```
Input:
  - Minority class samples (X_minority)
  - Number of synthetic samples to generate (N)
  - Number of nearest neighbors to consider (k)
Output:
  - Synthetic samples (X_synthetic)
1:Procedure SMOTE(X_minority, N, k):
2:   X_synthetic = []
3: for i = 1 to N do:
4:   random_sample = randomly select a sample from X_minority
5:  nearest_neighbors = find k-nearest neighbors of random_sample in X_minority
# Randomly select one of the nearest neighbors
6:  neighbor = randomly select a neighbor from nearest_neighbors
# Generate synthetic sample
7: synthetic_sample = random_sample + random () * (neighbor - random_sample)
8: X_synthetic.append(synthetic_sample)
9: return X_synthetic
10: Stop algorithm
```

X_minority: Input data containing samples from the minority class., N: Number of synthetic samples to generate., k: Number of nearest neighbors to consider.

Fig. 1 Pseudo-code of SMOTE

to apply SMOTE on a sample-by-sample basis, ensuring that synthetic samples are generated in the local feature space of each individual sample. Finding Nearest Neighbors (Line 4): For each minority class sample, the algorithm identifies its n_neighbors nearest neighbors within the same class. This step is crucial for generating synthetic samples that are consistent with the distribution of the original data. Applying SMOTE (Line 5): SMOTE is applied to the current minority class sample, taking into account the specified parameters n_neighbors and over_sampling_ratio. This generates synthetic samples that help balance the class distribution. Accumulating Synthetic Samples (Line 6): The synthetic samples produced by SMOTE are added to the list of synthetic samples. This step ensures that the generated samples are retained for further processing. Combining Original and Synthetic Data (Lines 7–8): The synthetic samples are appended to the original feature matrix (X_train), and their corresponding labels are added to y_train. This combines the resampled data with the original training data, creating a balanced dataset for training. Output Resampled Data (Line 9): Finally, the resampled feature matrix (X_resampled) and corresponding labels (y_resampled) are returned as the output of the algorithm. These resampled datasets can be used to train an SVM model on a balanced dataset (Fig. 2).

Adaptive synthetic sampling approach

The Adaptive Synthetic Sampling Approach (ADASYN) is an approach that can be used to solve class imbalance problems based on adaptive minority class generation on data samples. ADASYN increments the minority class by calculating the shuffled and randomly selected majority class distances to form the minority class [28] [N4]. While increasing the number of minority class elements, it not only reduces bias by randomizing the data instead of choosing the distances of those close to the majority class, but also increases the number of minority class elements by considering the majority class elements that are difficult to include in the calculations [29].

Input:
- X_train: Feature matrix of the training set
- y_train: Corresponding labels of the training set
- n_neighbors: Number of nearest neighbors to consider in SMOTE
- svm_kernel: Kernel function for SVM (e.g., linear, RBF)
- svm_C: Penalty parameter for SVM
- over_sampling_ratio: Ratio of over-sampling for the minority class

Output:
- X_resampled: Resampled feature matrix
- y_resampled: Corresponding resampled labels

1: Apply SVM to the original imbalanced dataset to train a classification model.
2: Identify the minority class samples.
3: For each minority class sample (x_i, y_i):
4: Find its n_neighbors nearest neighbors within the same class.
5: Apply SMOTE to generate synthetic samples, considering n_neighbors and over_sampling_ratio.
6: Add the synthetic samples to the list of synthetic samples.
7: Combine the original training data with the synthetic samples:
8: Append the synthetic samples to the original feature matrix (X_train) and their corresponding labels to y_train.
9: Return the resampled feature matrix (X_resampled) and corresponding labels (y_resampled).
10: Stop algorithm

Fig. 2 Pseudo-code of SVM SMOTE

Iterating Through Minority Class Samples (Line 1): This loop ensures that ADASYN is applied to each individual minority class sample. This allows for the creation of synthetic samples tailored to the specific characteristics of each minority class instance. Computing the Number of Synthetic Samples (Line 2–3): The number of synthetic samples to generate for each minority class sample is calculated based on the desired total number of synthetic samples (n_adasyn), taking into account the imbalance ratio between the majority and minority classes. This ensures that the generation of synthetic samples is proportional to the class distribution. Finding Nearest Neighbors (Line 4): Identifying the n_neighbors nearest neighbors within the same class is a crucial step in the ADASYN process. This allows for the selection of neighboring samples that are relevant to the specific minority class instance. Computing Difference Vector and Beta Value (Lines 5–8): The difference vector (d) between the neighbor and the current minority class sample is computed. The beta value, which influences the distribution of different labels around a sample, is calculated based on the difference vector. This step captures the local feature space around the current sample. Generating Synthetic Samples (Lines 10–16): For each synthetic sample to be generated (based on the calculated n_synthetic), a neighbor is randomly selected. The difference vector and beta value are computed again to ensure variability. The synthetic sample is then generated using a weighted combination of the current sample and the selected neighbor. Adding Synthetic Samples to List (Line 17): The generated synthetic samples, along with their corresponding labels, are added to the list of synthetic samples. This step ensures that the generated samples are retained for further processing and integration with the original data. Combining Original and Synthetic Data (Lines 18–19): Appending the synthetic samples to the original feature matrix (X_train) and their corresponding labels to y_train combines the resampled data with the original training data, creating a balanced dataset for training. Output Resampled Data (Line 20): Finally, returning the resampled feature matrix (X_resampled) and corresponding labels (y_resampled) provides the user with the balanced dataset, which can be used for training a machine learning model (Fig. 3).

Random under sampling

Random under sampling (RUS) technique is a method used to provide balance between minority class and majority class in large sample size data with class imbalance. In this method, instead of increasing the number of minority class elements, resampling is done by randomly selecting the majority class elements to equal the number of minority class elements [30]. While applying this method, no mathematical approach is applied, and majority class elements are chosen randomly. Therefore, in this case, data with potentially important information may be deleted [31].

Counting Majority Class Samples (Line 1): Counting the number of samples in the majority class (N_majority) is a crucial first step. This provides a clear understanding of the class distribution in the dataset. Calculating Number of Majority Class Samples to Keep (Line 2–3): By multiplying the ratio with the total number of majority class samples, the algorithm determines how many samples to retain after under-sampling. This allows for control over the level of under-sampling and can be adjusted based on specific needs. Initialization and Storage of Indices (Line 4): Initializing an empty list to store the indices of samples to keep is a necessary preparatory step. This list will be populated

Input:
- X_train: Feature matrix of the training set
- y_train: Corresponding labels of the training set
- n_neighbors: Number of nearest neighbors to consider in SMOTE
- n_adasyn: Desired number of synthetic samples to generate
- beta: Distribution of different labels around a sample

Output:
- X_resampled: Resampled feature matrix
- y_resampled: Corresponding resampled labels

1: For each minority class sample (x_i, y_i):
2: Compute the number of synthetic samples to generate for x_i:
3: n_synthetic = round(n_adasyn * (number of samples in the majority class / number of samples in the minority class))
4: Find its n_neighbors nearest neighbors within the same class.
5: For each neighbor (x_nn, y_nn) in the neighbors:
6: Compute the difference vector d = x_nn - x_i
7: Compute the beta value:
8: beta = exp(-d / var(d))
9: Generate n_synthetic samples:
10: For j in range(n_synthetic):
11: Randomly select a neighbor (x_nn, y_nn)
12: Compute the difference vector d = x_nn - x_i
13: Compute the beta value:
14: beta = exp(-d / var(d))
15: Compute the synthetic sample:
16: x_synthetic = x_i + beta * (x_nn - x_i)
17: Add (x_synthetic, y_i) to the list of synthetic samples.
18: Combine the original training data with the synthetic samples:
19: Append the synthetic samples to the original feature matrix (X_train) and their corresponding labels to y_train.
20: Return the resampled feature matrix (X_resampled) and corresponding labels (y_resampled)
21: Stop algorithm

Fig. 3 Pseudo-code of Adaptive Synthetic Sampling Approach

with the indices selected for retention. Iterating Through Class Labels (Line 5): Iterating through each unique class label in y_train allows the algorithm to distinguish between the minority and majority classes. Handling Minority Class (Lines 6–7): For the minority class, all indices corresponding to the minority class samples are added to the list of indices to keep. This ensures that all minority class samples are retained, preventing any loss of information from this class. Handling Majority Class (Lines 8–9): For the majority class, the algorithm randomly selects N_keep indices from the majority class samples. This introduces variability in the selection process, helping to prevent potential bias. Extracting Feature Matrix and Labels (Line 10): Using the selected indices, the algorithm extracts the corresponding feature matrix and labels. This creates the resampled feature matrix (X_resampled) and corresponding labels (y_resampled) that will be returned as output. Output Resampled Data (Line 11): Finally, the resampled feature matrix and labels are returned, providing the user with the balanced dataset, which can be used for training a machine learning model (Fig. 4).

Machine learning general concepts

Machine learning is a rapidly evolving discipline that involves the examination, learning and development of algorithms to improve computational performance when analysing data. These algorithms help us make the most accurate hypothesis-driven decisions. The two main categories of machine learning are supervised and unsupervised learning [32]. When applying ML methods, data should be divided into training and testing sets. Training is a data subset to train the algorithm, and the test data is for evaluating the performance of the algorithm. Test sets must be independent of the training set to avoid over-fitting. Test sets cannot be used for training the algorithm [33].

```
Input:
- X_train: Feature matrix of the training set
- y_train: Corresponding labels of the training set
- ratio: Ratio of majority class samples to keep (e.g., 1.0 means keep all, 0.5 means keep half)
Output:
- X_resampled: Resampled feature matrix
- y_resampled: Corresponding resampled labels
1: Count the number of samples in the majority class (N_majority).
2: Calculate the number of majority class samples to keep after under-sampling:
3:    N_keep = round(ratio * N_majority)
4: Initialize an empty list to store the indices of the samples to keep.
5: For each unique class label c in y_train:
6:    If c is the minority class:
7:       Add all indices corresponding to the minority class samples to the list of indices to keep.
8:    Else (c is the majority class):
9:       Randomly select N_keep indices from the majority class samples and add them to the list of indices to keep.
10: Extract the corresponding feature matrix and labels using the selected indices.
11: Return the resampled feature matrix (X_resampled) and corresponding labels (y_resampled)
12: Stop algorithm
```

Fig. 4 Pseudo-code of Random Under Sampling

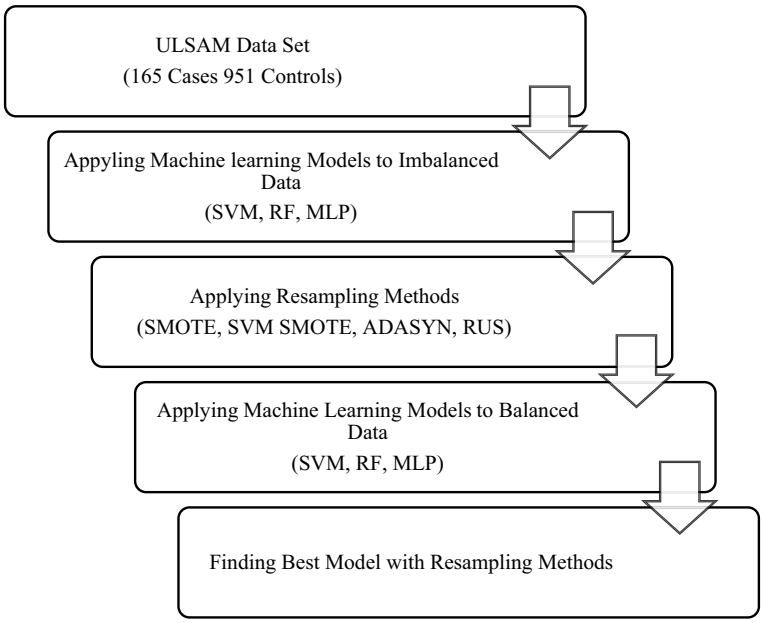


Fig. 5 Implementation Process

Supervised learning is generally concerned with examining classification problems. We can think of classification models in two steps: the first step is to classify the data according to the existing characteristics, constraints, and conditions, and the second step is to test the accuracy of the classes and the validity of the model [32]. If we want to define the classification, simplistically, we can say that it is the process of predicting the classes of data. It is the process of segmenting similar objects, observations, and events according to a specific purpose [33] (Fig. 5).

Random forests

Random forests (RF), which were introduced by Breiman et al. [34] in 2001, are combinations of several random decision trees, with each tree sampled independently and

with the same distribution. The main objective is to achieve higher accuracy for prediction. All trees are constructed with a different bootstrap sample selection from the original data set [35]. Random forests are generated using the Classification and Regression Tree (CART) algorithm, and an information gain criterion is used for splitting each node. Although RF can be used for data with many variables, it requires a lot of available computational memory [36]. The Gini index is used for building the sub-trees within the random forest. The Gini index formulation can be written as the following, where T represents the training set and $\left(f(G_i.T)|T|\right)$ is the probability of the cases selected belonging to the class G_i [37].

$$\sum \sum_{i=1}^{n} \left(f(G_i.T)|T|\right)(f(G_i,T)|T|) \tag{2}$$

Support vector machine

Support vector machines (SVM) were developed by Vapnik et al. [38]. SVM algorithms can be used to find the solution of both clustering (unsupervised) and classification (supervised) problems. Consider drawing a border separating two groups along a plane. SVM determines how this border is drawn [39, 40]. SVM algorithms search for the best solution to the classification problem by utilising optimisation principles which are useful for "big data" evaluation [41, 42]. SVMs can be used for classification, dimension reduction, and SNP selection, by applying one of three different kernels: linear kernel, polynomial kernel, or Radial basis function kernel (RBF)/Gaussian Kernel. RBF is used when solving non-linear problems [43–45]. It is crucial to draw a small margin amongst the two classes to reduce the number of errors produced through classification. If we select two points for the realisation of linear separation and define them as k and l, we maximise the equations $wx_k + b = -1 and wx_l + b = +1$ through subtraction.

$$w(x_k - x_l) = 2 \text{ and } \|\text{w}\| * \alpha = 2 \tag{3}$$

$$\alpha = \frac{w(x_k - x_1)}{\|w\|} \alpha = \frac{2}{\|w\|} \tag{4}$$

$$\alpha \text{ maximization}, \in \{-1, +1\} \tag{5}$$

Conditionally, if y_i=+1 then $w^T x - b \geq 1$ or if y_i=-1 then $w^T x - b \leq$ -1. From this, we can deduce that the minimum is calculated using, $\frac{1}{2}||w||^2$ Where, w is a weight vector, b is bias, x is the input vector, and y_i is the I'th target (i.e. in this case, 1 or -1).

Multi-layer perceptron

Multilayer perceptron (MLP) is a method that can be used for both classification and clustering. It is a method inspired by the human brain [46]. The MLP follows a neural network structure which consists of three different layers; i) input, ii) hidden (one or more), and iii) output. MLP, is a feed-forward method [47], where hidden layers provide a transmission from the input layer to the output layer to classify variables. MLP offers a solution for non-linear classification problems, because it uses the delta learning rule. This rule is based on

sigmoid rules such as a logistic function [48, 49]. The learning function of MLP is represented as:

$$\left(G_1, y_1\right), \left(G_2, y_2\right), \ldots., \left(G_n, y_n\right) where, i \in R^m and y_i \in \{0, 1\} \tag{6}$$

$f(x) = W_2 g\left(W_1^T SNP + b_1\right) + b_2$, W $\in R^m$ and b_1, b_2 are model parameters $\in R$, G represents the Genotypes and W represents the weights. Delta rules are defined by:

$$\nabla W_{j,i} = \alpha\left(target_j - y_j\right) * g\prime\left(h_j\right) * G_i \tag{7}$$

where, α is the learning rate,g(x) is the derivative of the target output. h_j is the weighted sum of the neuron's inputs and y_j is the actual output defined as

$$h_j = \sum G_i * W_{j,i} and y_{j*} g\left(h_j\right) \tag{8}$$

Logistic regression

Logistic regression (LR) is a statistical method used in binary data to determine the relationship between SNPs and disease in GWAS. Since LR is the classical statistical approach, it is used to measure the p-value and odds ratio ratios and the relationship between SNPs and disease [50]. Generally, SNPs associated with the disease are determined with the aid of a Manhattan plot, along the threshold of 10^{-8}. Since LR is a method used as a classification method, in this study, machine learning methods and classification success were compared [51].

Performance measurement metrics

We have assessed the prediction of these ML algorithms using five complementary metrics. Table 1 below defines the confusion matrix terms.

Accuracy is determined by dividing all data by the predicted values. This method, however, is not a sufficient measurement when using imbalanced data.

$$\text{Accuracy} = \frac{TN + TP}{TP + TN + FP + FN} \tag{9}$$

Sensitivity is calculated by dividing the true positive cases by all predicted positive cases.

$$\text{Sensitivity} = \frac{TP}{TP + FN} \tag{10}$$

Specificity is calculated by dividing the true negative cases by all predicted negative cases.

Table 1 Confusion matrix definitions for the predicted and real class

Confusion matrix	Predicted class		
	Status	Control	Case
Real class	*Control*	True negative (TN)	False positive (FP)
	Case	False negative (FN)	True positive (TP)

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{11}$$

The Positive Predictive Value is the percentage of cases that are true positives as predicted by the ML algorithms.

$$\text{Positive Predictive Value} = \frac{TP}{TP + FP} * 100 \tag{12}$$

The Negative Predictive Value is the percentage of controls that are true controls who are predicted by the ML algorithms.

$$\text{Negative Predictive Value} = \frac{TN}{TN + FN} * 100 \tag{13}$$

The F1 score (also known as an F-score or F-measure) is a measure of accuracy. When calculating the F1 score, both precision and recall metrics are used.

$$\text{F1Score} = 2 * \frac{Precision * Recall}{Precision + Recal} \tag{14}$$

The F1 score is used for the evaluation of data that has class imbalance. This metric is widely used for the evaluation of ML models [52, 53].

ULSAM study

The Uppsala Longitudinal Study of Adult Men (ULSAM) is an investigation of healthy elderly men in the Uppsala region of Sweden [54]. It was initiated as a health screen to identify metabolic risk factors for cardiovascular disease. In 1970, all 50-year-old men living in Uppsala were invited to participate, of whom 82% initially agreed to participate, and were subsequently invited back for further study at ages 60, 70 and 77. At each visit, a wide range of phenotypes were collected, including blood pressure, insulin metabolism, weight and height, lipid markers, diet, cognitive function, and socio-economic factors. At age 70, participants were given a glucose tolerance test and insulin clamp to measure insulin resistance. T2D case status was defined by doctor-diagnosed disease or fasting whole blood glucose > 6.1 mmol/l, with all non-cases defined as controls. A total of 1178 participants were genotyped with the Illumina 2.5 M Omni array and Illumina CardioMetaochip [54]. Samples were excluded if the call rate was less than 95%, if they had extreme heterozygosity (> 3 SD from the mean), if they were of non-European ancestry, or if they were female on the basis of X chromosome data. SNP quality control measures included exact p-value for deviation from (Hardy Weinberg Equilibrium) HWE < 10^{-6}, call rate less than 95% (or less than 99% for SNPs with (Minor Allele Frequency) MAF < 5%), and MAF < 1%. Multidimensional scaling (MDS) of a genetic relatedness matrix from LD-pruned autosomal data was performed to obtain principal components to adjust for population structure.

We conducted our analyses on the ULSAM dataset, comprising 165 cases and 951 controls, which exhibited a significant class imbalance. Initially, we assessed the models without addressing this imbalance. Subsequently, we applied techniques to mitigate class imbalance and re-evaluated the models.

The overarching purpose of this study was twofold: firstly, to apply machine learning techniques on real genetic data characterized by class imbalance, and secondly, to develop effective strategies for handling this critical issue. By doing so, we aimed not only to identify the most suitable algorithm for this specific dataset but also to contribute to the advancement of methodologies for analyzing genetic data in the presence of class imbalance.

Results

Machine learning results without clumping

The ULSAM GWAS data were divided into two parts; 70% training set and 30% testing set in preparation for the supervised learning algorithms for classifying cases and controls. tenfold cross-validation was performed to avoid over-fitting [32]. A total of 399,935 SNPs with no missing genotype information for each of the 1116 samples (165 cases and 951 controls) were used in the analysis. After using SMOTE for the adjustment of imbalanced classes, there were 1902 individuals, equally divided between the two classes (951 cases and 951 controls. After using SVM SMOTE for the adjustment of imbalanced classes, there were individuals, 1552 equally divided between the two classes (776 cases and 776 controls). After using ADASYN for the adjustment of imbalanced classes, there were individuals, 1872 equally divided between the two classes (936 cases and 936 controls). After using RUS for the adjustment of imbalanced classes, there were individuals, 466 equally divided between the two classes (233 cases and 233 controls). The choice of tuning parameters affects both the sensitivity and classification performance independently [55]. Therefore, we used the optimal tuning parameters according to the instructions in the Scikit-learn package documentation for Python 3.7 [49, 56].

In the initial phase of our study, we applied machine learning models, namely Support Vector Machines (SVM), Random Forest (RF), and Multi-Layer Perceptron (MLP), to a dataset characterized by class imbalance. The first columns of all respective tables denote the performance metrics achieved using the imbalanced data for each of the aforementioned models. Subsequently, we conducted an investigation involving the application of re-sampling techniques, namely Synthetic Minority Over-sampling Technique (SMOTE), Support Vector Machine Synthetic Minority Over-Sampling Technique (SVM SMOTE), Adaptive Synthetic (ADASYN), and Random Under-Sampling (RUS). The outcomes of these experiments were meticulously recorded in the subsequent columns for each of the models (SVM, RF, MLP). This rigorous methodology empowers us to methodically ascertain the optimal model performance amidst varying re-sampling methodologies.

When comparing the methods used to eliminate the whole class imbalance and the original data, SMOTE method gives the best results for support vector machines with the highest accuracy rate (91%) and F1 score (90%). The SMOTE method is followed by the ADASYN method with an accuracy of 90% and an F1 score of 89%. The RUS method, on the other hand, had the worst results. (Table 2).

It is evident that SMOTE demonstrated the highest efficiency among the imbalanced learning methods applied to enhance the performance of the SVM model, closely followed by ADASYN. While SVM SMOTE also exhibited improvement, its effectiveness was slightly lower than that of SMOTE. In contrast, employing the RUS method and

Table 2 The performances of support vector machine with imbalanced learning methods

Imbalanced learning methods number of SNP 399935		Prediction class		PPV*	NPV**	Sensitivity	Specificity	F1 Score	Accuracy
SVM		Controls	Cases						
Reel class	Controls	278	0	0.00	0.82	0.00	1.00	0.00	0.82
	Cases	57	0						
SMOTE		Controls	Cases						
Reel Class	Controls	287	0	1.00	0.85	0.83	1.00	0.90	0.91
	Cases	47	237						
SVM SMOTE		Controls	Cases						
Reel Class	Controls	293	0	1.00	0.83	0.67	1.00	0.80	0.87
	Cases	56	117						
ADASYN		Controls	Cases						
Reel Class	Controls	285	0	1.00	0.84	0.81	1.00	0.89	0.90
	Cases	51	226						
RUS		Controls	Cases						
Reel Class	Controls	47	0	**	0.47	0.00	1.00	0.00	0.47
	Cases	53	0						

* PPV: Positive Predictive Value

**NPV: Negative Predictive Value

applying the machine learning methods without any correction led to significantly lower classification success, as indicated by the ROC curve (Fig. 6).

When comparing the methods used to eliminate the whole class imbalance and the original data, the SMOTE method and ADASYN are the methods that give the best results for the random forest with the highest accuracy rates (92%) and approximately F1 scores (92%). Therefore, both methods can be used interchangeably. Although the SVM SMOTE method achieved accuracy with a 5% difference compared to the method whose class Imbalance problem was not resolved, it achieved better results in terms of F1 score and sensitivity and specificity than the case without class imbalance. Although the RUS method has the lowest accuracy rate, it has a sensitivity difference of 35% compared to the original data (Table 3).

As can be seen from the Roc curve, it is seen that the classification success of the RF results used without any correction with the RUS method is low, and at the same time, the lines for the SMOTE and ADASYN methods intersect at almost the same point, therefore, the ADASYN and SMOTE methods are almost equal in terms of all metrics (Fig. 7).

Among the methods used to correct class imbalance for MLP, the SMOTE method has the best result with the highest accuracy rate (97%) and F1 score (97%). The SVM SMOTE method, on the other hand, follows the SMOTE method with 92% accuracy and 88% F1 score rate. Although the ADASYN method is not as high in accuracy as SMOTE and SVM SMOTE, it has a very good F1 score (93%). The RUS method, on the other hand, was seen as a bad method (47%) with accuracy and (7%) F1 score (Table 4).

As can be seen from the Roc curve, SMOTE, SVM SMOTE, and ADASYN methods have very high classification success for MLP. The original data and RUS methods showed poor classification performance (Fig. 8.).

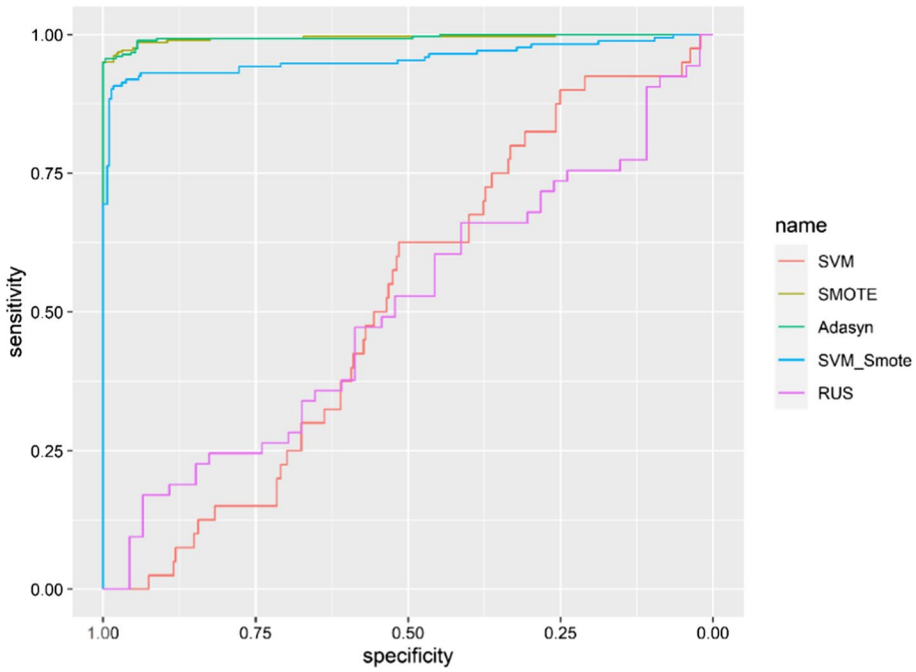


Fig. 6 ROC Curve: Comparison of the performances of Support Vector Machine with Imbalanced Learning Methods

Table 3 The performances of random forest with imbalanced learning methods

Imbalanced learning methods number of SNP 399935		Prediction class		PPV*	NPV**	Sensitivity	Specificity	F1 Score	Accuracy
RF		*Controls*	*Cases*						
Reel class	*Controls*	278	0	0.00	0.82	0.00	1.00	0.00	0.82
	Cases	57	0						
SMOTE		*Controls*	*Cases*						
Reel class	*Controls*	286	1	0.99	0.87	0.85	0.99	0.92	0.92
	Cases	40	244						
SVM SMOTE		*Controls*	*Cases*						
Reel class	*Controls*	290	3	0.97	0.84	0.68	0.98	0.80	0.87
	Cases	54	119						
ADASYN		*Controls*	*Cases*						
Reel class	*Controls*	285	0	1.00	0.86	0.84	1.00	0.91	0.92
	Cases	43	234						
RUS		*Controls*	*Cases*						
Reel class	*Controls*	26	21	0.47	0.43	0.35	0.55	0.40	0.45
	Cases	34	19						

* *PPV* Positive predictive value

** *NPV* Negative predictive value

The logistic regression results applied to the data with class imbalance were found to be quite similar to all machine learning results, with an accuracy rate of 82% and an F1 score of 0.00. The ADASYN method achieved very close results to the SMOTE method,

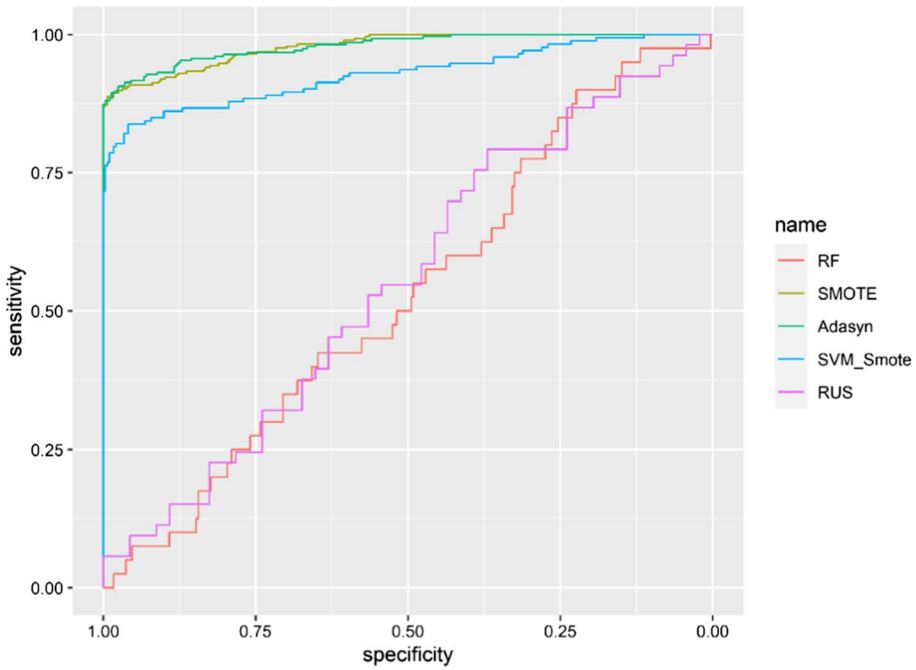


Fig. 7 ROC Curve: Comparison of the performances of Random Forest with Imbalanced Learning Methods

Table 4 The performances of multi-layer perceptron with imbalanced learning methods

Imbalanced learning methods number of SNP 399935		Prediction class		PPV*	NPV**	Sensitivity	Specificity	F1 Score	Accuracy
MLP		Controls	Cases						
Reel class	Controls	277	1	0.00	0.82	0.00	0.99	0.00	0.82
	Cases	57	0						
SMOTE		Controls	Cases						
Reel class	Controls	282	5	0.98	0.96	0.96	0.98	0.97	0.97
	Cases	9	275						
SVM SMOTE		Controls	Cases						
Reel class	Controls	290	3	0.97	0.89	0.80	0.98	0.88	0.92
	Cases	33	140						
ADASYN		Controls	Cases						
Reel class	Controls	285	0	1.00	0.89	0.88	1.00	0.93	0.84
	Cases	32	245						
RUS		Controls	Cases						
Reel class	Controls	45	2	0.50	0.46	0.03	0.95	0.07	0.47
	Cases	51	2						

* *PPV* Positive predictive value

** *NPV* Negative predictive value

with an accuracy of 95% and an F1 score of 93%. The SVM SMOTE method can be considered as an alternative to these two methods with 90% accuracy and 84% F1 score. Although the RUS method, like all other machine learning methods, had poor results, it achieved a 41% higher sensitivity rate than the data with unbalanced classes (Table 5).

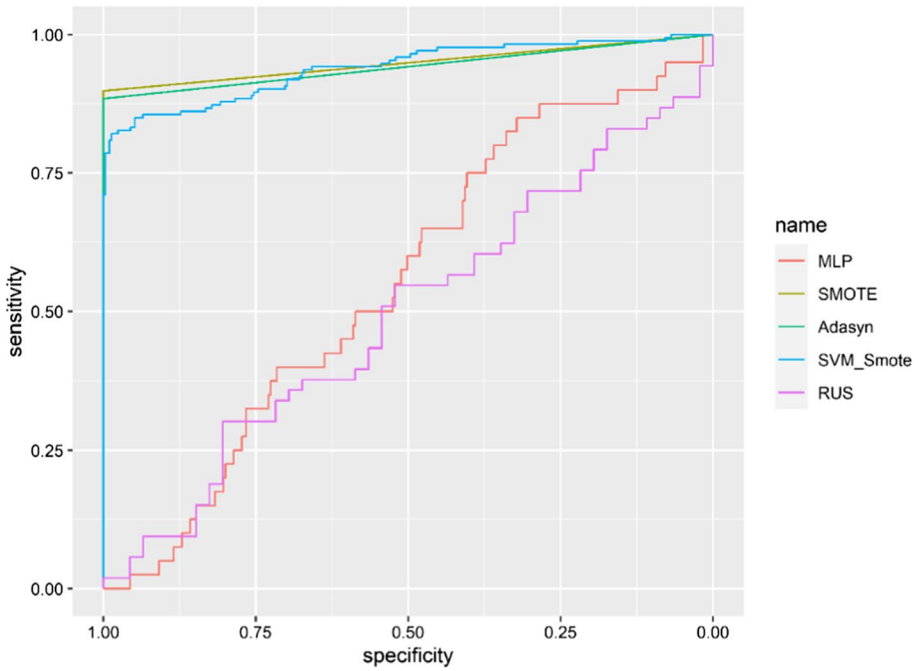


Fig. 8 ROC Curve: Comparison of the performances of Multi-Layer Perceptron with Imbalanced Learning Methods

Table 5 The performances of logistic regression with imbalanced learning methods

Imbalanced learning methods number of SNP 399935		Prediction Class		PPV*	NPV**	Sensitivity	Specificity	F1 Score	Accuracy
LR		Controls	Cases						
Reel class	Controls	276	2	0.00	0.82	0.00	0.99	0.00	0.82
	Cases	57	0						
SMOTE		Controls	Cases						
Reel class	Controls	284	0	0.90	1.00	1.00	0.91	0.95	0.95
	Cases	26	262						
SVM SMOTE		Controls	Cases						
Reel class	Controls	291	2	0.98	0.86	0.74	0.99	0.84	0.90
	Cases	44	129						
ADASYN		Controls	Cases						
Reel class	Controls	285	0	1.00	0.89	0:87	1.00	0.93	0.93
	Cases	35	242						
RUS		Controls	Cases						
Reel class	Controls	28	19	0.53	0.47	0.41	0.59	0.46	0.50
	Cases	31	22						

* *PPV* Positive predictive value

** *NPV* Negative predictive value

For LR, the SMOTE, ADASYN, and SVM SMOTE methods, which are used to eliminate class imbalance, yield almost identical metrics, resulting in very similar results for the entire Roc curve. For RUS and the original data, the ROC curve clearly showed that both methods failed in classification (Fig. 9.).

Machine learning results with clumping

Following the clumping procedure in PLINK [8], the thresholds were chosen as 0.0001, 0.01, 0.50 and 250 for the parameters p1, p2, r2 and kb, respectively. p1: Significance threshold for index SNPs, p2: Secondary significance threshold for clumped SNPs, r2: LD threshold for clumping, kb: Physical distance threshold for clumping. PLINK [8] was used to LD clump SNPs, using an r2 threshold of 0.50 in windows of 250 kb, based on a significance threshold of $p < 0.01$ for index SNPs and $p < 0.0001$ for clumped SNPs. In total, 29 SNPs with very low correlation with one another were obtained.

Data with class imbalances are similar to results before using clumping (e.g., 83% accuracy, 0% F1 score). All the Class Imbalance resolved methods significantly increased the sensitivity measure. The lowest sensitivity rate was found with RUS (72%). All imbalanced learning methods significantly increased the F1 score. SMOTE has the highest F1 score with 82%. (Table 6).

In terms of the Clumping and SVM method, the classification performances of all the methods used to eliminate the class imbalance are very close to each other. Because, when metric values such as sensitive and specificity are examined, it is seen that the results are close to each other (Fig. 10.).

The RF used with the SMOTE method achieved the best result in terms of F1 score (82%). There is no difference between the clumping method and the use of all SNPs

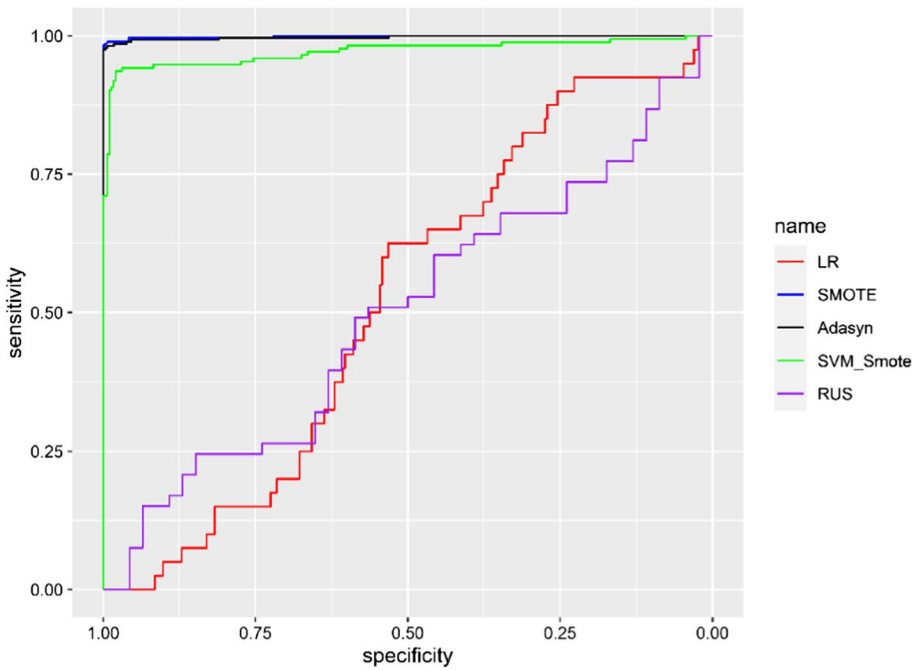


Fig. 9 ROC Curve: Comparison of the Performances of Logistic Regression with Imbalanced Learning Methods

Table 6 The performances of support vector machine with imbalanced learning methods with using clumped SNPs

Imbalanced learning methods number of SNP 29		Prediction class		PPV*	NPV**	Sensitivity	Specificity	F1 score	Accuracy
SVM		Controls	Cases						
Reel class	Controls	278	0	0.00	0.83	0.00	1.00	0.00	0.83
	Cases	57	0						
SMOTE		Controls	Cases						
Reel class	Controls	190	84	0.76	0.85	0.89	0.69	0.82	0.79
	Cases	34	263						
SVM SMOTE		Controls	Cases						
Reel class	Controls	225	53	0.81	0.77	0.77	0.81	0.79	0.79
	Cases	67	226						
ADASYN		Controls	Cases						
Reel class	Controls	201	83	0.75	0.83	0.86	0.71	0.80	0.78
	Cases	42	249						
RUS		Controls	Cases						
Reel class	Controls	44	12	0.72	0.79	0.72	0.79	0.72	0.76
	Cases	12	31						

* *PPV* Positive predictive value

** *NPV* Negative predictive value

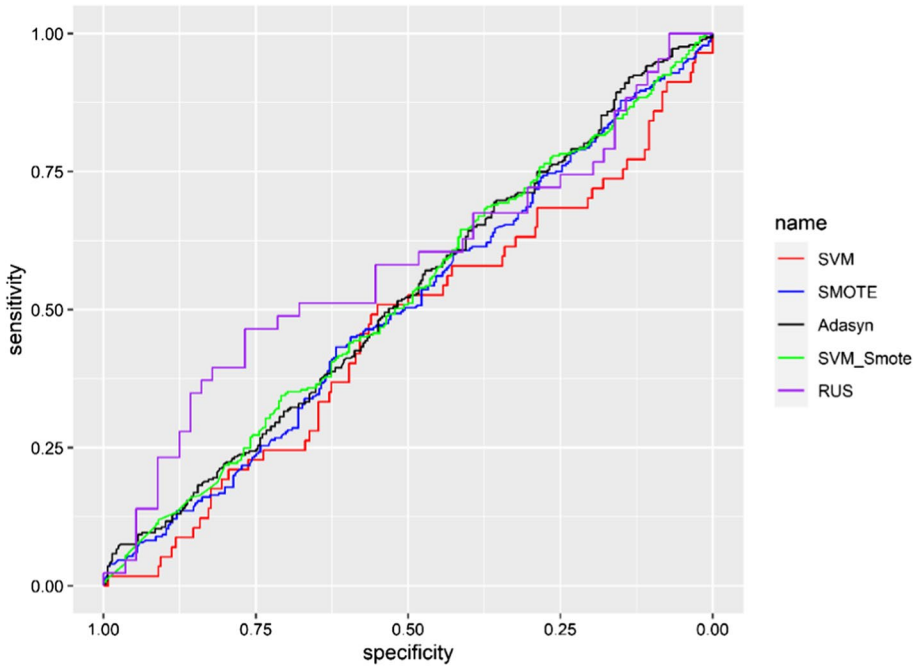


Fig. 10 ROC Curve: Comparison of the performances of Support Vector Machine with Imbalanced Learning Methods with using Clumped SNPs

in data with class imbalance, as in SVM. A good PPV value was obtained with SVM SMOTE (90%). The ADASYN method and the RUS methods obtained very close results with an accuracy rate of 75% (Table 7).

Table 7 The performances of random forest with imbalanced learning methods with using clumped SNPs

Imbalanced learning methods number of SNP 29		Prediction class		PPV*	NPV**	Sensitivity	Specificity	F1 score	Accuracy
RF		Controls	Cases						
Reel class	Controls	278	0	0.00	0.82	0.00	1.00	0.00	0.82
	Cases	57	0						
SMOTE		Controls	Cases						
Reel class	Controls	184	90	0.75	0.84	0.89	0.67	0.81	0.78
	Cases	34	263						
SVM SMOTE									
Reel class	Controls	260	18	0.90	0.65	0.53	0.94	0.67	0.73
	Cases	138	155						
ADASYN		Controls	Cases						
Reel class	Controls	216	68	0.76	0.74	0.74	0.76	0.75	0.75
	Cases	76	215						
RUS		Controls	Cases						
Reel class	Controls	39	17	0.67	0.83	0.81	0.70	0.74	0.75
	Cases	8	35						

* *PPV* Positive predictive value

** *NPV* Negative predictive value

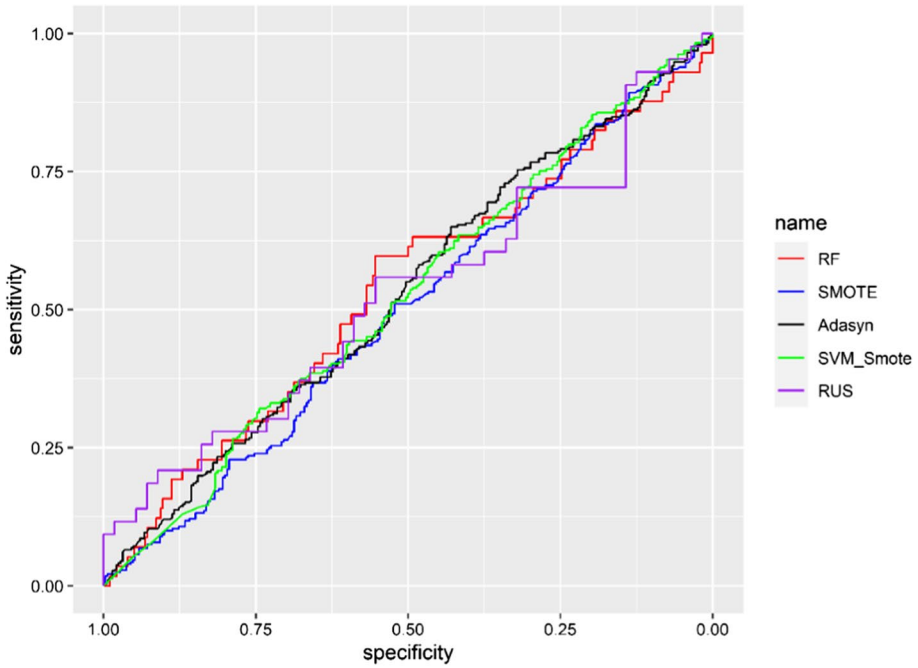


Fig. 11 ROC Curve: Comparison of the performances of Random Forest with Imbalanced Learning Methods with using Clumped SNPs

It is seen in the graph that the results very close to the use of the Clumping SVM method are also obtained with the RF method (Fig. 11).

Compared to SVM and RF, the use of clumping with MLP gave relatively better results with an accuracy rate of 82% and an F1 score of 43%. However, when the class imbalance problem was resolved, F1 scores above 80% were obtained with SMOTE, SVM SMOTE and ADASYN. It should be considered that the RUS method also achieved an F1 score of 63%, although it had a low accuracy rate of 69% compared to the case where the class imbalance was not resolved (Table 8).

For MLP used with the clamping method, there is no difference between the methods used to get rid of the class imbalance in terms of classification performance (Fig. 12.).

While high accuracy (89%) was obtained for data without class imbalance, low sensitivity rate (50%) and low F1 score (60%) were obtained due to class imbalance. In all the methods that eliminated the class imbalance, F1 score, and sensitivity were obtained at a higher rate than the original data, including RUS (Table 9).

All the methods used to eliminate class imbalance gave very close results for LR and LR is no different in terms of classification performance from the method used to eliminate any class imbalance. (Fig. 13.).

Assessment of machine learning results

As will be noted in all tables, accuracy is not the best indicator when evaluating ML models, whereas the F1 score is more informative. All models achieved "good" accuracy results and most methods achieved a positive predictive value and sensitivity value of 0.00. Zero precision algorithms cannot capture true positives. The methods used to resolve the class imbalance not only improved the performance of machine learning methods, but also increased the performance for the classical method,

Table 8 The performances of multi-layer perceptron with imbalanced learning methods with using clumped SNPs

Imbalanced Learning Methods Number of SNP 29		*Prediction Class*		*PPV**	*NPV***	*Sensitivity*	*Specificity*	*F1 Score*	*Accuracy*
MLP		*Controls*	*Cases*						
Reel Class	*Controls*	251	27	0.46	0.88	0.40	0.90	0.43	0.82
	Cases	34	23						
SMOTE		*Controls*	*Cases*						
Reel Class	*Controls*	210	64	0.79	0.80	0.82	0.77	0.81	0.80
	Cases	52	245						
SVM SMOTE		*Controls*	*Cases*						
Reel Class	*Controls*	221	57	0.81	0.81	0.82	0.79	0.81	0.81
	Cases	53	240						
ADASYN		*Controls*	*Cases*						
Reel Class	*Controls*	213	71	0.78	0.86	0.88	0.75	0.83	0.81
	Cases	36	255						
RUS		*Controls*	*Cases*						
Reel Class	*Controls*	42	14	0.65	0.71	0.60	0.75	0.63	0.69
	Cases	17	26						

* *PPV* Positive predictive value

** *NPV* Negative predictive value

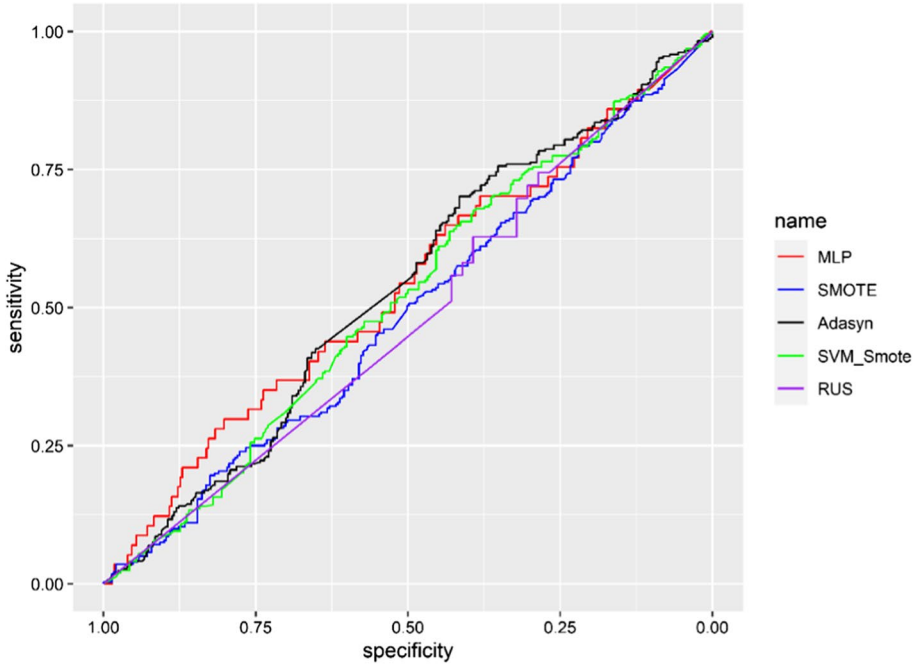


Fig. 12 ROC Curve: Comparison of the performances of Multi-Layer Perceptron with Imbalanced Learning Methods with using Clumped SNPs

Table 9 The performances of logistic regression with imbalanced learning methods with using clumped SNPs

Imbalanced learning methods number of SNP 29		Prediction class		PPV*	NPV**	Sensitivity	Specificity	F1 score	Accuracy
LR		Controls	Cases						
Reel class	Controls	272	6	0.82	0.90	0.47	0.98	0.60	0.89
	Cases	30	27						
SMOTE		Controls	Cases						
Reel class	Controls	210	64	0.79	0.76	0.79	0.77	0.79	0.78
	Cases	67	245						
SVM SMOTE		Controls	Cases						
Reel class	Controls	213	65	0.77	0.74	0.75	0.77	0.76	0.76
	Cases	73	220						
ADASYN		Controls	Cases						
Reel class	Controls	208	76	0.75	0.76	0.77	0.73	0.76	0.75
	Cases	67	224						
RUS		Controls	Cases						
Reel class	Controls	48	8	0.79	0.79	0.70	0.86	0.74	0.79
	Cases	13	30						

* *PPV* Positive predictive value

** *NPV* Negative predictive value

logistic regression. SMOTE applied to all SNPs has been shown to produce better results than clustering with SMOTE (Tables 2, 3, 4 and 5). MLP was the best method overall, but all methods were performed similarly for each analysis comparison.

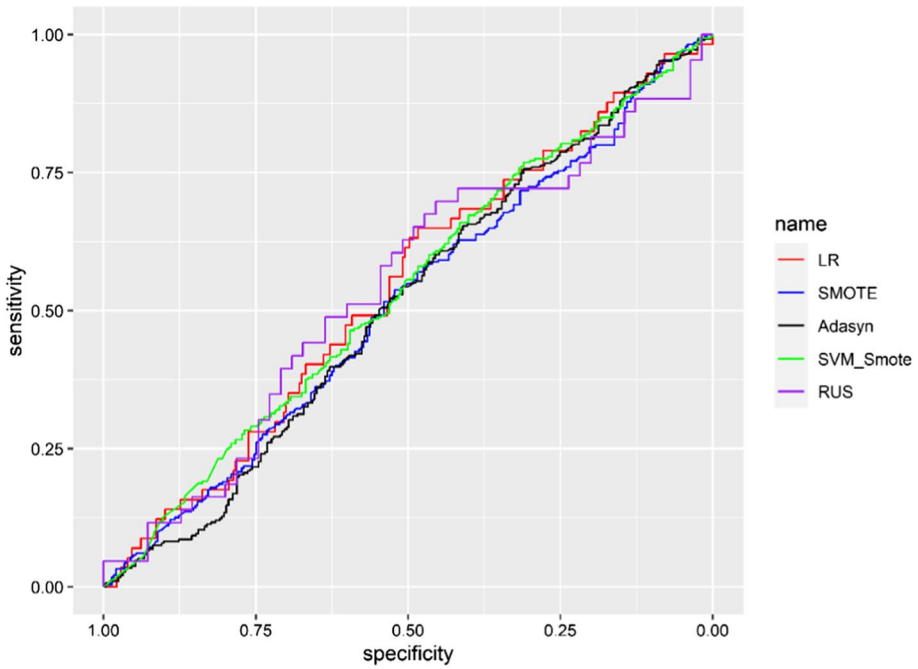


Fig. 13 ROC Curve: Comparison of the performances of Logistic Regression with Imbalanced Learning Methods with using Clumped SNPs

Considering the clumping method, all machine learning methods used to get rid of class imbalance only increased the sensitivity (Tables 6, 7, 8 and 9).

Since the methods we used to eliminate class imbalance in all methods gave close results and the method that gave the best results in general among them was SMOTE, when we compared the BMI correction using the SMOTE method with the BMI correction, it was seen that BMI had no effect on the classification success. As shown in Additional file 1: Tables S1-S4, although BMI is a highly significant predictor of T2D, it has little effect on the predictive metrics evaluated across the methodology. Most interesting is the effect of BMI on the MLP algorithm when clustered SMOTE is applied as there is an increase in prediction accuracy. When we compare the MLP performance in Table 5 and Additional file 1: Table S3, we can see an increase in prediction accuracy of 0.05 and better classification of cases and controls.

Our study constitutes a significant stride in the realm of genetic epidemiology. Previous investigations have not systematically assessed the performance of machine learning algorithms in distinguishing between patients and controls using authentic genetic data characterized by class imbalance, particularly in the context of GWAS data. By redressing this imbalance in genetic data, applying our novel approach to genuine genetic datasets, and refining genetic analytical techniques, our work not only conducts an empirical inquiry but also imparts a substantive contribution to the wider scientific community. This study provides valuable insights for researchers seeking to discern the optimal machine learning and resampling methods for effectively discriminating between patients and controls in genetic data featuring class imbalance.

Covariate investigation

Obesity is a well-established risk factor for T2D. The mean and standard deviation (SD) of BMI in the controls (N = 951) and cases (N = 165) were 25.97 ± 3,22 and 27.93 ± 3,94 respectively. As expected, BMI is strongly associated with T2D (t = − 6.035, p < 0.001). We repeated our classification with adjustment for BMI by including it as a variable in the model. The ML results including BMI are shown in the Additional file. All the models both with and without clumping were not affected by the inclusion of BMI. Nevertheless, the same results were obtained for the models with SMOTE (Additional file 1: Tables S1–S4).

Discussion

The results when using SMOTE (resampling) for imbalanced classification are shown to be the most accurate. The clumping method is better suited in terms of computational speed but is dependent on LD pruning because correlated SNPs are less likely to be eliminated from the analysis. Only a small number of related SNPs were selected using clumping and then pruned by the chosen LD threshold.

Our findings suggest that using the SMOTE method with all the SNPs in a given dataset should be implemented in order to avoid over-fitting. Doing this enabled the use of the whole dataset for SNP pattern recognition compared to the clumping procedure, where many potentially false positive or true negative associated SNPs were not eliminated.

The specificity results in Table 4, highlight a concern with using ML methods. ML methods fall short when dealing with an imbalanced class of data for selecting controls. For this reason, methods such as clumping and SMOTE are important for achieving unbiased results [57–59]. Because, when applying clumping, the SNPs that are lowly correlation with one another are included in the analysis. Applying SVM with clumping achieved very high classification metric results as shown in Table 6. As a final point, we recommend using SMOTE with any of the three ML models we discussed for GWAS data with imbalanced classes. This is because SMOTE seems unaffected by the imbalanced sample size and the volume of SNPs to be analysed. However further analysis is needed with a greater number of patients. Machine learning techniques are being applied to a variety of different data types and are growing in popularity in several industries. One key advantage over classical statistical methodology is that ML models do not require any assumptions. This helps in the search for patterns in large scale datasets produced in the field of genomics.

Different problems can be faced when using ML, such as class imbalance, computation time and memory usage. The imbalanced class problem causes underfitting, and we have shown that using SMOTE could be a solution [60]. Our results show that using SMOTE with RF can drastically improve prediction performance. On the other hand, clumping is beneficial for reducing computation time and memory usage with improved prediction performance over using ML methods without clumping. The clumping method is the best option for large datasets due to its stringent feature elimination criteria. The inflated accuracy results from all models may be related to the dimensionality of the data. Previous studies of a similar nature have shown comparable accuracy rates [61, 62].

In parallel with this methodology, previous studies have also explored the application of RF on datasets characterized by imbalanced class distributions when predicting Diabetes Mellitus risk. Remarkably, these investigations yielded noteworthy results, revealing that the integration of both SMOTE and SVM SMOTE techniques led to a significant enhancement in the classification performance of RF [63]. According to another study employing class-imbalanced data with a machine learning approach, it was observed that the utilization of MLP led to a significant enhancement in classification performance when class imbalance was mitigated through the integration of SMOTE and ADASYN. These findings underscore the efficacy of employing advanced resampling techniques in conjunction with MLP for achieving superior classification outcomes in imbalanced datasets [64]. In a separate study focusing on parameter optimization in the presence of class imbalance, SVM was employed. It was observed that the classification success of SVM experienced a significant boost when combined with the SMOTE [65]. While previous literature has recommended the application of resampling methods to address class imbalance in various datasets, none have specifically utilized GWAS data. In our study, we introduced an innovative approach to the literature by implementing resampling techniques in machine learning on GWAS data exhibiting class imbalance.

Among the methods used to eliminate the other class imbalance, SMOTE and ADASYN generally obtained close results. The RUS method, on the other hand, had poor results when used with all SNPs, because if examined, information loss may occur in randomly selected samples depending on the data size. The reason for the good results of using RUS with Clumping should be taken into account that the loss of information may be less because the data size is reduced (it has decreased to only 29 SNPs).

This paper makes recommendations and suggestions based on data similar to that of the ULSAM genetic and cohort data. Investigators should always be careful and mindful of the impact on their results when selecting and using ML models with imbalanced classes. Previous literature have shown that class imbalance can affect sensitivity or specificity [66]. ML is currently a useful method for the validation of classical GWAS approaches to identify SNPs associated with disease and for the prediction of disease status. As more and larger datasets become widely available, this will enable ML algorithms to predict disease status using SNP data more accurately in future cohorts. Leaving behind the need for stringent p-value thresholds and assumptions.

In this study, machine learning is not used for find out a new SNPs, it is used for diagnostics and precision medicine in individual. This study is a novel for GWAS data with Class-Imbalanced. Compared to other methods, we recommend using SMOTE with MLP.

Conclusion

This study underscores the critical importance of addressing class imbalance in genetic data analysis. Among the methodologies evaluated, the utilization of SMOTE (Synthetic Minority Over-sampling Technique) for imbalanced classification consistently yielded the most accurate results. Additionally, the clumping method demonstrated computational efficiency, although it requires LD (Linkage Disequilibrium) pruning to maintain efficacy. Our findings strongly advocate for the implementation of the SMOTE method using the entirety of SNPs within a dataset to prevent

overfitting. This approach enables comprehensive SNP pattern recognition, distinguishing it from the clumping procedure, which may retain potentially false-positive or true-negative associated SNPs.

It is worth noting that machine learning methods may face challenges, including class imbalance, computation time, and memory usage. Our results showcase that SMOTE, when applied in conjunction with Random Forest (RF), significantly enhances prediction performance, presenting a viable solution for addressing the imbalanced class problem. The clumping method, on the other hand, proves beneficial for large datasets due to its stringent feature elimination criteria. However, the inflated accuracy results from all models should be interpreted cautiously, considering the dimensionality of the data.

This study is not without limitations. It specifically addresses ULSAM genetic and cohort data, and caution should be exercised when extrapolating to different datasets. Our work pioneers the application of machine learning in genetic epidemiology, focusing on diagnostics and precision medicine at an individual level. The study contributes to the evolving landscape of GWAS data analysis, providing a novel approach for handling class imbalance. Moving forward, we recommend the continued use of SMOTE in conjunction with Multi-Layer Perceptron (MLP) models for similar studies.

As more extensive datasets become available, the integration of machine learning algorithms holds promise for enhanced disease status prediction, potentially circumventing the need for stringent p-value thresholds and assumptions. This study sets the stage for future endeavours in genetic epidemiology, poised to make substantial strides in disease understanding and personalized intervention strategies.

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1186/s40537-023-00853-x.

Additional file 1: Table S1. The Performances of Machine Learning Methods with SMOTE. **Table S2.** The Performances of Machine Learning Methods without using SMOTE. **Table S3.** The Performances of Machine Learning Methods without using SMOTE. **Table S4.** The Performances of Machine Learning Methods with using SMOTE. **Table S5.** Significant SNPs using Clumping.

Acknowledgements
None.

Author contributions
Conceptualisation: Dr. OÖ, Dr. HS, Prof. Dr. AM, Prof. Dr. BT. Methodology: Dr. OÖ. Formal Analysis and investigation: Dr. OÖ. Writing—original draft preparation: Dr. OÖ, Dr. HS. Writing—review and editing: Dr. OÖ, Dr. HS, Prof. Dr. AM, Prof. Dr. BT. Supervision: Dr. HS, Prof. Dr. AM, Prof. Dr. BT.

Funding
Not applicable.

Availability of data and materials
The datasets generated and/or analysed during the current study are not publicly available due to data privacy.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Received: 27 March 2023 Accepted: 21 November 2023
Published online: 30 November 2023

References

1. Fadista J, Manning AK, Florez JC, Groop L. The (in) famous GWAS P-value threshold revisited and updated for low-frequency variants. Eur J Hum Genet. 2016;24:1202–5.
2. Szymczak S, Biernacka JM, Cordell HJ, González-Recio O, König IR, Zhang H, Sun YV. Machine learning in genome-wide association studies. Genet Epidemiol. 2009;33:S51–7.
3. Cosgun E, Limdi NA, Duarte CW. High-dimensional pharmacogenetic prediction of a continuous trait using machine learning techniques with application to warfarin dose prediction in African Americans. Bioinformatics. 2011;27:1384–9.
4. Tang Y, Zhang Y-Q, Chawla NV, Krasser S. SVMs modeling for highly imbalanced classification, IEEE transactions on systems, man, and cybernetics. Part B. 2008;39:281–8.
5. Dai X, Fu G, Zhao S, Zeng Y. Statistical learning methods applicable to genome-wide association studies on unbalanced case-control disease data. Genes. 2021;12:736.
6. Zhou W, Nielsen JB, Fritsche LG, Dey R, Gabrielsen ME, Wolford BN, LeFaive J, VandeHaar P, Gagliano SA, Gifford A. Efficiently controlling for case-control imbalance and sample relatedness in large-scale genetic association studies. Nat Genet. 2018;50:1335–41.
7. Bao Z, Zhao X, Li J, Zhang G, Wu H, Ning Y, Li MD, Yang Z. Prediction of repeated-dose intravenous ketamine response in major depressive disorder using the GWAS-based machine learning approach. J Psychiatr Res. 2021;138:284–90.
8. Purcell S, Neale B, Todd-Brown K, Thomas L, Ferreira MA, Bender D, Maller J, Sklar P, De Bakker PI, Daly MJ. PLINK: a tool set for whole-genome association and population-based linkage analyses. Am J Human Genet. 2007;81:559–75.
9. Kinreich S, McCutcheon VV, Aliev F, Meyers JL, Kamarajan C, Pandey AK, Chorlian DB, Zhang J, Kuang W, Pandey G. Predicting alcohol use disorder remission: a longitudinal multimodal multi-featured machine learning approach. Transl Psychiatry. 2021;11:1–10.
10. He KY, Ge D, He MM. Big data analytics for genomic medicine. Int J Mol Sci. 2017;18:412.
11. Pirooznia M, Fayaz Seifuddin JJ, Mahon PB, Potash JB, Zandi PP, B.G.S. Consortium. Data mining approaches for genome-wide association of mood disorders. Psychiatr Genet. 2012;22:55.
12. Fan Y, Tang CY. Tuning parameter selection in high dimensional penalized likelihood. J Royal Stat Soc Series B. 2013;75:531–52.
13. Johnstone IM, Titterington DM. Statistical challenges of high-dimensional data. London: The Royal Society Publishing; 2009. p. 4237–53.
14. Nordhausen K. The elements of statistical learning: data mining, inference, and prediction, by trevor hastie, robert tibshirani, jerome friedman. New York: Wiley Online Library; 2009.
15. Draisma HH, Pool R, Kobl M, Jansen R, Petersen A-K, Vaarhorst AA, Yet I, Haller T, Demirkan A, Esko T. Genome-wide association study identifies novel genetic variants contributing to variation in blood metabolite levels. Nat Commun. 2015;6:1–9.
16. Shi H, Medway C, Brown K, Kalsheker N, Morgan K. Using Fisher's method with PLINK 'LD clumped'output to compare SNP effects across genome-wide association study (GWAS) datasets. Int J Mol Epidemiol Genet. 2011;2:30.
17. Bhowan U, Johnston M, Zhang M, Yao X. Evolving diverse ensembles using genetic programming for classification with unbalanced data. IEEE Trans Evol Comput. 2012;17:368–86.
18. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP. SMOTE: synthetic minority over-sampling technique. J Artif Intell Res. 2002;16:321–57.
19. Japkowicz N, Stephen S. The class imbalance problem: a systematic study. Intelligent Data Anal. 2002;6:429–49.
20. Lusa L. Improved shrunken centroid classifiers for high-dimensional class-imbalanced data. BMC Bioinf. 2013;14:1–13.
21. Turhan S, Özkan Y, Yürekli BS, Suner A, Doğu E. Sınıf Dengesizliği Varlığında Hastalık Tanısı için Kolektif Öğrenme Yöntemlerinin Karşılaştırılması: Diyabet Tanısı Örneği, Turkiye Klinikleri Journal of Biostatistics. 2020; 12.
22. Shrivastava S, Jeyanthi PM, Singh S. Failure prediction of Indian Banks using SMOTE, Lasso regression, bagging and boosting. Cogent Econom Finance. 2020;8:1729569.
23. Seo J-H, Kim Y-H. Machine-learning approach to optimize smote ratio in class imbalance dataset for intrusion detection. Computational İntell Neurosci. 2018;2018:1.
24. Hu F, Li H, A novel boundary oversampling algorithm based on neighborhood rough set model: NRSBoundary-SMOTE, Mathematical Problems in Engineering, 2013.
25. Zheng Z, Cai Y, Li Y. Oversampling method for imbalanced classification. Computing and Informatics. 2015;34:1017–37.
26. Wang Q, Luo Z, Huang J, Feng Y, Liu Z, A novel ensemble method for imbalanced data learning: bagging of extrapolation-SMOTE SVM, Computational intelligence and neuroscience, 2017 (2017).
27. Wang H-Y, Combination approach of SMOTE and biased-SVM for imbalanced datasets, 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), IEEE, 2008, pp. 228–231.
28. He H, Bai Y, Garcia EA, Li S, ADASYN: Adaptive synthetic sampling approach for imbalanced learning,. IEEE international joint conference on neural networks (IEEE world congress on computational intelligence). IEEE. 2008;2008:1322–8.

29. Alhudhaif A. A novel multi-class imbalanced EEG signals classification based on the adaptive synthetic sampling (ADASYN) approach. PeerJ Computer Science. 2021;7: e523.
30. Zuech R, Hancock J, Khoshgoftaar TM. Detecting web attacks using random undersampling and ensemble learners. J Big Data. 2021;8:1–20.
31. Razavi-Far R, Farajzadeh-Zanajni M, Wang B, Saif M, Chakrabarti S. Imputation-based ensemble techniques for class imbalance learning. IEEE Trans Knowl Data Eng. 2019;33:1988–2001.
32. Han J, Pei J, Kamber M, Data mining: concepts and techniques, Elsevier 2011.
33. Alpaydin E, Introduction to machine learning, MIT press2020.
34. Breiman L. Random forests. Mach Learn. 2001;45:5–32.
35. Chen X, Ishwaran H. Random forests for genomic data analysis. Genomics. 2012;99:323–9.
36. Pal M. Random forest classifier for remote sensing classification. Int J Remote Sens. 2005;26:217–22.
37. Strobl C, Zeileis A, Danger: High power!–exploring the statistical properties of a test for random forest variable importance, 2008.
38. Guyon I, Weston J, Barnhill S, Vapnik V. Gene selection for cancer classification using support vector machines. Mach Learn. 2002;46:389–422.
39. Mammone A, Turchi M, Cristianini N. Support vector machines. Wiley Interdiscip Rev Comput Stat. 2009;1:283–9.
40. Furey TS, Cristianini N, Duffy N, Bednarski DW, Schummer M, Haussler D. Support vector machine classification and validation of cancer tissue samples using microarray expression data. Bioinformatics. 2000;16:906–14.
41. I. Nitze, U. Schulthess, H. Asche, Comparison of machine learning algorithms random forest, artificial neural network and support vector machine to maximum likelihood for supervised crop type classification, Proceedings of the 4th GEOBIA, Rio de Janeiro, Brazil, 2012; 79: 3540.
42. Mieth B, Kloft M, Rodríguez JA, Sonnenburg S, Vobruba R, Morcillo-Suárez C, Farré X, Marigorta UM, Fehr E, Dickhaus T. Combining multiple hypothesis testing with machine learning increases the statistical power of genome-wide association studies. Sci Rep. 2016;6:1–14.
43. Ng KLS, Mishra SK. De novo SVM classification of precursor microRNAs from genomic pseudo hairpins using global and intrinsic folding measures. Bioinformatics. 2007;23:1321–30.
44. Statnikov A, Aliferis CF, Tsamardinos I, Hardin D, Levy S. A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis. Bioinformatics. 2005;21:631–43.
45. Deng F, Shen L, Wang H, Zhang L. Classify multicategory outcome in patients with lung adenocarcinoma using clinical, transcriptomic and clinico-transcriptomic data: machine learning versus multinomial models. Am J Cancer Res. 2020;10:4624.
46. Pal SK, Mitra S. Multilayer perceptron, fuzzy sets, and classification. IEEE Trans Neural Netw. 1992. https://doi.org/10.1109/72.159058.
47. Fergus P, Montanez CC, Abdulaimma B, Lisboa P, Chalmers C, Pineles B. Utilizing deep learning and genome wide association studies for epistatic-driven preterm birth classification in African-American women. IEEE/ACM Trans Comput Biol Bioinf. 2018;17:668–78.
48. Ç. Elmas, Y.Z. Uygulamaları, Yapay Sinir Ağları, Bulanık Mantık, Genetik Algoritmalar, 1, Basım, Ankara: Seçkin Yayıncılık, (2007).
49. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V. Scikit-learn: machine learning in python. J Mach Learn Res. 2011;12(2011):2825–30.
50. Staley JR, Jones E, Kaptoge S, Butterworth AS, Sweeting MJ, Wood AM, Howson JM. A comparison of Cox and logistic regression for use in genome-wide association studies of cohort and case-cohort design. Eur J Hum Genet. 2017;25:854–62.
51. Wakefield J. Bayes factors for genome-wide association studies: comparison with P-values. Genet Epidemiol. 2009;33:79–86.
52. Chicco D, Jurman G. The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. BMC Genom. 2020;21:1–13.
53. Korkmaz S. Deep learning-based imbalanced data classification for drug discovery. J Chem Inf Model. 2020;60:4180–90.
54. Lithell H, Sundström J, Ärnlöv J, Björklund K, Hänni A, Hedman A, Zethelius B, Byberg L, Kilander L, Reneland R. Epidemiological and clinical studies on insulin resistance and diabetes. Upsala J Med Sci. 2000;105:135–50.
55. N. Lavesson, P. Davidsson, Quantifying the impact of learning algorithm parameter tuning, AAAI, 2006, pp. 395–400.
56. G. Van Rossum, Python Programming language, USENIX annual technical conference, 2007, pp. 1–36.
57. De Velasco Oriol J, Vallejo EE, Estrada K, Taméz Peña JG, Initiative DN. Benchmarking machine learning models for late-onset alzheimer's disease prediction from genomic data. BMC Bioinf. 2019;20:1–17.
58. Privé F, Vilhjálmsson BJ, Aschard H, Blum MG. Making the most of clumping and thresholding for polygenic scores. Am J Human Genet. 2019;105:1213–21.
59. Schubach M, Re M, Robinson PN, Valentini G. Imbalance-aware machine learning for predicting rare and common disease-associated non-coding variants. Sci Rep. 2017;7:1–12.
60. Li J, Fong S, Mohammed S, Fiaidhi J, Chen Q, Tan Z. Solving the under-fitting problem for decision tree algorithms by incremental swarm optimization in rare-event healthcare classification. J Med Imaging Health Inf. 2016;6:1102–10.
61. Zheng T, Xie W, Xu L, He X, Zhang Y, You M, Yang G, Chen Y. A machine learning-based framework to identify type 2 diabetes through electronic health records. Int J Med Informatics. 2017;97:120–7.
62. Poplin R, Chang P-C, Alexander D, Schwartz S, Colthurst T, Ku A, Newburger D, Dijamco J, Nguyen N, Afshar PT. Creating a universal SNP and small indel variant caller with deep neural networks. Biorxiv. 2018. https://doi.org/10.1038/nbt.4235.
63. Sadeghi S, Khalili D, Ramezankhani A, Mansournia MA, Parsaeian M. Diabetes mellitus risk prediction in the presence of class imbalance using flexible machine learning methods. BMC Med Inform Decis Mak. 2022;22(1):36.

64. Temraz M, Keane MT. Solving the class imbalance problem using a counterfactual method for data augmentation. Mach Learn Appl. 2022;9: 100375.
65. Demir S, Şahin EK. Liquefaction prediction with robust machine learning algorithms (SVM, RF, and XGBoost) supported by genetic algorithm-based feature selection and parameter optimization from the perspective of data processing. Environ Earth Sci. 2022;81(18):459.
66. Afzal Z, Schuemie MJ, van Blijderveen JC, Sen EF, Sturkenboom MC, Kors JA. Improving sensitivity of machine learning methods for automated case identification from free-text electronic medical records. BMC Med Inform Decis Mak. 2013;13:30.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Submit your manuscript to a SpringerOpen journal and benefit from:

► Convenient online submission
► Rigorous peer review
► Open access: articles freely available online
► High visibility within the field
► Retaining the copyright to your article

Submit your next manuscript at ► springeropen.com