## Background Information

### Maternal Mortality

In 2015, the United Nations committed to the achievement of 17 Sustainable Development Goals (SDGs) by 2030 [1]. These SDGs targeted health, the environment, inequality, peace and justice with the aim of eliminating global poverty and protecting the planet [1]. Progress towards the Sustainable Development Goals is monitored by a panel of independent scientists [1]. In 2023, this panel produced a report warning the international community that many of these goals would fail to be met given the current level of progress and a combination of crises that further stalled and/or wiped away years of progress, such as the COVID-19 pandemic and environmental disasters [1].

The report emphasised the need to increase the pace at which maternal and child mortality is being targeted and eliminated [1]. In response to these concerns, in 2024 an additional resolution was passed at the Seventy-seventh World Health Assembly in 2024 to increase progress and further reduce maternal and child mortality [3]. SDG 3.1 was one of the main goals targeted by this resolution, where goal 3.1 aims to reduce the global maternal mortality ratio (MMR) to below 70 per 100 000 live births by 2030, with no single country having an MMR of greater than 140 [21]. The goal also The World Health Organisation defines that MMR as [3]:

“the number of maternal deaths in a given time period per 100 000 live births during the same time period”

Where the latest version of the International statistical classification of diseases and related health problems (ICD) defines a maternal death as [2]:

“the death of a woman while pregnant or within 42 days of termination of pregnancy, irrespective of the duration and site of the pregnancy, from any cause related to or aggravated by the pregnancy or its management, but not from accidental or incidental causes.”

These concerns were driven from the analysis that, despite falling by 40.0% between 2000 and 2023, as of 2023, the global MMR was 197 deaths per 100,000 live births (uncertainty interval 174 to 234), notably higher than the SDG goal of 70 deaths per 100,000 live births [3]. However, these global statistics hide country-level inequity, as approximately 95% of maternal deaths occur in low and lower middle-income countries and fragile settings [4]. For example, while Nigeria had an MMR of 993 in 2023 (uncertainty interval 718 to 1540), Australia had an MMR of 2 (uncertainty interval 2 to 4) in the same year [3]. As a result, only a small subset of countries is projected to meet this SDG [4].

The leading cause of maternal deaths between 2009 and 2020 was haemorrhage (27%), which is a type of direct obstetric death [4]. Studies have found that haemorrhage disproportionately causes maternal death among women in lower-middle income countries [4]. Many of these maternal deaths are considered preventable, as effective treatments have been developed to prevent haemorrhage induced maternal death [4]. Indirect obstetric deaths (23%) and hypertensive disorders (16%) were the second and third most common causes of maternal death between 2009 and 2020, followed by abortion and pregnancy-related infection (7%) [4]. Experts have predicted that, over time, national maternal mortality rates will decrease and the majority of maternal deaths will be caused by indirect, non-communicable conditions rather than by direct obstetric causes [23]. This shift has been called the ‘obstetric transition’ phenomenon’, and a country’s position within this transition has important implications for the design of strategies needed to reduce its maternal mortality rate [23].

### Monitoring Maternal Mortality

Reports published by the World Health Organisation (WHO) as well as academic researchers highlight how lack of access to accurate, complete data about maternal mortality hinders effective interventions [3, 4, 21]. For example, in 2015 the WHO highlighted the necessity of improving measurement of maternal mortality in its Strategies toward Ending Maternal Mortality report [4, 21]. Maternal mortality data helps policymakers identify high-risk regions as well as possible causes of maternal mortality, thus allowing them to implement timely, targeted, and useful programs to reduce maternal mortality [3, 21]. The maternal mortality ratio (MMR) is estimated from one or more of a diverse range of data sources, with a large sample size and/or complete records needed for stable MMR estimates given the relative rarity of maternal deaths [22].

Typically, MMR estimates are informed by national civil registration and vital statistics (CRVS) systems, which are country-specific data collection systems that continuously record births and medically certified deaths within a country [3, 22]. The data recorded in these systems is mandated by country-specific legislation [10]. Deaths are recorded with their associated cause of death in line with the International statistical classification of diseases and related health problems (ICD) [3, 22]. Thus, CRVS systems generate vital information for mortality monitoring and policy development, as in a perfect world they would record all deaths occurring within a country with their causes [10, 11]. However, in 2017, less than 40% of countries had civil registration systems that enabled continuous and accurate maternal mortality monitoring [10]. Unfortunately, this prevents monitoring of trends in maternal mortality, especially in the lowest income countries that have the highest MMR burdens, as they tend to have the most missing data [3, 4, 22]. For example, in 2017, of the 49 least developed countries, only 2 have greater than 50% death registration coverage [10].

Even when CRVS systems are in place, they are limited by their national coverage and can be subject to a myriad of underreporting and misclassification errors, reducing their data quality [10, 21, 22, 4]. More specifically, underreporting refers to when a maternal death is not registered in the correct system, while misclassification occurs when the incorrect cause of death is recorded in the CRVS database [11]. While maternal mortality is underreported at all stages of pregnancy, this is particularly true at the earliest stages of pregnancy when signs of pregnancy may be missed [21, 22]. Maternal deaths that occur at home are also more likely to be underreported [21]. Additionally, maternal deaths due to abortion or an extramarital pregnancy may be underreported due to social stigma or legal barriers [21]. Maternal mortality is often also misclassified because of the complexity of isolating the underlying cause, especially in relation to indirect maternal deaths caused by underlying health conditions [21, 22].

As a result of underreporting and misclassification errors, studies have found that a lower bound for underestimation of maternal mortality is by approximately 40%, with large differences between countries [21]. More specifically, studies have found that maternal deaths are underestimated by approximately 36% in high and upper-middle income countries versus roughly 45% in low and lower-middle income countries [21]. As a more explicit example, a study run in Guatemala found that adding data from medical charts and public healthcare centres to the country’s CRVS data increased the number of maternal deaths by a factor of three compared to the CRVS data alone [10]. Due to underreporting and misclassification errors, reliability of CRVS data must be confirmed before use [3].

MMR estimates are also informed by specialised studies on maternal mortality, whose goal is to determine the true MMR within a specific geographic region. These studies use data from information sources like police and medical records, national registries, administrative reviews, medical autopsies, and censuses [3]. They are often considered the gold-standard [3].

In addition to CRVS systems and specialised studies, the MMR can also be informed by broader surveys, censuses, and national surveillance data [3]. These sources can be particularly useful in low and middle-income countries that lack CRVS systems [21]. Unfortunately, the surveys may not provide adequate coverage, especially of rural areas that are more difficult and/or expensive to reach and may not be conducted in a timely fashion [21, 3]. Countries can use both national surveys and household surveys, where the latter is more popular in regions that lack routine data collection services [3]. However, the relative rarity of maternal mortality occurrence means these surveys require a large sample size to be statistically significant, which can be prohibitively expensive to conduct [3]. As an alternative, maternal deaths can be monitored using Multiple Indicator Cluster Surveys (MICS), which is the WHO recommended method for countries without other reliable sources of data [3]. The sisterhood method queries adult respondents about how many of their sisters have died from a pregnancy-related cause [3]. While asking respondents about their sisters’ health increases sample size, the survey does not provide current data [3]. In addition to surveys, many developing countries rely on health facility data, which consists of routinely collected data from health facilities and providers [10].

### Machine Learning

Emerging technologies enable the collection and transport of large quantities of data, such as the records described in the previous section [6]. By analysing a wide variety of data sources, from social media to Internet of Things (IoT) devices to health monitors and records, scientists and researchers have the opportunity to identify complex, insightful patterns in “Big Data” [6, 12]. These patterns are increasingly being detected and analysed using machine learning, which refers to the process of creating a model through learning patterns in data as opposed to encoding of hand-crafted rules [6, 5]. Machine learning is particularly useful when applied to datasets with many datapoints and/or variables, as the technique can find hidden patterns that may be missed by humans [5]. Machine learning models can then use these patterns to make predictions in the absence of empirical data and to gain a better understanding of underlying systems [5]. Thus, it is a useful technology to employ in the landscape of missing epidemiological data [24].

Conventionally, the input dataset to a machine learning model consists of a number of observations, where each observation is referred to as a ‘datapoint’ [5]. Each datapoint consists of a certain number of variables, which are referred to as ‘features’ in machine learning. Features with discrete values are called ‘categorical’ and features with continuous numerical values are called ‘continuous’ [5]. For example, if a feature describes ‘risk’ and its values were ‘high’, ‘medium’, or ‘low’, it would be considered categorical. In contrast, if its values were a risk score between 0 and 5, it would be continuous. Generally, each row of the input dataset corresponds to a datapoint, and each feature corresponds to a column.

Machine learning models can be broadly classified as supervised or unsupervised methods [5].

#### Unsupervised Learning

Unsupervised learning models can are used to identify patterns in unlabelled data [5]. Unlabelled data refers to an input dataset where the datapoints are not associated with a specific category or value. The aim of unsupervised learning is to uncover hidden patterns and learn the data’s structure [5]. By not providing labels, the model is not explicitly guided toward learning a specific type of pattern in the data. A common application of unsupervised learning methods is dimensionality reduction, which transforms a dataset with many variables into a dataset with fewer variables while retaining as much of the data’s original variation as possible [5]. The transformed dataset may contain linear and non-linear transformations of the original variables. One widely used dimensionality technique is called principal component analysis and is often applied to be able to visualise a dataset with many variables using only two variables [5].

#### Supervised Learning

In contrast, supervised learning occurs when a model is fit to a labelled dataset, where each input datapoint is associated with one or more output categories or values [5]. The true values of the output variables are referred to as the ‘ground truth’, and the model is trained to recognise patterns to help it predict the ground truth [5]. Supervised learning can be applied to classification problems, where the ground truth is one or more specific categories, or regression problems, where the ground truth is a continuous numerical output [5]. This thesis will focus on supervised machine learning methods for regression analysis, as models can be trained on data labelled with ground truth maternal mortality ratio, which are continuous values.

The dataset used to produce a machine learning model is often described as having input and output data [5]. More specifically, a dataset containing *n* samples is denoted by , where refers to a *d*-dimensional input feature vector and refers to the corresponding continuous, numeric output value [7]. When solving a regression problem, the model’s goal is to learn a mapping, *f(x)*, between the input data and the associated ground truth so that, for a new input datapoint, *x\**, the model can correctly predict the associated ground truth *y\** [6]. The type of mapping used defines the machine learning model being implemented [6]. At its core, this mapping is a mathematical function defined by a series of parameters, where the function takes in the input dataset and generates its prediction of the output [5]. To produce accurate predictions, the mapping must approximate the true, underlying relationships between the variables in the data [5].

##### Model Development

Model performance depends on whether the model’s parameters are well-suited to the model’s purpose and dataset [5]. The model’s performance is defined by a loss function, which quantifies the difference between the model’s output and the ground truth label [5]. The process of optimising the model’s parameters involves minimising this loss function, which commonly involves a technique called gradient descent [7, 25]. Intuitively, gradient descent takes advantage of the observation that the gradient quantifies the direction of greatest increase. Thus, taking the negative gradient of the loss function with respect to each parameter gives the direction that the parameter’s value would need to move to produce the greatest decrease in loss [25]. As a result, to minimise the loss function ,the gradient of the loss with respect to a specific parameter can be subtracted from the parameter’s current value, as shown in the formula below [26]. The symbol is the learning rate, which determines the degree to which the negative gradient is used to adjust the parameters’ value [26]. This gradient descent algorithm is applied to all model parameters with the aim of minimising the model’s loss function through optimising its parameter values. There are many different implementations of gradient descent, such as sample gradient descent, which calculates the gradient using only a subset of data rather than the entire dataset to reduce computational complexity [25].

Model development must be done with care, as optimising the model’s parameters with respect to the specific input dataset can result in the model accurately predicting the output variable for any datapoint in the input dataset but failing to predict accurately for out-of-sample data [5]. This may occur because the input data contained noise, which the model learned as a true pattern, preventing the model from learning the true, underlying patterns in the data that would allow high predictive performance on out-of-sample data [5]. However, the model must still be fit to the data to prevent underfitting, which occurs when the model cannot accurately capture the underlying relationships in the data [5]. Overfitting and underfitting is related to the bias-variance trade-off, where bias refers to errors in the model’s predictions while variance refers to change in the model’s predictions based on the training data used [5]. The goal of model development is to produce a model with low bias and low variance. However, to reduce bias, the model generally must become more complex, which can cause overfitting and increase variance, necessitating a trade-off between bias and variance [5].

To balance the goal of low bias while avoiding overfitting, the dataset is generally split into non-overlapping training and testing subsets, generally in a ratio between 72:25 and 90:10 [5]. The model’s parameters are fit to the training dataset through minimising the loss function [5]. Then, the model’s performance is evaluated on the previously unseen test dataset to determine whether the model is generalisable or is overfit to the training data [5].

However, the model should not be overly adjusted based on the results of the testing dataset to prevent overfitting to the testing data, thus preventing any accurate measurements of out-of-sample performance from being made [5]. To address this problem, the training data can be further split into non-overlapping training, validation subsets [5]. In this scenario, the model’s parameters are fit using the training data, and the model’s out-of-sample performance is tested on the previously unseen validation set [5]. The performance of different versions of the model can be tested on the validation set before the model’s ability to generalise is evaluated on the unseen test set [5]. The validation set is particularly useful for evaluating the performance of different hyperparameter specifications, where hyperparameters govern the architecture of a model and the training process, but are not themselves fine-tuned during training [5]. For example, the learning rate, or the rate at which parameter values are changed during training, is a hyperparameter [5]. Given that the model’s performance on the test is often considered a measure of its real-world performance, the test set should only be used once and should not be used to fine-tune any parameters or hyperparameters [5].

Training data is often split into training, validation subsets through a process called K-fold cross-validation [5]. In this process, the training data is split into K-1 equally sized, non-overlapping subsets. For each of K iterations, the training data consists of K-1 folds while the validation data consists of the single, remaining fold [5]. One version of the model is trained per iteration on the K-1 training folds, with its performance tested on the validation fold. By having K iterations, each individual fold has a turn to be the validation fold, testing the model’s ability to generalise on all parts of the training-validation set. The performance of the K models (one from each iteration) is then compared and/or combined [5].

###### Loss Functions and Parameter Tuning

To maximise model performance on the training set, the model’s performance is measured using a loss function, as described above [7]. There are a variety of possible loss functions that can be used for regression problems, with one of the most common bring the mean squared error (MSE), or L2 loss [7]. The MSE is the averaged squared difference between the true output, , and the predicted output, , across all datapoints in the data subset of size *n* [7]. The MSE is defined as follows. One of the major limitations of the MSE is its sensitivity to outliers, as squaring the difference between the true and predicted outputs places high importance on large errors [7].

A widely used variation of MSE is mean absolute error (MAE), also referred to as the L2 loss [7]. MAE measures the average absolute difference between the true and predicted outputs, and is defined more formally in the equation below [7]. Taking the absolute difference instead of the squared difference, MAE is less affected by outliers than MSE [7]. However, MAE is not differentiable everywhere, unlike MSE, presenting possible difficulties to applying gradient based optimisation techniques [7].

###### Performance Metrics and Hyperparameter Tuning

In contrast to a loss function, a performance metric evaluates the model after training is completed [7]. More specifically, a performance metric captures both the model’s accuracy and whether it can generalise to out-of-sample data [7]. Performance metrics are generally less dependent on a model’s specific architecture, as the metrics are often used to compare the performance of different candidate models, architectures, and hyperparameter choices [7]. While MSE and MAE can be used as performance metrics as well as loss functions, there are a variety other widely used metrics [7].

One of the most commonly used metrics is the square root of the MSE, called the root mean squared error (RMSE) [7]. The RMSE is formally defined below [7]. Similar to MSE, the squared function in RMSE heavily penalises outliers, unlike MAE [7]. However, unlike MSE but similar to MAE, RMSE is in the scale of the original data, making it easier to interpret [7].

Another widely used metric is the mean absolute percentage error (MAPE), which calculates the average prediction error as a percentage of the true value [7]. It is defined below [7]. The MAPE score is criticised for being asymmetrical, as always dividing by the true output, , can produce different errors depending on whether the predicted value underestimates or overestimates the true value [7]. For example, predicting a value of 50 if the true value is 100 gives a MAPE of 50% while predicting a value of 100 if the true value is 50 gives a MAPE of 100%, despite the absolute error being the same in both cases. Another limitation of using the MAPE score is that it can become very large or undefined if is close to zero [7]. Additionally, MAPE is strongly affected by outliers, especially if is small, making any error caused by outliers a large relative error [7].

The coefficient of determination, denoted as the R2 score, is another commonly used performance metric [7]. It determines the proportion of variation in the output variable explained by the model [7]. R2 is defined below, where is the mean true *y* value [7]. R2 is equal to 1 if the model explains all variation in the data [7]. However, a high R2 score can sometimes reflect overfitting in the model. R2 is negative if the model performs more poorly than if it simply predicted [7]. Unfortuantely, the R2 is known to be sensitive to bias and can arbitrarily increase with the number of features [7].

#### Linear Regression

One of the most well-known, basic machine learning models is linear regression, which is often described as a ‘line of best fit’ through the data [8]. Model development focuses on minimising the distance between the line and the true datapoints [5]. More formally, linear regression is often used to predict output using a linear combination of *d*-dimensional input feature vectors, [8]. The model is described in the following equation, with the *d*-dimensional weights denoted by [8].

When there are many feature dimensions with large weights, the model can overfit the data, as the model can become arbitrarily complex and fit to noise in the dataset [8]. Thus, many linear regression implementations incorporate a regularisation term, which is added to the loss function to penalise model complexity. More specifically, the regularisation term increases the loss function by some function of the model parameters [5]. To minimise the loss function during the model development stage, training generally involves actions to reduce the regularisation term and thus prevent these parameter values from becoming too large [5]. The L1 norm, or the sum of the parameters’ absolute values, is a commonly used regularisation function [8]. By penalising parameters’ absolute values, it encourages the model to use zero feature weights, thus performing automatic feature selection [8]. Another widely used regularisation function is the L2 norm, which is the sum of the squared parameter values, and thus severely penalises large parameter values [8]. A special version of the linear regression model that combines the L1 and L2 norm is called Elastic Net [8]. The Elastic Net model’s regularisation term is set as follows, where is a hyperparameter that controls the influence of the L1 norm versus the L2 norm [8]:

Linear regression is solely linear in the parameters, meaning the feature variables do not need to be linear [5]. However, the model can still underfit the data if the relationship between the feature variables is non-linear [5]. Thus, other, more complex machine learning models have been developed.

#### Support Vector Machines

Support vector regression defines a function that takes in datapoints and returns predictions that are within distance epsilon from the ground truth [9]. Support vectors are defined as the datapoints falling outside a cylinder of epsilon radius centred around the model’s predictions, and all model predictions are made using a linear combination of the support vectors [9]. Often, the input data is first transformed into a higher dimensionality feature space that enables more effective modelling of non-linear relationships [9].

#### Decision Tree Based Methods

Since their original proposal in the 1960s, decision trees have become an important part of the most widely used learning models [12]. Intuitively, decision tree models function like flowcharts [12]. A regression decision tree is visualised below, with the tree’s internal nodes given by circles and its terminal nodes given by squares. When predicting the output variable for a specific datapoint, the model starts at the root node and applies a logical test to the values of one or more of x’s feature variables. For regression, this test is usually in the form of *feature <= value*, and defines a split [12]. Based on the test’s Boolean result, the model moves to the right or left child node. This process is repeated until the model reaches a terminal node, which is a node that has no children. The terminal node’s value is the model’s prediction [12]. An alternative way to conceptualise decision trees is as a specific partitioning of the input space, where each node partitions the feature space and each new partition is passed down to the node’s children. The tree’s prediction then corresponds to a specific area of the feature input space [12].

age <= 20

age <= 65

age <= 15

False

True

0

5

0

37

Figure 1: Regression decision tree visualisation, where splits are defined in terms of the feature ‘age’ and the model is trying to predict the number of hours worked per week. If the result of the test is True, the model moves to the leftmost node, but if it is False, it moves to the rightmost node. The values inside the terminal leaf nodes are the decision tree’s predictions.

The decision tree’s structure is developed during training, where the series of splits that best predict the outcome variable are chosen [12]. For regression problems, the logical test at each node is determined through finding the split that minimises the mean squared error in the associated child nodes [19]. Traditionally, this is done in a greedy manner, where specific splits are evaluated solely by the effect they have on their children’s error [12].

One of the primary advantages of decision trees is their ability to work with data that has missing values [12], with specific implementations having different methods [19]. CART (Classification and Regression Trees) is one of the classic decision tree implementations and uses ‘surrogate’ splits to deal with missing data [12, 19]. When a datapoint is missing a value in a feature dimension, nodes that partition the input space using that feature dimension instead split using an alternate, related variable in the CART implementation [19]. Additionally, by following a flow-chart like structure, decision trees are highly interpretable, making them extremely value in disciplines that place more emphasis on understanding why machine learning models have made a specific prediction [12]. This interpretability is related to the concept of ‘explainable machine learning’ [12]. Another benefit of decision trees is their relatively low computation cost when compared to other machine learning models [12].

However, a major limitation of decision trees is their propensity to overfit to training data, where the input space is partitioned by overly complex rules that are based on the specific training examples and/or noise [12]. While this would produce strong performance on the training data, it would substantially reduce the model’s generalisability and performance on out-of-sample data [12]. As a result, in general, shallower trees with fewer partitions tend to generalise better, but may have lower performance due to the lower complexity [12].

There are many variations of decision trees in the literature. For example, to better represent more complex functions, studies have explored basing the splits in internal nodes on multiple feature variables and/or having predictive models in their terminal nodes instead of a constant number [12]. Additionally, there has been research into replacing the greedy approach used to determine splits by look-ahead algorithms to avoid suboptimality [12].

#### Ensemble Based Methods

Studies have found that ensemble learning, or combining predictions from multiple models, can produce better predictive performance than using a single model [13]. In particular, ensemble methods can reduce generalisation error when the models being combined, called base estimators or weak learners, are independent and diverse and thus cover a wider range of possible outcomes [14]. Additionally, a single model may become stuck in a local optimum, but if each base estimator in an ensemble model starts in a different place and/or has a different formulation or training trajectory, the ensemble model is better able to escape the local optimum [13]. Ensemble methods also perform well when some combination of base estimators can approximate complex relationships in the data better than any single base estimator [13].

Ensemble models can generally be categorised as a bagging, boosting, voting, or stacking algorithm [14].

##### Bagging

During bootstrap aggregating, or ‘bagging’, predictions from multiple versions of the same type of base estimator are combined [15]. Different versions of the same base estimator are produced by training each new version on a bootstrap replicate of the training set [15]. In other words, datapoints are drawn at random and with replacement from the training set to form independent, bootstrapped datasets of the same size [15]. Then, each base estimator is trained on one of the bootstrap replicates of the dataset, producing an ensemble of base learners whose predictions are combined [15]. For regression tasks, the predictions are generally averaged [15]. Bagging works particularly well when models trained on different versions of the training set are substantially different, allowing the ensemble model to cover a wider variety of outcomes [15]. Additionally, bagging can reduce variability and overfitting by cancelling out noise in the dataset [14]. An example of bagging is combining the predictions of multiple decision trees made on separate, bootstrap versions of the training dataset [14].

###### Random Forest

The Random Forest model is a widely used variation of the basic decision tree-based bagging ensemble [13]. In the Random Forest algorithm, each split in the base decision trees is created using a random subset of features [13]. This modification further reduces overfitting in the ensemble [13]. However, there is no guarantee that an important feature will be used for splitting, potentially causing essential information to be lost [14].

##### Boosting

While bagging trains base estimators independently, boosting ensemble methods train the base estimators sequentially [14]. During boosting, each base estimator in the sequence tries to correct the errors of the previous estimator [14]. Intuitively, this allows the model to focus on learning more difficult relationships in the data [14]. This approach enables each subsequent base estimator to correct the errors of the previous, giving the ensemble model higher prediction accuracy and lower bias [14, 16]. The base estimators in boosting ensembles are often decision trees due to their empirically demonstrated prediction accuracy [16].

There are many implementations of the boosting approach, with one of the simplest being Adaptive Boosting (AdaBoost) [16]. The AdaBoost model initially fits a base estimator to a uniformly weighted dataset [16]. It trains subsequent base learners on copies of the original dataset, where each datapoint is weighted by the previously trained base estimator’s error for that datapoint [16]. Thus, datapoints with higher associated error are weighted more strongly [16].

Gradient boosting is another popular version of the boosting ensemble method, and when the base learners are decision trees, it is referred to as the gradient boosting decision tree algorithm (GBDT) [16]. Studies have shown GBDT model to be accurate, efficient, and interpretable, precipitating its wide use in a variety of disciplines [18]. In gradient boosting, the first base estimator attempts to predict the output variable. Then, each new base estimator is trained to minimise the current model’s loss function [18]. To do so, each new base estimator predicts the negative gradient of the previous estimator’s loss function [18]. This negative gradient indicates the direction of greatest decrease in loss. By learning this direction, the new base estimator can move the ensemble’s prediction in a direction that most reduces its prediction loss [18]. This process is expressed more formally by the equation below, where is the *m*th base estimator in the sequence, is the base estimator trained on the negative gradient of , and is the weight attached to , quantifying its importance [16].

Boosting, unlike bagging, is an additive combination of base learners [17]. Thus, the final prediction from the boosting ensemble model is the sum of predictions from each component base learner [17].

Two of the most widely used GBDT methods are Extreme Gradient Boosting (XGBoost) and Light Gradient Boosting (LightGBM).

###### Extreme Gradient Boosting (XGBoost)

The XGBoost algorithm is a high-performance, scalable GBDT method [16, 17].

In the base gradient boosting method described above, the *m*th base estimator, , predicts the negative first-order gradient of the (*m-1*)th base estimator to move the model’s predictions in the direction that best reduces loss [16]. The XGBoost model takes this a step further. It constructs a 2nd-order Taylor approximation of the current model’s loss function, which involves the loss function’s first and second-order derivatives [17]. The algorithm minimises this Taylor approximation to find the optimal leaf node weights, which are the tree’s predictions [17]. This method provides a more controlled error correction mechanism, as the second-order derivative indicates how quickly the gradient is changing, guiding the amount of change the model should make in response to the gradient [17].

When building the new base estimator, the algorithm must evaluate the value of different logical tests to define the splits at internal nodes [17]. When evaluating a candidate logical test, the algorithm separates the input data into two groups – the data that would push the model to the left child and the data that would push it to the right child [17]. The model then calculates the approximated loss function for each group using the current predictions from the previous base estimators [17]. By minimising this loss, it determines the optimal node weight for the children nodes and can determine the potential reduction in loss produced by this specific split [17].

Another modification that the XGBoost model makes to the base GBDT algorithm is to add an additional regularisation term to the model’s loss function [17]. This regulariser is a function of the number of leaves in the base estimator’s decision tree and the squared absolute values of the leaf node scores [17]. Thus, adding this regulariser to the loss function causes loss to increase when the number of leaf nodes increase [17]. Consequently, the regularisation term penalises model complexity, as it encourages the model to have fewer internal nodes and input space partitions to reduce the number of terminal nodes [17]. XGBoost also supports feature subsampling, similar to the Random Forest implementation, to further reduce overfitting [17].

One of XGBoost’s major strengths is its ability to work with missing, or sparse, data, whereas many tree based methods are only optimised for dense data [17]. When building the ensemble’s base estimators, only datapoints with non-missing values for the relevant feature variable/s are used to optimise the logical tests defining internal nodes [17]. Then, for the datapoints with missing values for the relevant feature/s, the model calculates whether always moving to the left child or the right child results in lower predictive loss [17]. The direction with lower associated loss is defined as the ‘default direction’ and is chosen whenever the model encounters missing values for the relevant feature/s at that specific internal node [17].

One of the key hurdles to constructing GBDT ensembles is the need to trial every possible internal node split to determine the optimal base learner tree structure [17]. In the exact greedy algorithm approach, all possible splits for all features must be tested, with the split’s performance quantified by the amount that the split reduces loss [17]. While this algorithm has strong performance, it is computationally demanding, especially when considering all possible splits for continuous, numeric feature variables and when the data does not fit into memory [17]. The approximate algorithm was introduced to address this problem [17]. This algorithm splits the continuous features’ distributions into percentiles, with the differences between percentiles using as candidate split points, thus reducing the number of possible splits needed to be evaluated [17].

###### Light Gradient Boosting Machine (LightGBM)

LightGBM is another, commonly used GBDT algorithm that places an even greater focus on maximising computational efficiency, again identifying the computational cost of evaluating every possible candidate split [18].

One of the main modifications proposed by the LightGBM algorithm is gradient-based one-side sampling (GOSS) [18]. GOSS reduces the number of samples used to determine the internal node splits. Instead of using all data points to determine each split, GOSS uses the most informative data points and samples a subset of less informative points to maintain the same general data distribution [18]. Its choice of samples is derived from the observation that datapoints associated with small gradients offer smaller potential reduction in error and are thus less useful for increasing model performance [18]. Using this observation, GOSS takes all datapoints with gradient greater than a certain threshold and samples randomly among the remaining datapoints with smaller gradients [18]. It uses this subsampled dataset to determine the internal node split, increasing computational efficiency [18].

Another innovation used in the LightGBM model is exclusive feature bundling, which can be applied to sparse feature spaces (EFB) [18]. More specifically, sparse feature spaces generally have mutually exclusive feature variables, which are defined as groups of features where no more than one feature takes a non-zero value at the same time [18]. Groups of mutually exclusive features can be ‘bundled’ together into a single feature, further increasing computational efficiency [18].

##### Voting

The voting ensemble model is another method of aggregating the predictions from multiple base estimators [14]. However, as opposed to bagging, in this case the base estimators can have different model architectures, and all models are trained on the same dataset [14]. In regression, the final prediction from a voting ensemble model is the unweighted or weighted average of the base estimators’ predictions [14]. Using a weighted average allows more importance to be placed on specific base estimators [14]. The voting strategy benefits from combining the strengths of each model class in the ensemble but can show lower performance if the base estimators are too similar [14].

##### Stacking

In a stacking ensemble model, predictions from the base estimators serve as inputs to a meta-learning model, which combines the inputs to produce a single, final output [13]. The meta-learner can have a different structure from the base estimators, with examples of meta-learners being Random Forests, support vector machines, linear regressors, and even neural networks [14]. The meta-learning model can learn how to most effectively combine the predictions from the base estimators, and which base estimators are the most important [14]. While stacking ensembles can improve performance in similar ways to those discussed above, they can be computationally expensive to train, as all of the base estimators and meta-learning model must be fit to the data [14].

Citations:

1. Independent Group of Scientists appointed by the Secretary-General, Global Sustainable Development Report 2023: Times of crisis, times of change: Science for accelerating transformations to sustainable development, (United Nations, New York, 2023).
   1. <https://sdgs.un.org/sites/default/files/2023-09/FINAL%20GSDR%202023-Digital%20-110923_1.pdf>
2. <https://icdcdn.who.int/icd11referenceguide/en/html/index.html#standards-and-reporting-requirements-related-for-maternal-mortality>
3. Trends in maternal mortality estimates 2000 to 2023: estimates by WHO, UNICEF, UNFPA, World Bank Group and UNDESA/Population Division. Geneva: World Health Organization; 2025. Licence: CC BY-NC-SA 3.0 IGO.
4. <https://www.thelancet.com/journals/langlo/article/PIIS2214-109X(24)00560-6/fulltext>
   1. Global and regional causes of mmr
5. <https://www.nature.com/articles/s41580-021-00407-0?fromPaywallRec=false>
   1. Information about ML basics
6. <https://www.science.org/doi/pdf/10.1126/science.aaa8415>
   1. Intro to ML
7. <https://link.springer.com/article/10.1007/s10462-025-11198-7>
   1. Loss functions and performance metrics
8. <https://academic.oup.com/jrsssb/article/67/2/301/7109482>
   1. Elastic net
9. <file:///Users/rosalitarosenberg/Downloads/B_STCO.0000035301.49549.88.pdf>
   1. A tutorial on support vector regression
   2. Smola and Sch¨olkopf
10. <https://academic.oup.com/bmb/article/121/1/121/2926155>
    1. Recording MMR methods
11. <https://onlinelibrary.wiley.com/doi/epdf/10.1002/sim.9335?getft_integrator=tfo&src=getftr&utm_source=tfo>
    1. BMIS methods
12. <https://link.springer.com/article/10.1007/s10462-022-10275-5>
    * Decision tree review
13. <https://www.sciencedirect.com/science/article/pii/S095219762200269X?pes=vor&utm_source=scopus&getft_integrator=scopus>
    1. Ensemble review
14. <https://pmc.ncbi.nlm.nih.gov/articles/PMC10298658/#abstract1>
    1. Ensemble review medicine context
    2. Mahajan, P.; Uddin, S.; Hajati, F.; Moni, M.A. Ensemble Learning for Disease Prediction: A Review. Healthcare 2023, 11, 1808. https://doi.org/10.3390/ healthcare11121808
15. <https://link.springer.com/article/10.1007/BF00058655>
    1. Bagging predictors
16. <https://journalofbigdata.springeropen.com/articles/10.1186/s40537-025-01071-3>
    1. Gradient boosting trees
17. <https://arxiv.org/pdf/1603.02754>
    1. Xgboost paper
18. <https://proceedings.neurips.cc/paper_files/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf>
    1. Lightgbm
19. <https://onlinelibrary.wiley.com/doi/epdf/10.1111/insr.12016>
    1. Decision tree (missing, gain metrics)
20. <https://dl.acm.org/doi/pdf/10.1145/3292500.3330701>
    1. Optuna
    2. Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, Masanori Koyama. 2019. Optuna: A Next-generation Hyperparameter Optimization Framework. In The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD ’19), August 4–8, 2019, Anchorage, AK, USA. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3292500.3330701>
21. <https://bmcpregnancychildbirth.biomedcentral.com/articles/10.1186/s12884-023-06077-4>
    1. Maternal mortality recording errors
22. <https://www.tandfonline.com/doi/epdf/10.1080/2330443X.2023.2286313?needAccess=true>
    1. About the latest WHO model
23. <https://obgyn.onlinelibrary.wiley.com/doi/epdf/10.1111/1471-0528.12735>
    1. Obstetrics – maternal and perinatal morbidity
24. <https://pmc.ncbi.nlm.nih.gov/articles/PMC11010755/#section14-21501319241245847>
    1. Review of AI in healthcare
25. <https://journalofbigdata.springeropen.com/articles/10.1186/s40537-023-00839-9#:~:text=The%20gradient%20descent%20(GD)%20is,computationally%20expensive%20for%20large%20datasets>.
    1. Gradient descent overview
26. <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=363438>
    1. Gradient descent formula