4
Classification using machine learning techniques

As discussed and observed in previous chapters, there are many ways to diagnose sarcopenia but, for various reasons, we are following the algorithm developed by EWGSOP. The "Confirm" step of this algorithm requires muscle power which is one of the expensive requirements. The same was discussed in chapter 3 which resulted into a predictive model for muscle power based on BMI, gait velocity and grip strength out of 5 variables, used as inputs. Although, this model was based on already estimated muscle power using Takai and Smith models. The discussion concludes that STS, along with some other physical performance parameters, is one of the most important tests in the diagnosis of Sarcopenia. In this chapter, we use some of the functional screening tests and physical performance parameters to classify sarcopenic subjects from non-sarcopenic subjects. We are using both the data sets, collected in India as well as in the U.K., for the purpose of classification. We use some of the frequently used machine learning algorithm for this classification. One of the problems, which we observed, was that of imbalanced data, that is, divided into majority and minority classes. It is important to note here that balanced nature of data is one of the prerequisites of most of the classification algorithms. We, therefore, briefly discuss the problem of imbalance data classification before we actually classify the given data.

4.1 CLASSIFICATION OF IMBALANCED DATA

It has been very well established that machine Learning techniques are useful in predicting diseases automatically with high frequency rate. It also reduces errors due to human intervention or due to any other operational reasons. Generally, when an expert needs to understand whether a disease is present or not, a binary classification learning algorithm becomes a powerful tool. Decision Tree, Support Vector Machine, Logistic Regression, Artificial Neural Network etc are some of the frequently used techniques for classification. No algorithm is universally preferable to all others but depends on the data set and its use. While analyzing the data, it is often assumed that the distribution of classes is uniform, that is, the data set is balanced with respect to all the categories of the response (output) variable. Clearly, an algorithm gives good results for balanced data set. On the other hand, when the data set is imbalanced, the data distribution features can't be properly displayed by these algorithms. Conventionally, an imbalanced data set is defined as the data with unequal number of examples in each class. There are two categories of imbalance nature of the data –
(i) Between class imbalance – Uneven ratio between two or more outcome classes originating from the difference in class prior probabilities, that is, non-uniform distribution. (ii) Within class imbalance – Rare cases such as, presence of uncommon type of disease. Most of the medical data fall into this category. If the given data is to be classified into presence of disease or not, that is, two class classification, then one of the two would be a majority class. This majority class dominates everywhere in the analysis. Which obviously will result into wrong classification. Almost all the classifiers are sensitive to class imbalance nature of underlying data; however, some of them are more sensitive than others. For example, C5.0 (Decision Tree) is most sensitive as it works globally without paying attention to specific data points. Because of the flexible nature of multilayer perceptron, it is slightly less prone to the class imbalance problem than C5.0. Support vector machine is even less prone since boundaries between classes are calculated with respect to only a few support vectors and the class sizes may not affect the

class boundary too much. Similar is the case with other classifiers also. When the data is imbalanced, the algorithm may put unknown data into the majority class, which leads to wrong classification. In literature, the methods for addressing the class imbalance problem are primarily categorized into Data Level Methods and Algorithmic Level Methods. The data level methods are useful in changing the distribution of training set into uniform (Phua, Alahkoon, Lee 2004). These are resampling methods applied in the data pre-processing stage. This creates a new relatively balanced data set based on the original (imbalanced) one.

The idea behind data pre-processing is, before the training data is presented to the learning algorithm several techniques can be applied so as to make the learning algorithm properly learned from the imbalanced data. They include reduction in majority class sample points, under sampling or magnification of majority class sample points, over sampling. Both methods have obvious drawbacks: under sampling results in some valuable information loss about the characteristic concepts of the majority class, while oversampling increases the risk of over fitting significantly.

A straightforward way is random re-sampling which means random replication of minority class data or random elimination of majority class data. However, exactly replicating sample points multiple times can bias the algorithm to believe that certain examples of the predictor space are directly linked to the outcome of interest. This risk of over fitting is the main issue when performing oversampling. To force algorithms to focus more on the patterns instead of individual configurations, certain techniques of generating synthetic training data are used. We discuss the sampling techniques next.

4.2 SAMPLING TECHNIQUES

We assume that the data needs to be classified into two classes only. The sampling techniques depends on the number of underlying samples. If there are more samples, we go for underlying sampling. In the other case, oversampling will be preferred.

4.2.1 Oversampling techniques

These techniques consider developing synthetic data by increasing the samples of minority class. A very commonly used and efficient algorithm in this category is *Synthetic Minority Oversampling technique (SMOTE)*. This is cluster based algorithm which uses K-nearest neighbors in following manner.

For the minority class A and for each $x \in A$, the k-nearest neighbors of x are obtained by calculating the Euclidean distance between x and every other sample in set A. The sampling rate N is set according to the imbalanced proportion. For each $x \in A$, N examples $(x_1, x_2, \cdots, x_N)$ are randomly selected from its k-nearest neighbors, and they construct the set A_1.

For each example $l \in A_1 (l = 1, 2, \cdots, N)$, the following formula is used to generate a new example:

$$X_{new} = x + \text{rand}(0,1) * d(x, x_l)$$

where $rand(0,1)$ is a random number between 0 and 1. To get rid of the risk due to over-generalization, SMOTE is sometimes combined with certain data cleaning techniques.

4.2.2 Under Sampling techniques

- The objective here is to remove a set of majority class data that creates difficulty for a classifier to learn over imbalanced data set. Some of the common approaches under this category are:

1. *Near Miss Algorithm* - This undersampling method selects examples based on the distance of majority class examples to minority class examples. There are three versions of the technique - i) *NearMiss-1*: select examples from the majority class that have the smallest average distance to the three closest examples from the minority class, ii) *NearMiss-2*: select examples from the majority class that have the smallest average distance to the three farthest examples from the minority class,

and *NearMiss-3* that involve selecting a given number of majority class examples for each example in the minority class that are closest.

The NearMiss-3 seems desirable, given that it will only keep those majority class examples which lie on the decision boundary.

2. *Tomek Link removal* Let x be an instance of positive class A and y be an instance of negative class. Let $d(x,y)$ be the euclidean distance between x and y. (x,y) is a T(Tomek)-Link, if for any instance $z, d(x,y) < d(x,z)$ or $d(x,y) < d(y,z)$.

   If any two examples are T-Link then one of these examples is a noise or otherwise both examples are located on the boundary of the classes. Resampling can occur by either removing all such pairs or, more commonly, only the majority class sample points that form a Tomek link with a minority class point. It is more of a data cleaning method but it is considered as under-sampling.

3. *Condensed Nearest Neighborhood (CNN)* - The objective here is to form a subset of the original (unbalanced) data set such that all original samples are correctly classified by this subset when using Nearest Neighbor classification rule. This subset becomes much smaller than the original data set is advantageous also due to less required computation time and storage capacity.

   The algorithm is as follows -

   Choose any sample point x_1 randomly. Now pick another point x_2 closest to this sample point by using Nearest Neighbor. If the point is classified correctly, set aside. Otherwise, put them together in a set x_1, x_2. Now repeat the process until each point is either set aside or included in the set $U = x_1, x_2, \cdots, x_N$. Repeat the process again with the points set aside or one loop is completed without adding extra points to U. Discard leftover points, use U as the desired subset.

   As this method is practically very useful for data reduction, many useful variations such as GCNN (Generalized Condensed Nearest Neighbor) by Chou, Kuo and Cheng in 2006, FCNN (Fast Condensed Nearest Neighbor) by Angiulli (2007) have been seen in the literature.

   Note that all of the resampling methods only allow resampling the data to a desired ratio, and it is not necessary that they exactly produce the balanced samples.

4.2.3 Machine Learning Classification Approaches

We first discuss, in brief, some of the commonly used classification techniques. These techniques, along with resampling methods, will be used for sarcopenia data classification in the later section.

Instance Based Learning Instance based learning is a class of machine learning models that instead of performing a generalization over the training data, compares the new test data with the training instances stored in the memory, that is, there is no hypothesis learning on such algorithms and the hypothesis are training examples themselves. It is thus a non-parametric approach of classification.

The complexity of model increases as the number of training examples increases. One of the most popular example of Instance based learning is the K-nearest neighbor (KNN) algorithm. In KNN, the label associated with the majority of the K neighbors to any test sample is considered as its final label. The nearest neighbors can be estimated using some distance metrics such as Euclidean distance or cosine distance or Manhattan distance.

Naive Bayes It is a classification technique that is based on Bayes' Theorem that considers strong independence assumptions between the features, that is, a Naive Bayes classifier assumes that the presence of a particular feature in a class is not related to the presence of any other feature. For

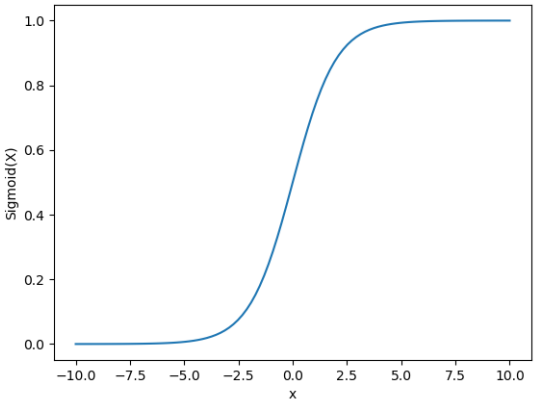

Figure 4.1 : Sigmoid function of logisitic regression

example, a fruit like apple may be red, round and 3 inches in diameter. All these features may depend on the existence of the other but all of these properties are considered to be independently contributing to the probability of a fruit being an apple and that is why it is known as 'Naive'. Bayes theorem provides a way of calculating posterior probability of observing a class c given a feature vector $x = (x_1, \cdots, x_n)$ ($P(c|x)$) using the probability of observing certain features, $P(x)$ and the probability of observing features x given a class c, $P(x|c)$. This is given by:

$$P(c|x) = \frac{P(x|c) * P(c)}{P(x)}$$

where $P(x|c) = P(x_1|c) * P(x_2|c) * P(x_3|c) * ....P(x_n|c)$ is referred to Naive assumption.

Logistic Regression Logistic regression is named after logistic function or the sigmoid function. Sigmoid function is an S-shaped function (Figure 4.1) that maps a real value z to the range $(0, 1)$. Using the formula

$$sigmoid(z) = 1/(1 + e^{-z})$$

Logistic regression uses an equation as the representation like linear regression. The input value x is transformed to an output value z using weights w and bias b using the linear regression equation $z = \theta * x + b$ which is then transformed to the final hypothesis using the sigmoid activation over z, i.e. $h_\theta(x) = sigmoid(z)$. The output value $h_\theta(x)$ represents the probability of the input to belong to the default class. Thus the logistic regression takes its name from a combination of logistic functions applied to the regression equation to finally provide the probability output. If $h_\theta(x) > 0.5$, label 1 is assigned to the sample, and 0 otherwise.

The coefficients θ and b are learned from the training data using maximum-likelihood estimation. The best coefficients would result in a model that predicts a value close to 1 for the default class and 0 otherwise. Maximum likelihood is the search procedure that seeks the value that minimizes the errors in probabilities predicted by the model and the ground truth labels.

Loss function: Loss Function: For a single training instance, let $h_\theta(x)$ be the logistic regression

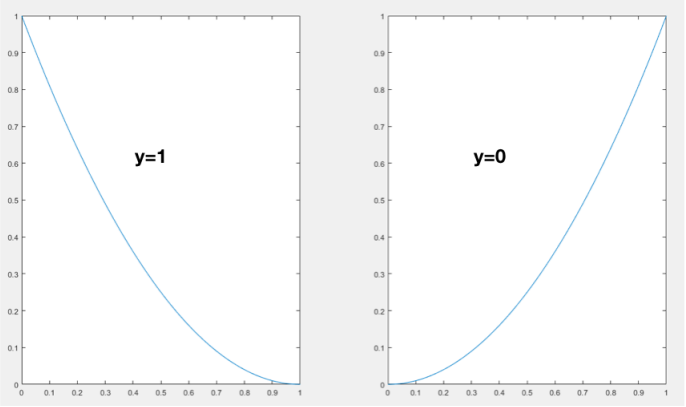

Figure 4.2 : Loss function

output and let y be the groundtruth label of the training instance, then loss is defined as follows.

$$Cost(h_\theta(x), y) = \begin{cases} -log(h_\theta(x)), & if\ y = 1 \\ -log(1 - h_\theta(x)), & if\ y = 0 \end{cases}$$

The loss can be explained using the plots as in figure 4.2

As can be seen from the plots, on the left we have the function $-log(h_\theta(x))$, where $h_\theta(x)$ ranges between zero to one. In case the ground truth label for the training instance is 1 and the predicted hypothesis is $h_\theta(x)$ is also 1, the loss is zero and incase the predicted hypothesis is zero, the loss increases to a very large value. The reverse scenario is seen when y=0 in the plot on right.

We can re-write the loss as

$$Cost(h_\theta(x), y) = -y * log(h_\theta(x)) - (1 - y) * log(1 - h_\theta(x))$$

The overall cost function over all training examples thus becomes,

$$-\frac{1}{m}\sum_{i=1}^{m} y_i * log(h_\theta(x^i)) + (1 - y_i) * (1 - log(h_\theta(x^i))$$

The cost function is then minimized with respect to θ which gives us the following update equations for θ

$$\theta_j := \theta_j - \frac{\alpha}{m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)x_j^i$$

Which is exactly what we have for linear regression. Thus logistic regression is similar to linear regression in terms of weight update rules. However the hypothesis function in the two algorithms is different.

Support Vector Machines Support Vector Machines (SVM) is a widely used preferred machine learning algorithm used for classification problems, which is especially effective in small sample sizes. The SVM method works by finding a hyperplane in an N-dimensional space that classifies the points in this space. In practice, there are large number of hyperplanes that could separate any two classes within

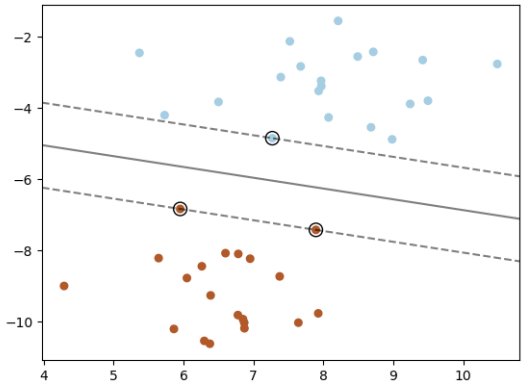

Figure 4.3 : Support Vector Machine optimal hyperplance example

a set of data. Therefore, the aim of SVM is to find the hyperplane that has the biggest margin from all of the points from both classes, which means taking the distance of all of the points from each class from the hyperplane. The greatest value among all the possible hyperplanes is used to classify data into classes, which means that any future data points should also be classified more accurately with this hyperplane than any of the other choices.

In SVM, support vectors are the points that are the closest to the hyperplane, as shown by the circled data points in Fig 4.3. These points have the greatest influence on the hyperplane and can affect both the orientations and the position of the plane. In SVM the output of the linear function is taken, with values greater than 1 classified as a single class, while those less than -1 are grouped into a separate class.

The loss function that maximizes the margin in SVM analysis is hinge loss, which is defined as:

$$c(x, y, f(x)) = \begin{cases} 0, & if\ y * f(x) = 1 \\ 1 - y * f(x), & otherwise \end{cases}$$

Hinge loss will be 0 for cases where both the predicted value and the actual value have the same sign $(-1 or +1)$ or is otherwise non-zero. The weights are then updated using gradients computed using the partial derivative of loss with respect to weights. These updates continue until the loss is minimized, or until there are no more misclassifications. There are a range of different kernels that can be used in SVM analysis such as Linear kernels to determine linear hyperplane and the Radial Basis Function (RBF) kernel that find non-linear hyperplane.

Decision Trees

Decision tree is an easy to implement classification algorithm that can take both numerical and categorical features together. The root nodes constitute the entire training dataset and the tree is designed using step-by-step splits, where at each split the dataset is divided into categories based on a criteria. The simplest split decision is made using the Information Gain criteria where the tree is split on the basis of a condition that maximizes the information gain. Information gain is defined as the decrease in the entropy from the stage i to stage $i+1$, where entropy is the degree of randomness of the data and is defined as

$$H(X) = -\sum_{i=1}^{n} p_i log_2\,(p_i)$$

Other measures include Gini index which considers the goodness of the split by checking how mixed the response classes are in the groups created by the split. The Gini score is defined by :

$G = \sum(p_k^2)$ for $k \in C$

A perfect split is the one in which there is only on class is probable and hence $p_k = 1$ and $G = 0$. The worst split is the one in which both the classes are equally probable and it is difficult to take a decision *i.e.* $p_k = 0.5$ and $G = 0.5$

If decision trees are prone to overfitting for the scenarios when there are too many features and the tree becomes too complex. To overcome the problem of overfitting, we either restrict the length of the tree or restrict the decision splits where every decision made has atleast N number of samples, where N is what we specify.

Random Forests A random forest, as the name implies, is when many decision trees are formed to operate as a forest. In this case, each of the trees in the forest has its own predictive model, with the overall decision chosen by maximum votes. Given that the final decision of the random forest method is based on multiple decision trees, it is more accurate as even if one tree gives a wrong ans it is rectified by the other. To develop multiple trees we may set random subsets of features every time in each tree to make the splits. Random forest is a type of Bagging ensemble technique i.e. it takes a bag of trees and estimates the final decision while considering the majority decision by all the trees and the decision of all the trees are given equal importance.

Adaboost Adaboost is used in conjunction with many other types of learning algorithms to improve or boost their performance. Unlike the bagging ensemble algorithms like random forest that gives an equal weightage to all the models while predicting the label for a test sample, in adaboost the output of the learning algorithms is combined into a weighted sum that represents the final output of the boosted classifier. The weights are assigned to each model based on how accurate a model is. A model that predicts the test instances with a highest accuracy is assigned a highest weight while the one which is least accurate is assigned the lowest weight. In this way the predictions are influenced by stronger models than by the weaker examples.

XG Boost The XGBoost implements the gradient boosting decision tree algorithm. Boosting as we discussed previously is an ensemble technique where new models are added with an objective of correcting the errors made by the existing models. The addition of these models is done until no further improvements can be made. Adaboost that we discussed previously was one such technique. Gradient boosting is an approach where new models are created that predict the residuals or errors of prior models and then added together to make the final prediction. It is called gradient boosting because it uses a gradient descent algorithm to minimize the loss when adding new models.

Next, we define the evaluation metrics over which the performance of various classification algorithms is measured.

4.3 EVALUATION MEASURES

In general, *Accuracy* is the most commonly used measure for classification. However, in case of imbalance data, it is not a proper measure as the accuracy will be dominated by the majority class and the minority class will have less impact on it.

Literature has shown that different accuracy measures are generated for many studies reported,

with each study needing to take into account the misclassification cost with respect to the condition being studied [Maratea, A..et al. 2014]. In the present study, we are interested in sarcopenia, which is a condition thought to be present in around 15-20% of the Indian population. Although the long-term effects of sarcopenia can lead to disability and death, it is a treatable condition, with the treatment including physical activity. Accordingly, it is more important to detect all cases of sarcopenia, with no particular problem if controls are mistakenly classified as sarcopenic. Accordingly, the performance metrics privileged in the first stage of this analysis using the imbalanced data set is recall in the minority data set.

Consider the confusion matrix –

Observed	**Predicted**	
	Positive	**Negative**
Positive	TP	FN
Negative	FP	TN

Following measures can be now derived -

$$\text{Accuracy} : \frac{(TP+TN)}{(TP+TN+FP+FN)}$$

The accuracy is the most important measure but could be misleading for non-uniform distribution of classes.

$$ErrorRate : 1 - Accuracy$$

$$Sensitivity(Recall) : \frac{TP}{(TP+FN)}$$

Recall is a measure of completeness, that is, how often a test correctly generates a positive result for people who have the condition being tested for. More sensitive test will flag almost everyone who have the disease and not generate many false-negative results.

$$Specificity : \frac{TN}{(TN+FP)}$$

Specificity measures the ability to correctly generate a negative result for people who don't have the condition being tested for.

$$Precision : \frac{TP}{(TP+FP)}$$

Precision is a measure of exactness, that is, out of all predicted positives, how many are correctly classified.

$$False\ Positive\ Rate : \frac{FP}{(TN+FP)} = 1 - specificity$$

F-Score (Harmonic Mean of Precision and Recall)

$$F - Score = (1+\beta^2)(PR*RE)/(\beta^2*PR+RE)$$

Where, β is to adjust the relative importance of PR and RE. For balance data, the most common value of β is 1. With $0 < \beta < 1$ we care more about precision and so the higher the threshold the higher the F-score. When $\beta > 1$ our optimal threshold moves toward lower thresholds.

$$G - Mean = \sqrt{SENS*SPEC}$$

The G-mean measures the balanced performance of a learning algorithm. It actually measures the overall accuracy. Mathew's correlation coefficient

$$MCC = \frac{(TP * TN - FP * FN)}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$$

It always lies between -1 and +1 and takes into account all four values in the confusion matrix. A high value means that both classes are predicted well, even if the data is imbalanced. AUC (Area under the curve) – ROC (Receiver Operating Characteristics) curve - The ROC curve is plotted with Recall against the False Positive Rate where Recall is on y-axis and False Positive Rate is on the x-axis. Higher value of AUC indicates that the measure of classification is able to separate the classes nicely.

4.4 SARCOPENIA DETECTION

In order for the Sit to stand test to diagnose sarcopenia, it needs to be able to replicate all the stages of the updated EWGSOP algorithm (Figure 1.1) [Cruz-Jentoft *et al.* [2018]]. This includes an evaluation of muscle strength, muscle quantity or quality, and physical performance. The recent development of the EWGSOP which has included the 5STS as a measure of muscle strength has also seen the change from the SPPB to gait speed for the recommended test of physical performance [Cruz-Jentoft *et al.* [2018]].However, a recent study has shown that chair stand tests can be used as a surrogate test for gait speed[Nishimura *et al.* [2017]]. This would suggest that, rather than using two different tests for muscle strength and physical performance, it might be possible to use the STS for both components of the sarcopenia diagnosis algorithm (Figure 1.1).

Prediction of an older person as meeting or not meeting the physical performance threshold for sarcopenia can be considered to be a classification problem, which in the medical field has become an important application. However, disease detection using classification techniques is problematic as identifying rare disease is by definition an unlikely event, meaning it is hard to find cases of the condition being studied. This means that, in practice, most cases of medical data are imbalanced, meaning that the distribution of the target class is not uniform between cases and controls. If we use the predicted prevalence of sarcopenia in India from the WHO SAGE dataset as an estimate of sarcopenia likely to be found, there are 18% of older people with sarcopenia[Tyrovolas *et al.* [2016]]. This means that a random sample of older people will have a ratio of 4:1 between cases and controls and would this be imbalanced.

A solution to the problem of imbalanced data sets is to use machine learning classifiers, which are designed to maximize accuracy and minimize error. Despite this, imbalanced medical data classification accuracy for the minor class (usually cases) is very low, meaning that the prediction of disease is rare and often wrong.

A standard solution in machine learning is to use random under/ over sampling on the data to solve the problem of imbalanced data. However, for such data sets it is advisable to have deep knowledge of the different features to extract valuable information by using advanced classification techniques.

As previously discussed, in order to detect the prevalence of sarcopenia we need to consider three parameters: muscle strength/power, physical performance and muscle mass. The power estimated using the STS equation had a strong relationship with muscle mass (r= 0.70), see Chapter 3. This was despite only using a standard STS test, without using measurements specifically related to muscle power or STS movement velocity, meaning that this relationship could be stronger were an iSTS to be used. Given the strong link between power and muscle mass, the aim of this discussion is to use muscle power as a surrogate for muscle mass, and accordingly try to predict muscle power. Other studies have also shown that STS is a good measure of physical performance such as gait velocity.

The objective is to evaluate the performance of the STS to predict physical performance against the three physical performance components of the EWGSOP algorithm, which contains tests and corresponding thresholds for the Timed-up-and-go (TUG), gait speed, and the Short Physical Performance Battery (SPPB). A range of different models are tested and compared, with a focus on methods to balance the data sets to include more participants with sarcopenia. This section concludes with a prediction on the prevalence of sarcopenia in the dataset on the basis of threshold values of different physical performances and anthropometric equation on power using the STS. This finding could indicate that the STS is a sufficient measure for the prediction of the prevalence of sarcopenia.

Data set

The dataset, briefly described in Chapter 3, is taken from two studies carried out in Jodhpur in 2015 and 2019 for the Culturally Appropriate Geriatric Screening study, in partnership with the University of Bedfordshire (UK). Participants in this combined dataset were 111 community-dwelling older people, with 26% of the sample having fallen in the previous 6 months and 39% having fallen in the previous two years. Subjects were required to be aged 60 years and over. The data included basic subject characteristics of age, gender, height and weight, from which BMI was calculated. In addition, a hand-timed 5STS was performed along with three of the four performance measures from the EWGSOP algorithm (gait speed, SPPB, and TUG).

Testing protocol

The participants were asked to participate in three tests of physical function, which were administered in a randomized order and interspersed with sections of a questionnaire that included questions about past medical records, health, fear of falling, and falls history. Fall history was recorded using the History of Falls questionnaire [Myers et al, 1996], which has 17 questions related to not only the number of falls within the last 6-24 months, but also information about how, when, and where each fall occurred. The three tests of physical function were all from the EWGSOP2 algorithm, namely the TUG, gait velocity, and the SPPB, which includes the 5STS. It should be noted that the fourth test of physical performance included in the EWGSOP2 algorithm, the 400m walk test, was not included in the study as it was a part of the original EWGSOP algorithm that was available when the study was designed and submitted for ethical approval.

In the TUG test participants are asked to stand up from a chair, walk for 3 meters (9.84 feet) and return to the chair and sit. The gait velocity test requires participants to walk for a distance of 15 feet (4.57 meters) at their normal pace, without overreaching themselves. Each of these tests was performed twice, with the best value used in all subsequent analyses. Finally, participants performed the SPPB, which has three components. Firstly, participants perform four balance tests for 10-seconds using different feet positions, the last of which is a one-leg stance (OLS). Successful completion of each test scores one point, with failure meaning the balance tests are discontinued with the number of successful tests used as the balance score for the SPPB. In practice, this means someone who failed to complete a 10-sec OLS would score three points on the four-point scale. The SPPB also includes a separate gait-velocity test carried out over eight feet (2.44 meters), with the best of two times used in the study. The gait velocity test is scored from 0-4, depending on the time taken to complete the test. The final component of the SPPB is the 5STS, which has been previously described. Participants were briefly instructed to fold their arms across their chest, stand up and sit down as quickly as possible five times, without stopping, keeping their arms in the original position. The fastest of the two attempts was taken as their STS performance. The 5STS is also scored from 0-4, depending on the time taken to complete the test. Each of the three SPPB component scores is added to provide a final score from 0-12, with 12 indicating good physical function and with scores from 0-6 indicating poor physical function.

Table 4.1 : Distribution of dataset

	Gait		SPPB		TUG	
	Majority	Minority	Majority	Minority	Majority	Minority
Train	71	7	69	9	74	4
Test	17	16	17	16	32	1
Total	88	23	86	25	106	5

4.5 ANALYSIS

The EWGSOP-2 algorithm specifies cut-off points of 20 sec for the TUG, eight points on the SPPB, and gait speed of less than or equal to 0.8 m/s. Each participant was classified for the three performance variables using the cut-off points specified by the EWGSOP-2 algorithm, with failing to meet the standard scored as one and meeting the standard scored as zero. These labels become our ground truth for the machine learning approaches that we discuss next.

4.5.1 Experimental Section

The sarcopenia data in the study is clearly an imbalanced data set, with 5 (4.5%), 25 (22.5%), and 23 (21%) in the sarcopenia groups for the TUG, SPPB, and gait velocity categories respectively. The objective of any classification in an imbalanced data set is to have a better recall for the minority class data and high precision on majority class data.

We discuss the sampling techniques in the previous section. In our case the dataset is not large enough for this with only 111 participants, which means that under-sampling would reduce the total sample count and lead to an extremely poor performance of any classification algorithm developed. Thus, we require oversampling techniques in order to increase the number of examples in the dataset. In this work, we use two upsampling techniques: SMOTE and a combination of SMOTE and Tomek link, where Tomek Link does the cleaning of the upsampled data.

Classification Techniques on Original Data

The first step in the analysis was to use the original data set, which was divided into subsets for training and testing. The most commonly used ratio between training and testing is 70:30, so accordingly the data set was randomly sampled into train and test sets. The details of the dataset and the count of minority and majority class in the dataset is given in Table 4.1. After creating the two subsets, we applied the classification techniques, given above, on the dataset. The classifications for the data set were carried out separately for each of the three available measures of physical performance used in the EWGSOP algorithm, TUG, gait velocity and SPPB. The EWGSOP threshold values were used for each measure of physical performance, with input data taken as 5STS time and BMI.

Classification Techniques for Upsampled Data

After carrying out classification on the original data set, the two upsampling techniques were applied to the data, followed by an evaluation of the performance of the different classification models. The classification for the resampled data sets followed the same system as that for the original data, with separate classification for each item of the EWGSOP algorithm and 5STS and BMI as inputs.

4.6 RESULTS

The results of our classification techniques on original and the upsampled techniques for Gait Velocity, SPPB and TUG are presented in Tables **??**, **??** and **??** respectively.

The column headings 1, 2 and 3 in all the tables represent the performance on original data, SMOTE upsampled data and SMOTE+Tomelink resampled technique respectively.

Our overall analysis of all the classification techniques state that SPPB is the best measure to identify patients with Sarcopenia. This is better than what we achieve in case of Gait velocity. While for TUG the performance is reasonably low. This may adhere to the fact that the number of samples of the minority class was too less in case of TUG and hence reflects in its poor performance.

Further, the upsampling technique SMOTE followed by TOMEK Link gives the best results in all the cases. In all the three cases, Logistic Regression gives an overall best performance over the other classification techniques thus giving us an intuition that the Sarcopenia and Non-Sarcopenia data points are linearly separable.

Table 4.2 : PERFORMANCE (F-MEASURE, G- MEAN, AUC AND ACCURACY) FOR DIFFERENT CLASSFIERS FOR GAIT VELOCITY . EVALUATION METRICS OBTAINED BY THREE APPROACHES 1. WITHOUT PRE-PROCESSING, 2. WITH SMOTE, 3. WITH SMOTE COMBINED WITH TOMEK LINK

Classifier	Accuracy			G Mean			ROC-AUC			F-Measure		
	1	2	3	1	2	3	1	2	3	1	2	3
Logistic	0.8235	0.6176	0.6176	0	0	0	0.4414	0.3724	0.3655	0	0	0
Random Forest	0.8235	0.7647	0.6471	0	0.5754	0.3806	0.431	0.5966	0.5828	0	0.3333	0.1429
Ada boost	0.7353	0.7941	0.7647	0.4068	0.5872	0.5754	0.4931	0.7655	0.6724	0.1818	0.3636	0.3333
SVM	0.8529	0.5	0.4706	0	0.3322	0.3216	0.5	0.3759	0.3586	0	0.1053	0.1
Decision Tree	0.6176	0.7647	0.7647	0.3714	0.5754	0.5754	0.4448	0.6138	0.6138	0.1333	0.3333	0.3333
Naïve Bayes	0.8529	0.6765	0.6765	0.4394	0.3895	0.3895	0.4759	0.5172	0.4	0.2857	0.1538	0.1538
KNN	0.8235	0.6176	0.5882	0	0.3714	0.362	0.3379	0.4379	0.5069	0	0.1333	0.125

Table 4.3 : PERFORMANCE (F-MEASURE, G- MEAN, AUC AND ACCURACY) FOR DIFFERENT CLASSFIERS FOR SPPB. EVALUATION METRICS OBTAINED BY THREE APPROACHES 1. WITHOUT PREPROCESSING, 2. WITH SMOTE, 3. WITH SMOTE COMBINED WITH TOMEK LINK

Classifier	Accuracy			G-Mean			ROC-AUC			F-Measure		
	1	2	3	1	2	3	1	2	3	1	2	3
Logistic	0.8529	0.9118	0.9118	0.6124	0.8492	0.8492	0.8894	0.8846	0.8942	0.5455	0.8	0.8
Random Forest	0.8529	0.8529	0.8824	0.6934	0.6934	0.7752	0.7933	0.7933	0.7885	0.6154	0.6154	0.7143
Ada boost	0.8235	0.7647	0.8529	0.6005	0.576	0.6934	0.8245	0.8053	0.8413	0.5	0.4286	0.6154
SVM	0.8824	0.9118	0.9118	0.7752	0.8492	0.8492	0.7933	0.8558	0.8558	0.7143	0.8	0.8
Decision Tree	0.8529	0.8235	0.8529	0.6934	0.6005	0.6934	0.7308	0.6683	0.7308	0.6154	0.5	0.6154
Naïve Bayes	0.8529	0.9118	0.9118	0.6124	0.8492	0.8492	0.8798	0.875	0.851	0.5455	0.8	0.8
KNN	0.8824	0.8824	0.8824	0.7752	0.8321	0.8321	0.8486	0.851	0.8413	0.7143	0.75	0.75

4.7 DISCUSSION

The aim was to evaluate the performance of the STS to predict physical performance against the TUG, gait velocity, and SPPB physical performance components of the EWGSOP algorithm. The analysis used both the original data set as well as two different resampling methods and 8 different classification methods. The classification obtained were better for the SPPB test than for TUG and gait velocity. None of the classification methods were successful for the original data set, which was highly imbalanced.

Table 4.4 : PERFORMANCE (F-MEASURE, G- MEAN, AUC AND ACCURACY) FOR DIFFERENT CLASSFIERS FOR TUG. EVALUATION METRICS OBTAINED BY THREE APPROACHES 1. WITHOUT PREPROCESSING, 2. WITH SMOTE, 3. WITH SMOTE COMBINED WITH TOMEK LINK

classifier	ACCURACY			G Mean			ROC-AUC			F-Measure		
	1	2	3	1	2	3	1	2	3	1	2	3
Logistic	0.9412	0.9412	0.9412	0	0.696	0.696	0.9688	0.9844	0.9844	0	0.5	0.5
Random Forest	0.9412	0.9118	0.9118	0	0	0	0.5703	0.5313	0.5313	0	0	0
Ada boost	0.9412	0.9118	0.9118	0	0	0	0.4844	0.7031	0.6094	0	0	0
SVM	0.9412	0.9412	0.9412	0	0.696	0.696	0.5	0.7344	0.7344	0	0.5	0.5
Decision Tree	0.9412	0.9118	0.9118	0	0	0	0.5	0.4844	0.4844	0	0	0
Naïve Bayes	0.9412	0.9118	0.9118	0	0	0	0.5781	0.4844	0.4844	0	0	0
KNN	0.9412	0.9118	0.9118	0	0	0	0.5	0.4844	0.4844	0	0	0

However, for all resampling methods, the performance improved with the best performance observed when a combination or SMOTE and TomekLink were applied.

This classification study was given only two parameters for classification, namely 5STS time and BMI. It is expected that better results would be obtained with addition of other descriptive parameters such as age, sex, or even simple body composition measures such as calf circumference. In addition, the use of iSTS parameters such as velocity, power would also be likely to improve the performance of the classification methods used. This was shown by the results of Chapter 3 (iSTS Systematic Review), in which better results for discrimination between fallers and non-fallers and or frailty classification were obtained when using iSTS parameters such as STS velocity.

The finding of this study suggests that the STS could be a sufficient measure for the prediction of the severity of sarcopenia using the EWGSOP algorithm. Future work will focus on the effect using iSTS parameters and simple measures of body composition to enhance classification performance.

...