Predicting antibiotic exposure in IC patients

Ravish Gangapersad (4627369)

September 1 2021

Abstract

In this paper, we want to create a prediction model for target attainment and a scoring system based on these models. The Random Forest model, Logistic Regression model, and Naive Bayes model are employed to achieve this. Classification trees and predictors from the random forest model are used to create the scoring system. The theory behind each model and scoring system is described in detail. The properties of proper scoring rules were checked when making our scoring systems. We have made three models which can reasonably predict target attainment. The logistic regression model performed the best for internal and external validation, with an AUC of 0.84 and 0.80, respectively, whereas the RF model performed slightly worse. The AUC of the random forest differed by 0.01 in both validation sets. The scoring system performed reasonably with an AUC of 0.74. These models and the scoring system perform reasonably well and will help the medical practitioner assess if an ICU patient will attain their beta-lactam antibiotic target.

1. Introduction

Beta-Lactam antibiotics are extensively used in critically unwell patients who have infections. Only 40 to 60% of critically ill patients treated with beta-lactam antibiotics reach the target at which the antibiotic is effective [1, 2].

To understand target attainment, we first need to define the minimum inhibitory concentrations (MICs), defined as the lowest concentration of an antibiotic that will inhibit the visible growth of a microorganism in vitro after overnight incubation. Beta-lactam antibiotics are classified as time-dependent. The time dependence means that these antibiotics require the drug concentration to be above the MIC for a certain percentage of the dosing interval to effectively kill the organism (time above MIC)[3].

At the Erasmus Medical Center (Erasmus MC), we have a data set of Intensive care Unit (ICU) patients. All the relevant information to build a model for target attainment is available. This data has nine different antibiotics in two distinct classes (Ciprofloxacin and eight beta-lactam antibiotics). For our purpose, we will only look at the target attainment of the beta-lactam antibiotics. This data set is a combination of the EXPAT [1] and Dolphin trial [4]. Furthermore, we also have an external data set which is a combination of other beta-lactam antibiotics studies [5, 6].

Our main objective is to make a model that will predict whether a patient will attain the target at which the antibiotic has an effect (100 % time above MIC), with readily available predictors on the day of starting the antibiotic (day 0). The models which will be used are the Random Forest (RF) model, logistic regression (LR) model and the Naive Bayes (NB) model. The RF model and NB model are non-linear models which will be able to detect a non-linear relationship between the predictors. The logistic regression is a linear model that detects linear relationships in the predictors. The models are made on an internal data set and further validated on external data.

Another objective is to see if we can make a scoring system from our predictive models that can be easily implemented in hospitals. This will be done using decision trees, where the predictor variables are obtained from the RF algorithm.

The report has the following structure. The methods section will show the mathematical background and methods used to make the model and scoring system. The results we achieved using the provided methods will be discussed in the third section. Finally, we will have a conclusion and discussion to discuss our findings.

2. Methods

In this section, we will introduce the models we will use and explain how we filtered the data and found the best set of prediction variables. Furthermore, we will also describe how the different models are compared. Everything was done using the programming language R (R version 4.11; The R Foundation for Statistical Computing, Vienna, Austria) [7]. The two-sided significance level was set to 0.05.

2.1. Logistic regression

Logistic regression [8] works very similar to linear regression [9], but with a binomial response variable. Compared to Mantel-Haenszel Odd Ratio (OR), the most significant advantage is the ability to use continuous explanatory factors and the ease with which more than two explanatory (predictor) variables may be handled simultaneously. This last property, albeit seemingly insignificant, is critical when we want to know how different explanatory variables affect the target variable. We neglect the covariance among variables and are prone to confounding effects when looking at various explanatory factors separately.

Logistic regression will model the chance of an outcome based on individual characteristics. The following equation gives the logistic regression,

$$\log\left(\frac{\pi}{1-\pi}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_m x_m,$$

where π indicates the probability of an event, β_i are the regression coefficients associated with the reference group and the x_i the predictor variables. At this point, an important concept must be highlighted. The reference group, represented by β_0, is constituted by those individuals presenting the reference level of every variable $x_1 \dots x_m$.

2.2. Naive Bayes

For predicting target attainment, we used the Naive Bayes method. The Bayes theorem is defined as the following:

Theorem 2.1. *Let A and B be events with $P(B) \neq 0$ then we can define*

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)},$$

where $P(A|B)$ is the conditional probability of event A occurring given that B is true and $P(B|A)$ is the conditional probability of event B occurring given that A is true. Furthermore, $P(A)$ and $P(B)$ are the probabilities of A and B occurring independently of one another (the marginal probability). This is called the Bayes theorem.

Now, by using the theorem, we can introduce the Naive Bayes model [10]. We have the predictor variable X and the target variable y. Then, we can use the Bayes theorem to get $P(y|X)$, which is known as the posterior probability of the target variable y given predictor variable X. We have $P(y)$, which is the prior probability of the target variable y. Lastly, we have the likelihood $P(X|y)$, which is the probability of the predictor variables given the target variable $\mathbf{y}$.

Suppose target variable y and N predictor variables $X_1, \ldots, X_N$. By implementing the Bayes theorem, we obtain

$$P(y|X_1, \ldots, X_N) = \frac{P(X_1, \ldots, X_N|y)P(y)}{P(X_1, \ldots, X_N)}, \tag{1}$$

We assume 'naive' independence meaning that

$$P(X_i|y, X_1, \ldots, X_{i-1}, X_{i+1}, \ldots X_N) = P(X_i|y) \tag{2}$$

Using Equation (2) in Equation (1) we get

$$P(y|X_1, \ldots, X_N) = \frac{\prod_{i=1}^{N} P(X_i|y)P(y)}{P(X_1, \ldots, X_N)}. \tag{3}$$

Now, we have set up the framework for the Naive Bayes probability model. The Naive Bayes classifier combines the Naive Bayes probability model with a decision rule, which means the class for prediction is picked by the maximum a posterior (MAP) decision rule. This is done in the following way. It should be noted that $P(X_1, \ldots, X_N)$ is a normalization constant. Thus, we can drop that term and approximate the posterior by

$$P(y|X_1, \ldots, X_N) \propto \prod_{i=1}^{N} P(X_i|y)P(y).$$

Using the obtained expression, the Naive Bayes Classification Rule (MAP) can be expressed as

$$\hat{y} = \arg\max_{y} \prod_{i=1}^{N} P(X_i|y)P(y).$$

2.3. Random Forest Algorithm & Decision Trees

For predicting sub-therapeutic concentrations (target non-attainment) we used the random forest model [11]. This model was chosen because it is excellent in finding non-linear relationships in the predictor variables. First, we discuss tree-based models because they form the building blocks of the random forest algorithm.

Decision trees can be used for regression and classification. A tree-based model involves recursively partitioning the given data set into two groups based on a specific criterion until a predetermined stopping condition is met. In this section, we go into more detail on how this is done. To explain this, we introduce simple functions.

Definition 2.2. *For $S \subset \mathbb{R}$, the **characteristic** (or **indicator**) **function of** S is the function $\mathbb{1}_S(x) : \mathbb{R} \to \mathbb{R}$ defined by*

$$\mathbb{1}_S(x) = \begin{cases} 1 & \text{if } x \in S \\ 0 & \text{otherwise} \end{cases}$$

*A function $\widehat{\phi} : \mathbb{R} \to \mathbb{R}$ is called a **simple function** if it is a finite linear combination of characteristic functions. This means that there exists $n \in \mathbb{N}$, numbers $a_1, \ldots, a_n \in \mathbb{R}$, and subsets $S_1, \ldots, S_n \subset \mathbb{R}$ such that $\widehat{\phi} = \sum_{i=1}^{n} a_i \mathbb{1}_S$*

In regression and classification trees, we are looking for a simple function $\widehat{\phi}(x)$, where x are predictors/inputs and ϕ is the output (reality). The main idea is to partition the input space into disjoint regions and fit a simple function/model on these regions. Ideally, we would like to find optimal partitions which minimize the training error, but this is computationally infeasible. We use recursive binary splitting, which is defined in the following way,

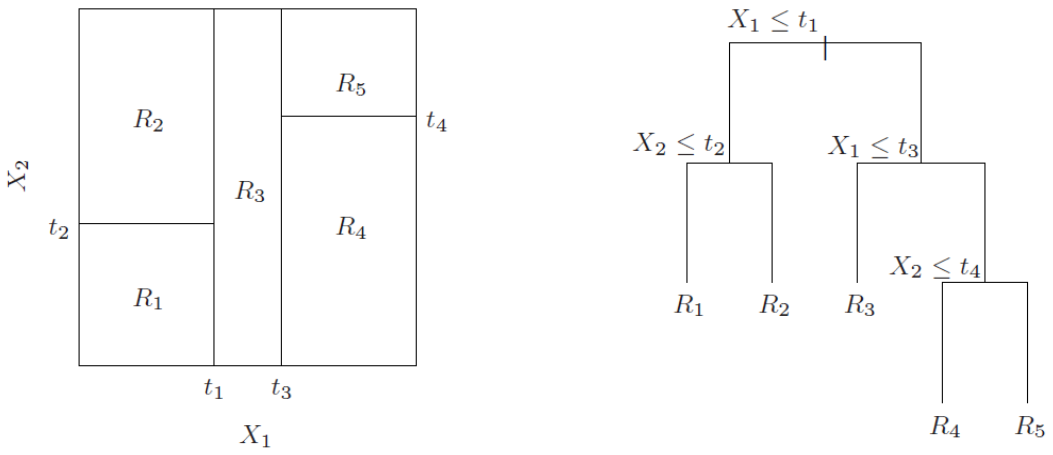
Definition 2.3. ***Recursive binary splitting*** *:*

1. *Select one input variable x_j and a cut point s. Partition the input space into two half-spaces,*

$$\{x : x_j < s\} \text{ and } \{x : x_j \geq s\}$$

2. *Repeat the splitting for each region R_j until some stopping criterion is met (for example, no region contains more than 5 training data points).*

In Figure 1 we see a visualization of the Binary splitting process.



(a) Partition of input spaces

(b) Tree representation of the partition

Figure 1: Binary splitting: with regions R_j, cut points t_j and input variables X_1, X_2.

We summarize decision trees using Figure 2.

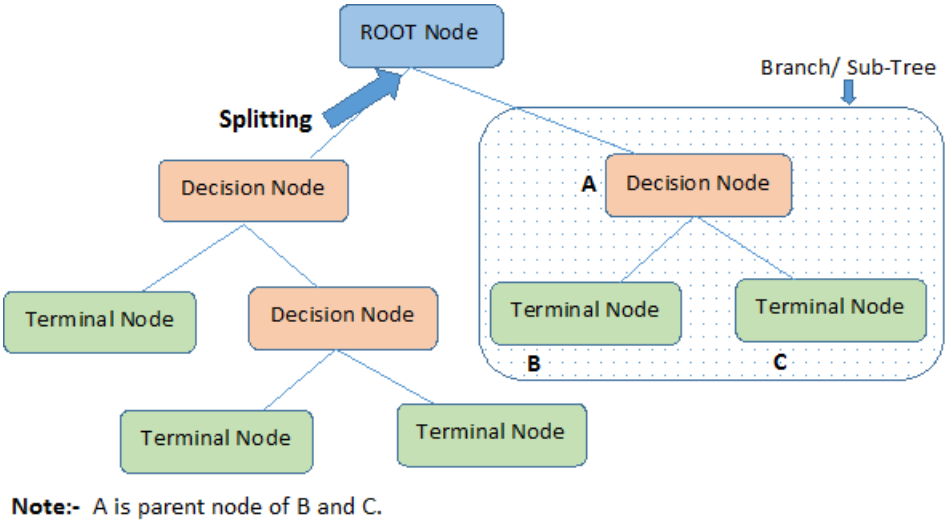


Figure 2: Visualization of a decision tree.

If we look at Figure 2 we see that we begin at a root node. This represents the whole data set. We then have a split, which divides a node into two or more sub-nodes. If a node splits, it is called a decision node. Otherwise, it is called a terminal node or leaf node. Removing nodes from a decision tree is called pruning.

2.3.1. Regression Decision Trees

Using everything we have defined above, we will introduce regression trees using just two regions. So we get for any j (jth input variable) and s (cut point)

$$R_1(j,s) = \{x|x_j < s\} \text{ and } R_2(j,s) = \{x|x_j \geq s\}$$

We then seek (j,s) that minimize the square loss function

$$\sum_{i:x_i \in R_1(j,s)} (\phi_i - \widehat{\phi}_1(j,s))^2 + \sum_{i:x_i \in R_2(j,s)} (\phi_i - \widehat{\phi}_2(j,s))^2,$$

where

$$\begin{aligned}
&\phi_i \text{ is reality/ outcome,} \\
&\widehat{\phi}_1 = \text{average}\{\phi_i : x_i \in R_1(j,s)\} \\
&\widehat{\phi}_2 = \text{average}\{\phi_i : x_i \in R_2(j,s)\}
\end{aligned}$$

This optimization problem is solved by evaluating all possible splits. Now we have introduced regression trees.

2.3.2. Classification Decision Trees

For classification trees we first have to compute the proportion of the data points from class m in node k corresponding to region R_k with N_k points, defined as

$$p_{km} = \frac{1}{N_k} \sum_{i:x_i \in R_k} \mathbb{1}_{\phi_i = m}$$

Instead of using squared loss as in the regression trees we, use measures more suitable for categorical outputs, so we have:

1. Misclassification error: $L(p_k) = 1 - \max_k p_{km}$
2. Entropy/deviance: $L(p_k) = -\sum_{m=1}^{K} p_{km} \log(p_{km})$
3. Gini index: $L(p_k) = \sum_{m=1}^{K} p_{km}(1 - p_{km})$

Each split is chosen to minimize sum of losses over possible splits:

$$L(p_1) + L(p_2).$$

The prediction for each region is chosen by majority vote within the region, this means we choose the class with the highest p_{mk}

2.3.3. Random Forest

One of the big drawbacks of decision trees is that it is prone to overfit. This means that the model works very well for the train data set, but for a new data set the same model can perform not so well. The solution proposed for this is the random forest model. Below we have the mathematical definition of random forest.

Definition 2.4. *A **random forest** is a classifier consisting of a collection of tree structured classifiers $\{h(x, \sigma_k), k = 1, \dots\}$ where the $\{\sigma_k\}$ are independent identically distributed random vectors and each tree casts a unit vote for the most popular class at input x.*

Random Forest model can be made using the following Algorithm:

Implementation of Random Forest.

Given:
- Z: training set with n instances, p predictor variables and one target variable
- K: number of classes of the target variable
- B: number of classifiers in random forest

For b = 1 to B:
    Draw a bootstrap sample Z_b^* of size N from the training data.
    Grow a random-forest tree T_b to the bootstrapped data, by re-cursively repeating the following steps, until the minimum node size N_{min} is reached:
        Select m variables at random from the p predictor variables.
        Pick the best variable/ split-point among the m.
        Split the node into two daughter nodes.
Output the ensemble of trees $\{T_b\}_{b=1}^{B}$.
Test Phase:
To make a prediction at a new point x:
Regression:

$$\hat{y}_{rf}(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x)$$

Classification:

$$\hat{C}_{rf}^{B}(x) = \text{majority vote}\{\hat{C}_b(x)\}_1^B$$

We call m, N_{min} and B the hyperparameters of the RF model. These parameters will be tuned using 2 times repeated 10-fold cross validation (CV). This means that the data set on which we train the model is split into 10 subsets. These ten are then used to train and test the model. So 9 of these data sets are to train the model, and the 10th is to test the model. This process continues until all ten have been used as the validation set. After each iteration, the error is calculated. In our case, we train and test the model 2 times (10-fold) and calculate the error ten times. The mean of these errors is calculated, which tells us how good the model performs on average. This whole process is then repeated two times where the mean of the resulting two averaged errors are taken and used as our resulting error.

2.4. Scoring rules

In this section, we will go over various scoring rule definitions and properties that will prove useful when we create our own scoring rule.

Definition 2.5. *Suppose $Y : \Omega \to X$ and $G : \Omega \to Z$ are two random variables defined on a sample space Ω with $f_Y : X \to V$ and $f_G : Z \to V$ as their corresponding density (mass) function, in which Y is a forecast target variable and G is the random variable generated from a forecast schema. Also, assume that the $Y = y$, for, $y \in X$ is the realized value. A scoring rule is a function such as $S : Z \times X \to \mathbb{R}$ which calculates the distance between G and y (S(G,y)). [12, 13, 14]*

Definition 2.6. *Let assume we have a decision rule $S : Z \times X \to \mathbb{R}$. With $Y : \Omega \to X$ the forecast target variable, with probability density (mass) function $f_Y : X \to V$. Then we can define the expected score as*

$$\bar{S}(G;Y) = \mathbb{E}_{y\sim Y} S(G, y).$$

For continuous random variables we have

$$\bar{S}(G;Y) = \mathbb{E}_{y\sim Y} S(G, y) = \int_{-\infty}^{\infty} f_Y(y) S(G, y)\,\mathrm{d}y,$$

and for discrete random variables we have

$$\bar{S}(G;Y) = \mathbb{E}_{y\sim Y} S(G, y) = \sum_{y} f_Y(y) S(G, y)$$

Definition 2.7. *A scoring rule $S : Z \times X \to \mathbb{R}$, is proper if for all the forecast target variables Y and all random variables G generated from a forecasting scheme not equal to Y have the following property,*

$$\bar{S}(Y;Y) \geq \bar{S}(G;Y).$$

Furthermore the scoring rule is strictly proper for all Y and all $G \neq Y$ if

$$\bar{S}(Y;Y) > \bar{S}(G;Y).$$

Now if we assume that $Y \sim Bernoulli(q)$ and apply the definition 2.7 we see that we need to have that

$$\arg\min_{Y} \bar{S}(G;Y) = q,$$

to have a proper scoring system. If the q is unique, then we have a strictly proper scoring rule. [12]

2.4.1. Our own scoring rule

Suppose we have a set of regressor random variables (RV) $\{X_1, \dots X_n\}$ with $n \in \mathbb{N}$ and binary outcome RV Z which consists of $z_i \in \{0, 1\}$. Then the scoring system is defined in the following way: First we make a decision tree for each individual regressor with the outcome being Z. The decision trees that we make should adhere to the following rules:

1. The loss function should be minimized.
2. Each tree will be given weights, the weights are depended on the ratio of the different classes in our outcome Z and are chosen in such a way that they sum up to 1.
3. Then we prune the decision tree using 10-fold cross validation (CV) [15] and repeat this 1000 times, we pick the tree with the smallest CV error over all 1000 repeated 10-fold CV and in the case of a tie we pick the tree with the least splits.

After the decision tree is made for each individual regressor X_i, scores will be given. This is done in the following manner:

The final tree of each regressor will be checked, and the probability given to the last class times 100 will be used as the score we give to each final leaf. Then each all the scores for each tree are added, this is our final scoring system.

Note that we can choose the loss function and also some other parameters inside the decision tree such as min split, max depth.

The weights are calculated using the following formula:

$$W_{class_1} = \frac{1}{M \cdot T_{class_1}} \qquad \dots \qquad W_{class_M} = \frac{1}{M \cdot T_{class_M}}$$

where

M (= 2) is the number of classes in the outcome variable Z,
W_{class_1} is the weights for the class 1,
W_{class_M} is the weights for the class M,
T_{class_1} is the number of people of class 1 and
T_{class_M} is the number of people class M

Now using this we get the following scoring rule. Let's have N decision trees made using the scheme described above, with $\mathbf{x} = \{x_i\}_{i=1}^{G}$, with G the number of data points.

We first have to compute the proportion of the data points from class m in leaf node k corresponding to region R_k with N_k points, this is defined as

$$p_{km}(\mathbf{x}) = \frac{1}{N_k} \sum_{i:x_i \in R_k} \mathbb{1}_{z_i = m} \qquad \text{, where } z_i \text{ is the reality/outcome.}$$

Now using this we get

$$Sc(M, \mathbf{x}, k, z_i, \text{prior}) = (1 - \text{prior}) \cdot z_i \cdot 100 \cdot p_{kM}(\mathbf{x}) + (1 - z_i) \cdot \text{prior} \cdot 100 \cdot (1 - p_{kM}(\mathbf{x})),$$

with prior, the prior probability that is chosen for class M. The Score above is for 1 tree. There are N trees, that means we have N predictor variables. So we get

$$F_{Sc}(M, C, N, \bar{\bar{X}}, z_i, \text{prior}) = Sc(M, \mathbf{x}_1, u_1, z_i, \text{prior}) + Sc(M, \mathbf{x}_2, u_2, z_i, \text{prior}) + \cdots + Sc(M, \mathbf{x}_N, u_N, z_i, \text{prior}),$$

where

$\bar{\bar{X}}$ is $\{\mathbf{x_1}, \cdots, \mathbf{x_N}\}$,
$C = \{u_j\}_{j=1}^{N}$, is the specific leaf node corresponding to each predictor variable

After we have all the $F_{Sc}(M, C, N, \bar{\bar{X}}, z_i, \text{prior})$ with all combinations of the leaf nodes, we get the vector

$$\mathbf{V_{F_{Sc}}} = [F_{Sc}(M, C_1, N, \bar{\bar{X}}, z_1, \text{prior}) \dots F_{Sc}(M, C_G, N, \bar{\bar{X}}, z_G, \text{prior})],$$

where G is the number of data point and $[C_1, \dots, C_G] = \left\{\{u_j^1\}_{j=1}^{N}, \dots, \{u_j^G\}_{j=1}^{N}\right\}$.

We then use this to define the normalized score, which is,

$$F_{NS} = \frac{F_{Sc}(M, C, N, \bar{\bar{X}}, z_i, \text{prior}) - \min \mathbf{V_{F_{Sc}}}}{\max \mathbf{V_{F_{Sc}}} - \min \mathbf{V_{F_{Sc}}}} \times 100. \tag{4}$$

Another way to get the same normalized is score is by defining the following score for each tree,

$$S_{cN}(M, \mathbf{x}, k, z_i, \text{prior}) = (1-\text{prior}) \cdot z_i \cdot 100 (p_{kM}(\mathbf{x}) - \min_l \{p_{lM}(\mathbf{x})\}) + (1-z_i) \cdot \text{prior} \cdot 100 (1 - p_{kM}(\mathbf{x}) - \min_l \{1 - p_{lM}(\mathbf{x})\}),$$

with $l \in \mathbb{N}$.

After this we can combine the scores for N trees and define

$$\begin{aligned} F_{ScN}(M, C, N, \bar{\bar{X}}, z_i, \text{prior}) = {} & S_{cN}(M, \mathbf{x}_1, u_1, z_i, \text{prior}) + S_{cN}(M, \mathbf{x}_2, u_2, z_i, \text{prior}) + \dots \\ & + S_{cN}(M, \mathbf{x}_N, u_N, z_i, \text{prior}). \end{aligned} \tag{5}$$

If we use Equation (5) we can get the same normalized score as we had in Equation (4),

$$F_{NS} = \frac{F_{ScN}(M, C, N, \bar{\bar{X}}, z_i, \text{prior})}{\max \mathbf{V_{F_{ScN}}}} \times 100,$$

with

$$\mathbf{V_{F_{ScN}}} = [F_{ScN}(M, C_1, N, \bar{\bar{X}}, z_1, \text{prior}) \dots F_{ScN}(M, C_G, N, \bar{\bar{X}}, z_G, \text{prior})],$$

the vector with all the scores.

2.4.2. Properties of our own scoring system

First we will be determining the expected score for our data. We find that the distribution of our target variable Z is Bernoulli distributed with parameter $q = \mathbb{E}(Z)$. If this is the case, then the expected score can be calculated as the following

$$\begin{aligned} \bar{S}(F_{Sc}; Z) &= \mathbb{E}(S(F_{Sc}, z)) = q \cdot F_{Sc}(M, C_D, N, \bar{\bar{X}}, 1, \text{prior}) + (1 - q) \cdot F_{Sc}(M, C_D, N, \bar{\bar{X}}, 0, \text{prior}) \\ &= q \left(Sc(M, \mathbf{x}_1, u_1, 1) + \cdots + Sc(M, \mathbf{x}_N, u_N, 1)\right) + (1 - q) \left(Sc(M, \mathbf{x}_1, u_1, 0) + \cdots + Sc(M, \mathbf{x}_N, u_N, 0)\right) \\ &= 100 \cdot q \cdot (1 - \text{prior}) \cdot (p_{\mu_1 1}(\mathbf{x}_1) + \cdots + p_{\mu_N 1}(\mathbf{x}_N)) + 100 \cdot (1 - q) \cdot \text{prior} \cdot (1 - p_{\mu_1 1}(\mathbf{x}_1) + \cdots + 1 - p_{\mu_N 1}(\mathbf{x}_N)) \\ &= 100 \left(\sum_{i=1}^{N} p_{u_i 1}(q - \text{prior}) - N \cdot \text{prior} \right) \end{aligned}$$

To have a proper scoring rule, we must have that

$$q = \arg\min_{p \in [0,1]} \mathbb{E}(S(F_{Sc}, z))$$

So we take the derivative of $\bar{S}(F_{Sc}; Z)$ with respect to the $\sum_{i=1}^{N} p_{u_i 1}$ and set that to zero, so we get

$$\frac{\mathrm{d}\ \left(100\left(\sum_{i=1}^{N} p_{u_i 1}(q - \text{prior}) - N \cdot \text{prior}\right)\right)}{\mathrm{d}\ \left(\sum_{i=1}^{N} p_{u_i 1}\right)} = q - \text{prior} = 0 \implies q = \text{prior}$$

So we have that if $q = \text{prior}$ then we have that our scoring rule is proper. And if this minimum is unique, we have that our scoring rule is strictly proper. In the results section, we will give some examples of proper scoring rules and describe our own scoring rule on the data.

2.5. Internal & External Data

In this section, we will explain how we described our internal and external data. On the internal data we build our model. The external data is used to validate the model. All the variables of both data sets are categorized into two groups: target attained and target not attained. Furthermore, we have listed their mean, standard deviation, median, and the number of patients for each variable. Lastly, we wanted to know whether there was a statistically significant difference between the groups, so we used the Wilcoxon sum rank test for the continuous variables, the chi-squared test for the categorical variables with categories of more than five observations and the Fisher exact test with categories of less than five observations. Lastly, some histograms were made to see how the internal data was distributed.

2.5.1. Random Forest model for internal Data

Our first model was built using the random forest Algorithm in R. The package h2o (version 3.36.0.4) was used to build this model. Using the h2o package, we could also optimize the hyperparameters of the RF model. The loss function we used for our model was the log loss. The EXPAT data contained 147 patients with beta-lactam antibiotics, while the Dolphin contained 300 patients. The two data sets were combined, with the Ciprofloxacin patients being excluded because we wanted to make a model for beta-lactam antibiotics. Furthermore, we excluded Renal Replacement Therapy (RRT) because patients with RRT were on a machine that substituted the kidney blood filtering mechanism, which meant that the Serum creatinine levels of these patients were not reliable. The metrics of the final data set we used can be seen in Table 1.

After removing these patients, the data was split into a train (80 % of the data) and a validation set (20 % of the data). This was done so that the ratio of people attaining the target and not attaining the target was equal in the validation and the train set. So, if we have 100 patients in the train set who attain the target and 50 who do not, we have 2 out of 3 patients who attain the target. The exact ratio was then also present in the validation set. Because we have eight different beta-lactam antibiotics, we want them to be equally present in the validation and train set. So we have ensured that the ratio of the beta-lactam antibiotics and the patient with Target (non)attainment are equally distributed over the train and validation set. We will call this process stratification. This was done using the package splitstackshape (version 1.4.8).

The hyperparameters were tuned using 10-fold (2 times) repeated cross-validation [15], and the model was also trained using this process. Using this process, we found the best possible parameters for our model. We also used early stopping when training the RF model, which means the model would stop if the simple moving average of the set number of rounds of the stopping metric did not change during the stated number of rounds. We choose 30 as the number of rounds in our case, which we found a reasonable number of rounds. This is done to avoid overfitting the model on the training data. The stopping metric, the area under the curve (AUC) of the receiver operator curve (ROC). In the next section, we will explain the AUC and ROC.

The variable selection of random forest was done using the Boruta algorithm [16]. We gave this algorithm the predictor variables age, length, serum creatinine, type of antibiotic, SOFA score, serum urea day 0, the dose given to the patient, dosing frequency of the patient, APACHE score, CRP day 0, weight, sex, albumin, WBC, fluid balance, number of days on ICU before antibiotic given and sepsis. This algorithm found that the set of age, sex, serum creatinine day 0, serum urea day 0, sepsis, SOFA score, the dose of antibiotic, albumin, APACHE score, frequency of dosing antibiotic, CRP day 0 and type of antibiotic were relevant variables to predict target attainment. We wanted to make a model that performed reasonably well with the least amount of predictor variables. To do this, we again used the Boruta algorithm, but with different unique combinations of the attained variables. So, for example, there were 495 unique combinations of four predictor variables out of the twelve. We found that the best performing combination was age, sex, serum creatinine day 0 and type

of antibiotic. To check whether the Boruta algorithm performed well, our own variable selection procedure was done. We added and removed variables and let the model run and check for which variables the model worked the best (brute force approach). Out of this analysis, we found that the output of the Boruta Algorithm was reliable.

The last way we looked if these variables were reliable was by checking them in multivariate logistic regression and found that all variables, except antibiotics Ceftazidime & Meropenem, were statistically significant ($p<0.05$). One of the reasons that the antibiotic Ceftazidime was insignificant could be that there were very few patients that got this antibiotic. Only one patient did not attain the target. Meropenem is also statistically insignificant, but only just. These variables could still have a marginal effect on the outcome, so we decided to keep these variables in the model.

Following this, the predictors we chose were age, length, serum creatinine, and type of antibiotic. With the outcome being target non-attainment, if we get yes as a prediction, we did not meet the target and vice versa.

The model was trained on the train set and validated on the validation set. Because these sets are made using the stratification process, we want to check if, for different train and validation sets of the data, the model would perform as well. So we did a 1000 times repeated cross-validation of the train (80 %) and validation set (20%). One thousand was chosen because it was a large number of cross-validations, which would give us a good indication of the average performance of the model. The averaged results are shown in the results section.

2.5.2. Logistic model for internal Data

Our second model was built using logistic regression in R. We used the package caret (version 6.0-92). The function train was used from this package, with the method parameter being the general linear model (glm) and the family parameter being binomial with link function logit. The train and validation set were the same as for the RF model. The predictors we chose for this model were the same as we had for the RF model. These were age, sex, serum creatinine on day 0 and type of antibiotic. The average performance of the model over 1000 times repeated cross-validation was also calculated for this model.

2.5.3. Naive Bayes model for internal Data

The last model was built using the Naive Bayes (NB) algorithm in R. The package naivebayes (version 6.0-92) was used. From this package, we used the function naive_bayes, with the prior parameter equal to the prevalence of target attainment. This means that if we have 20 out of 50 patients that attain the target, then the prior that we will use is 0.4 for target attainment and 0.6 for target non-attainment. The train and validation set were the same as for the RF model. The predictors we chose for this model were the same as we had for the RF model. Finally, we also looked at the average performance over 1000 times repeated cross-validation.

2.5.4. Model validation

In this chapter, we will discuss which methods we used to validate the model. To check if the model had a good performance, we made use of the Receiver Operator Curve (ROC) [17]. The random forest Classification model gives votes to each class. These votes can be interpreted as probabilities. Using these weights, we can create the ROC curve. This curve has on the y-axis the sensitivity and the x-axis specificity. The sensitivity and specificity are calculated using the following formulas,

$$\text{Sensitivity} = \frac{TP}{TP+FN} \text{ and Specificity} = \frac{TN}{FP+TN},$$

where

TN is true negative,
TP is true positive,
FP is false positive,
FN is false negative.

Using sensitivity and specificity we can then calculate the area under the curve (AUC) of the ROC. To find the probability at which both the sensitivity and specificity are the most optimal, we made use of Youden's J statistic.

Furthermore, we looked at the negative predictive value and positive predictive value which were calculated using the following formula

$$NPV = \frac{TN}{FN + TN} \text{ and } PPV = \frac{TP}{TP + FP}$$

In our case, we must emphasize that TN means that the model accurately predicts that a patient will attain the target and TP means the model accurately predicts that the patient will not attain the target. Another metric we looked at was the number of miss-classifications, this is just the sum of FP and FN.

Finally, we also investigated the Decision Curve Analysis (DCA). First, we introduce the net benefit that is given by,

$$\text{net benefit} = \frac{TP}{N} - \frac{FP}{N}\left(\frac{p_t}{1 - p_t}\right),$$

where N is the number of patients and p_t is the threshold probability which the practitioner can choose. In the DCA curve we compare the model net benefit to two extreme strategies these are treating all the patients (where $\frac{TP}{N} = \pi$ and, $\frac{FP}{N} = 1 - \pi$ in the equation above, with $\pi = \frac{(TP+FN)}{N}$ being the prevalence (prior) of the disease) and of treating none of the patients (where $TP = 0$ and $FP = 0$ the net benefit being thus 0 at any threshold probability).

Using all this, we can see how well our models performed.

2.6. Model Calibration

We want to make use of the probabilities generated by the different models. The logistic regression model is well-calibrated. However, the RF model and Naive Bayes are prone to miss calibration [18]. The RF model gives weights to each class to classify if a person will or will not attain the target. Note that these weights should not be confused with the weights given to the model as an input. We made the ROC curve using the weights (probabilities) that the RF model generated. If we want to change the threshold, we must know if the weights used to classify each patient are reliable. In the literature [18], we find that we can calibrate the RF model using Platt scaling. This method transforms the output of a classification model into a probability distribution over classes. The same method will be used for the Naive Bayes model.

2.7. Model comparison

The models were compared by looking at the average performance of the 1000 times repeated cross-validation. The metrics we used to compare the performance were AUC and misclassification. We chose ROC because this gives us a good indicator of how the model performs over different probability thresholds. We also used the DCA curve to see which probability thresholds the models had a net benefit. The ROC & DCA gave us a combined look at how good the model performed for different thresholds. The ideal threshold was chosen where we had a sensitivity above 0.80 and specificity around 0.60 in the internal data.

2.7.1. Models on External data

The models that were built on the internal data set will be validated on the external data set. This is done using the ROC curve. The performance will then also be compared to that of the internal validation set. Furthermore, the performance of the models at the different thresholds will be compared.

3. Results

3.1. Data

3.2. Internal Data

In this section, we are going to present the data that we used to build our models.

Variables	**Target Attained (N=261)**	**Target Not Attained (N=115)**	**Overall (N=376)**	**P-value**
Age (years)				

Mean (SD)	63.9 (13.3)	56.4 (14.4)	61.6 (14.1)	**<0.001**
Median [Min, Max]	65.0 [18.0, 91.0]	61.0 [19.0, 85.0]	64.0 [18.0, 91.0]	
N	261	115	376	
Length (cm)				
Mean (SD)	173 (10.2)	176 (10.1)	174 (10.2)	0.005
Median [Min, Max]	173 [145, 198]	176 [143, 198]	175 [143, 198]	
N	261	115	376	
Weight (kg)				
Mean (SD)	81.7 (21.0)	81.1 (17.4)	81.5 (19.9)	0.997
Median [Min, Max]	80.0 [39.0, 191]	80.0 [45.0, 140]	80.0 [39.0, 191]	
N	261	115	376	
Sex				
Male	155 (59.4%)	77 (67.0%)	232 (61.7%)	0.202
Female	106 (40.6%)	38 (33.0%)	144 (38.3%)	
BMI (kg/m^2)				
Mean (SD)	27.3 (7.15)	26.1 (5.25)	26.9 (6.65)	0.131
Median [Min, Max]	26.0 [13.8, 72.8]	25.3 [17.8, 44.6]	25.8 [13.8, 72.8]	
N	261	115	376	
Sepsis				
No	126 (48.3%)	86 (74.8%)	212 (56.4%)	<0.001
Yes	135 (51.7%)	29 (25.2%)	164 (43.6%)	
SOFA score				
Mean (SD)	8.97 (4.49)	7.46 (4.01)	8.51 (4.40)	0.002
Median [Min, Max]	9.00 [0, 21.0]	7.00 [0, 17.0]	8.00 [0, 21.0]	
N	260	115	375	
Type Antibiotic				
Amoxicillin	7 (2.7%)	8 (7.0%)	15 (4.0%)	<0.001
Cefotaxim	40 (15.3%)	37 (32.2%)	77 (20.5%)	
Ceftazidim	14 (5.4%)	1 (0.9%)	15 (4.0%)	
Ceftriaxon	100 (38.3%)	12 (10.4%)	112 (29.8%)	
Cefuroxime	47 (18.0%)	25 (21.7%)	72 (19.1%)	
Flucloxacillin	6 (2.3%)	11 (9.6%)	17 (4.5%)	
Meropenem	40 (15.3%)	14 (12.2%)	54 (14.4%)	
Piperacillin/tazobactam	7 (2.7%)	7 (6.1%)	14 (3.7%)	
APACHE score				
Mean (SD)	57.1 (31.9)	43.0 (26.4)	52.7 (31.0)	<0.001
Median [Min, Max]	59.0 [3.00, 162]	33.0 [7.00, 112]	50.0 [3.00, 162]	
N	260	115	375	
WBC d0 ($\times 10^9/L$)				
Mean (SD)	14.6 (11.8)	15.0 (8.25)	14.7 (10.8)	0.177
Median [Min, Max]	13.1 [0.0500, 100]	13.5 [0.200, 54.7]	13.2 [0.0500, 100]	
N	260	113	373	
CRP d0 (mg/L)				
Mean (SD)	174 (139)	134 (124)	162 (136)	0.005
Median [Min, Max]	151 [0, 666]	119 [1.00, 483]	136 [0, 666]	
N	256	111	367	
Fluidbalance d0				
Mean (SD)	1810 (2400)	1420 (2120)	1690 (2330)	0.107
Median [Min, Max]	1390 [-3540, 11400]	873 [-2230, 9500]	1240 [-3540, 11400]	
N	260	112	372	
Serum creatinine d0 (mg/dL)				
Mean (SD)	1.52 (1.24)	0.915 (0.682)	1.34 (1.13)	<0.001
Median [Min, Max]	1.24 [0.0570, 10.3]	0.735 [0.283, 6.31]	0.995 [0.0570, 10.3]	

N	261	115	376	
Normalized Dosing				
Mean (SD)	1.21 (0.645)	1.34 (0.727)	1.25 (0.673)	0.099
Median [Min, Max]	1.00 [0.500, 6.00]	1.00 [0.750, 6.00]	1.00 [0.500, 6.00]	
N	257	115	372	
DDD				
Mean (SD)	1.21 (0.517)	1.35 (0.716)	1.25 (0.588)	0.18
Median [Min, Max]	1.00 [0.333, 6.00]	1.00 [0.667, 6.00]	1.00 [0.333, 6.00]	
N	257	115	372	
MDRD				
Mean (SD)	75.5 (159)	105 (51.1)	84.5 (136)	<0.001
Median [Min, Max]	55.0 [4.82, 2530]	102 [8.89, 278]	66.8 [4.82, 2530]	
N	261	115	376	
CKD eGFR (mL/min/1.73m^2)				
Mean (SD)	63.0 (38.8)	91.6 (29.6)	71.7 (38.5)	<0.001
Median [Min, Max]	58.1 [4.12, 373]	96.3 [8.36, 143]	71.0 [4.12, 373]	
N	261	115	376	
Adjusted CLCR				
Mean (SD)	76.4 (125)	113 (53.7)	87.7 (109)	<0.001
Median [Min, Max]	60.3 [5.96, 1940]	102 [12.6, 250]	72.9 [5.96, 1940]	
N	261	115	376	
CLCR				
Mean (SD)	86.0 (154)	122 (57.1)	96.9 (133)	<0.001
Median [Min, Max]	65.0 [5.37, 2420]	109 [14.5, 304]	78.1 [5.37, 2420]	
N	261	115	376	
CKD EPI eGFR 2021				
Mean (SD)	65.2 (36.2)	93.7 (28.3)	73.9 (36.4)	<0.001
Median [Min, Max]	61.3 [4.53, 283]	101 [9.04, 140]	74.8 [4.53, 283]	
N	261	115	376	
Urea d0 (mmol/L)				
Mean (SD)	12.8 (8.01)	8.45 (6.00)	11.5 (7.71)	<0.001
Median [Min, Max]	10.7 [1.70, 44.5]	7.20 [2.80, 48.4]	9.00 [1.70, 48.4]	
N	253	112	365	
Frequency of Dosing				
Mean (SD)	2.43 (1.36)	3.42 (1.39)	2.73 (1.44)	<0.001
Median [Min, Max]	3.00 [1.00, 6.00]	3.00 [1.00, 6.00]	3.00 [1.00, 6.00]	
N	260	115	375	
Dose of Antibiotic				
Mean (SD)	1640 (650)	1460 (853)	1580 (722)	<0.001
Median [Min, Max]	1500 [500, 4500]	1000 [1000, 4500]	1500 [500, 4500]	
N	261	115	376	
Albumin (g/L)				
Mean (SD)	27.2 (6.97)	29.8 (7.11)	28.0 (7.11)	0.003
Median [Min, Max]	27.0 [11.0, 47.0]	30.0 [14.0, 46.0]	28.0 [11.0, 47.0]	
N	239	107	346	
Cardiac Surgery				
No	253 (96.9%)	114 (99.1%)	367 (97.6%)	0.556
Yes	3 (1.1%)	0 (0%)	3 (0.8%)	
Trauma				
No	243 (93.1%)	101 (87.8%)	344 (91.5%)	0.048
Yes	13 (5.0%)	13 (11.3%)	26 (6.9%)	
Days in ICU before d0				
Mean (SD)	4.85 (8.05)	4.81 (6.27)	4.84 (7.54)	0.609

Median [Min, Max]	2.00 [0, 76.0]	2.00 [1.00, 35.0]	2.00 [0, 76.0]	
N	261	115	376	

Table 1: All the variables are categorized into target attainment and target non-attainment.

In Table 1 it can be seen that a lot of variables are available. To accomplish our main objective, we will need to find which of these variables are the best for predicting target attainment. Note that we are more interested in people who do not attain the target because these people will get extra care. This means that the model that we build should have a high accuracy in predicting people who did not attain the target.

To see how the data is distributed, we made use of histograms. Below we have put some of these histograms.

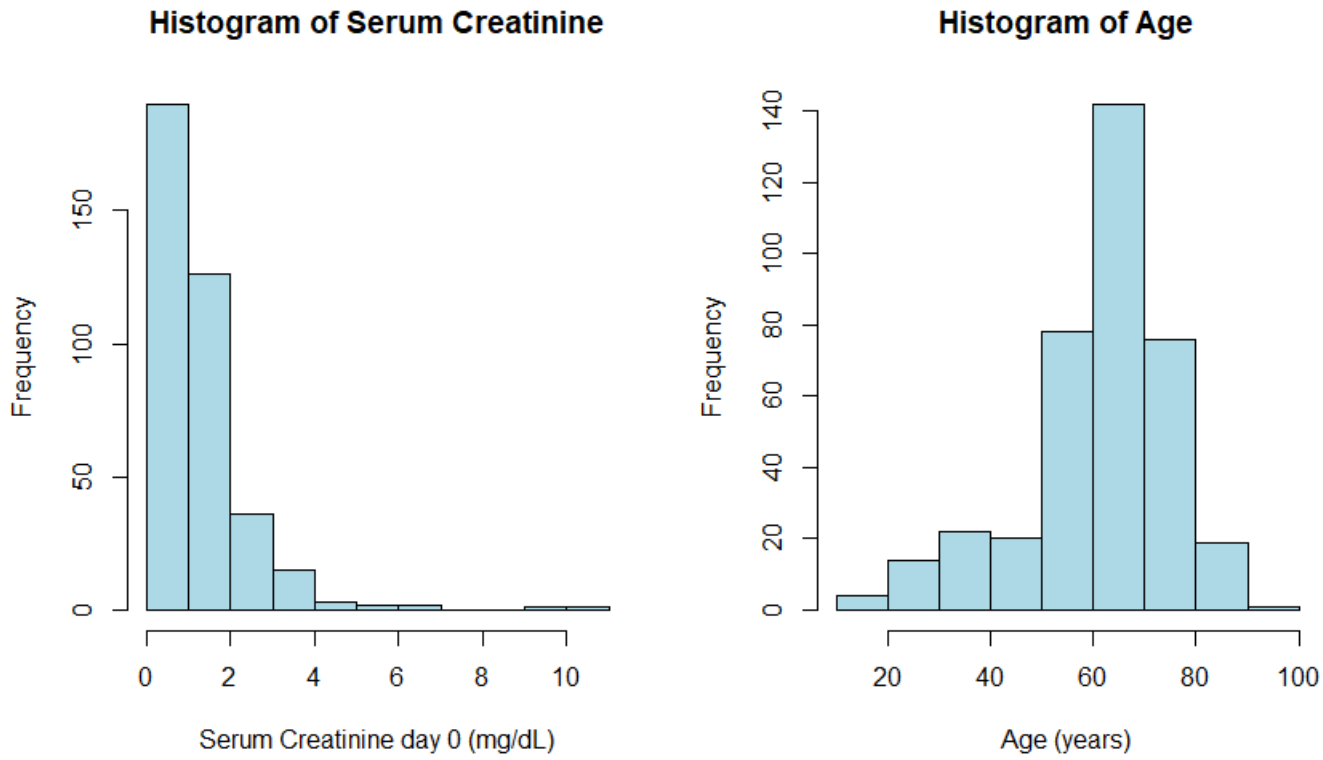


Figure 3: Histograms of Serum Creatinine (left) and Age (right) on the internal data.

3.3. External data

In this section, the external data will be presented. Only the variables which were used to make the models were present in the external validation data. This can be seen in the table below.

	Target attained (N=78)	**Target not Attained (N=72)**	**Overall (N=150)**	**P-value**
Serum creatinine d0 (mg/dL)				
Mean (SD)	1.17 (0.757)	0.680 (0.269)	0.935 (0.625)	**<0.001**
Median [Min, Max]	0.940 [0.0600, 3.56]	0.680 [0.0700, 1.61]	0.770 [0.0600, 3.56]	
N	78	72	150	
Type Antibiotic				
Ceftazidim	4 (5.1%)	2 (2.8%)	6 (4.0%)	**0.002**
Ceftriaxon	17 (21.8%)	4 (5.6%)	21 (14.0%)	
Meropenem	39 (50.0%)	38 (52.8%)	77 (51.3%)	
Piperacillin/tazobactam	18 (23.1%)	19 (26.4%)	37 (24.7%)	
Amoxicillin	0 (0%)	9 (12.5%)	9 (6.0%)	
Gender				
Female	31 (39.7%)	22 (30.6%)	53 (35.3%)	**0.315**
Male	47 (60.3%)	50 (69.4%)	97 (64.7%)	
Age (years)				
Mean (SD)	63.9 (14.1)	58.0 (13.8)	61.1 (14.2)	**0.013**
Median [Min, Max]	64.0 [30.6, 93.0]	59.0 [23.0, 88.0]	62.0 [23.0, 93.0]	
N	78	72	150	

Table 2: External data categorized into target attainment and target non-attainment.

To see how the external data is distributed, compared to the internal data we made histograms of the age and serum creatinine.

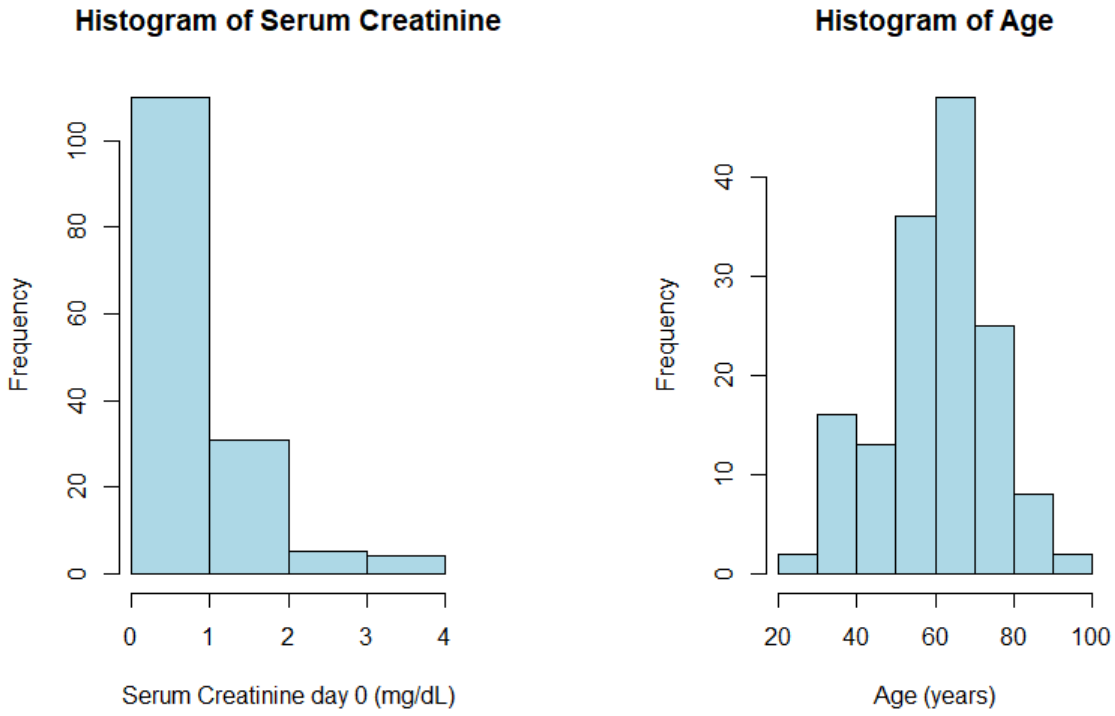


Figure 4: Histograms of Serum Creatinine (left) and Age (right) on the external data.

The histograms of age do not differ significantly in the Figures 3 and 4, as validated by the Wilcoxon test, which yielded a p-value of 0.36. However, looking at the histograms of Serum Creatinine reveals a difference, which is further verified by the Wilcoxon test with a p-value of $9.71 \cdot 10^{-6}$.

3.4. RF Model

The following section will show how our random forest model performed on our data. The first thing we did was make a random train and validation set. After that, we trained our model on this train set with the predictors we found using the Boruta algorithm and tested it on the validation set. The ROC curve was then used to determine how well the model performed.

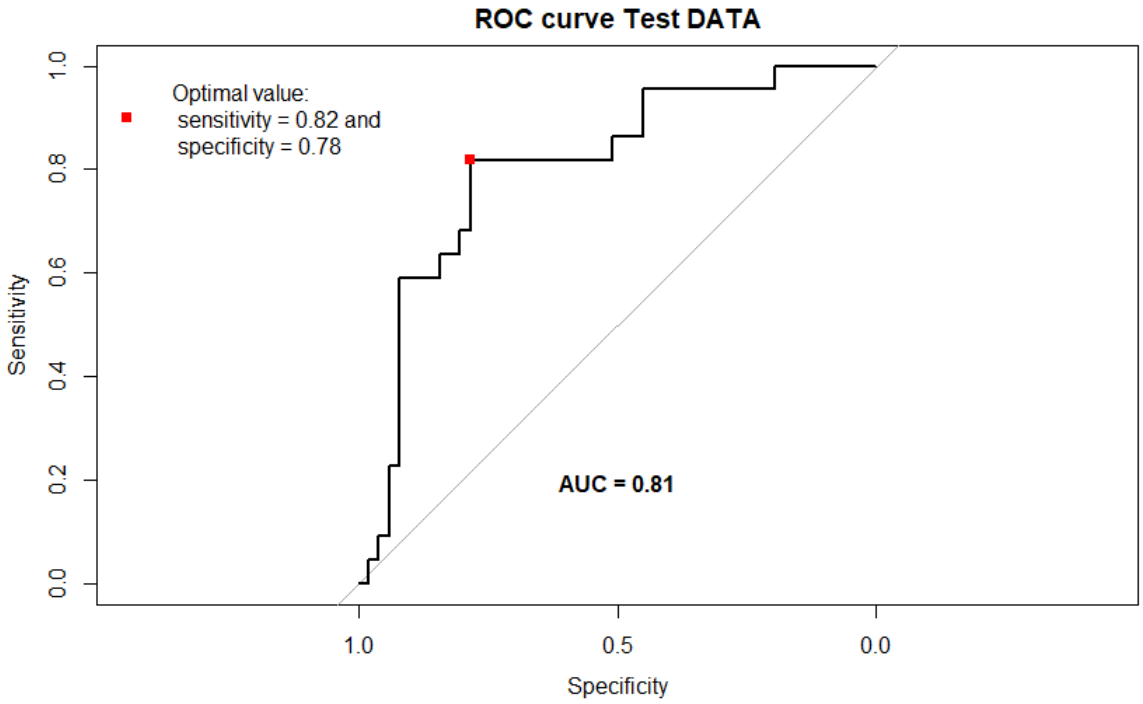


Figure 5: ROC curve for the Random Forest model.

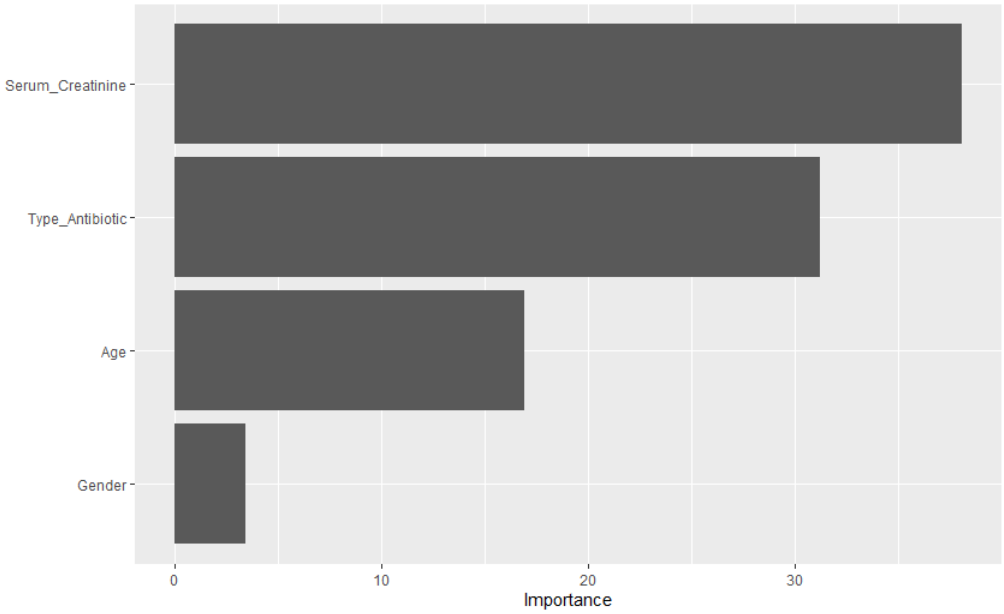


Figure 6: Variable importance of the Random Forest Algorithm.

In Figure 5, we have the ROC curve with a sensitivity of 0.82, specificity of 0.78 and Area under the curve (AUC) of 0.81. This model performs reasonably well on our data. In Figure 6, we see the importance plot of the random forest Algorithm. It can be seen that serum creatinine is the most crucial predictor variable. This makes sense because it is a reasonably good indicator of kidney function. The type of antibiotic, sex and age are also essential, as seen in the Figure.

As mentioned in the method section, we looked at different train and validation sets using the stratification process. We calculated the ROC for each of these different train and validation sets, after which we took the average of the metrics. This gave us the average cut-off point (threshold) for the most optimal sensitivity and specificity. This was at a probability of 37%. Furthermore, the average AUC was 0.83, with an average sensitivity of 0.76 and an average specificity of 0.84. The average sensitivity and specificity are taken for their best probability threshold (Youden J statistic) and not the average probability threshold (0.37).

To see whether these different thresholds have a net benefit, we look at the DCA curve of this model.

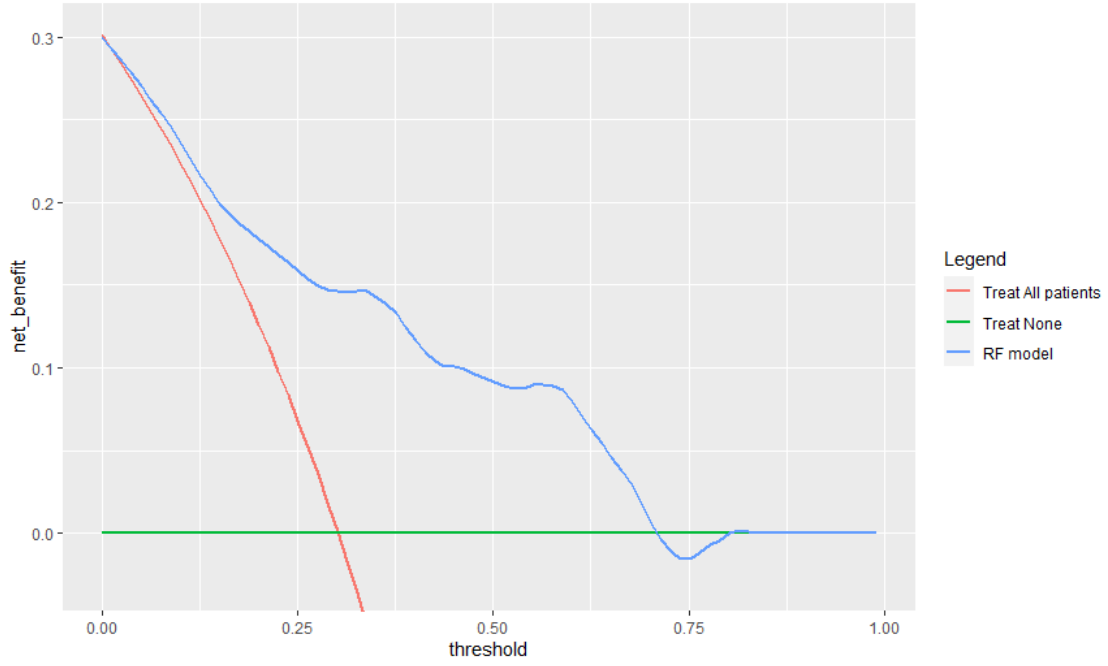


Figure 7: DCA curve for the Random Forest model.

The model has a net benefit until a threshold of 0.7, as seen in Figure 7. This indicates that the model is clinically relevant until a threshold of 0.7. We wanted to use different thresholds to see how the model did for all these different thresholds. This was done by looking at how all the 1000 models performed at these different thresholds and taking the average. This can be seen in Table 3 below.

Threshold	Specificity	Sensitivity	NPV	PPV	Misclassification
0.50	0.90	0.61	0.84	0.72	0.19
0.45	0.87	0.65	0.85	0.69	0.19
0.40	0.85	0.68	0.86	0.66	0.20
0.37	**0.83**	**0.71**	**0.87**	**0.64**	**0.21**
0.35	0.82	0.72	0.87	0.63	0.21
0.30	**0.78**	**0.76**	**0.88**	**0.60**	**0.23**
0.25	0.73	0.79	0.89	0.56	0.25
0.20	**0.65**	**0.83**	**0.90**	**0.51**	**0.29**
0.15	0.54	0.87	0.91	0.46	0.36
0.10	0.38	0.94	0.93	0.40	0.45
0.05	0.17	0.99	0.97	0.34	0.58

Table 3: Table with different thresholds and the average performance at the given threshold.

Note that we are interested in patients who do not attain the target because those are the patients that will receive extra care. This means that in the ROC curve, we want a high sensitivity but still a reasonable specificity. If we look at the model's performance at the average cut off point of 0.37, we see that the model has a sensitivity of 0.71 and specificity of 0.83. Suppose we took the cut-off at 0.3, which is the prevalence of target non-attainment in our data. We find that the model has a sensitivity of 0.76 and specificity of 0.78. So if we want a reasonably high sensitivity (above 0.8), we can take a probability threshold between 0.30 and 0.20. At these thresholds, the sensitivity is above 0.80, and the specificity around 0.6. The ideal threshold that we can choose for the RF model is 0.20. At this threshold the specificity is 0.65 and the sensitivity 0.83.

3.4.1. Parameter Tuning

For our random forest model, we looked at five parameters [19]. The first one we looked at was the number of random variables selected from the predictor variables (mtries) . The second one was the min rows, which means it controls minimum number of observations in a terminal (leaf) node. The third parameter we tuned was the sample rate, this is the size of the sample data drawn for training each tree. These three parameters were tuned using 10-fold CV, this resulted in mtries equal to 1, min rows equal to 0.3 and sample rate equal to 0.75.

The fourth parameter we looked at was the weight of the classes. Because we have unbalanced classes, meaning we have more people who have attained the target than have not. To offset this imbalance, we used weights. The weight we used is given by the formula below:

$$W_N = \frac{1}{2T_N} \qquad \text{and} \qquad W_Y = \frac{1}{2T_Y}, \tag{6}$$

where

W_N is the weights for the class of Target not attained,
W_Y is the weights for the class of Target attained,
T_Y is the number of people who did attain the target and
T_N is the number of people who did not attain the target.

These weights are chosen in such a way that they all sum up to 1. To use these weights in the h2o RF model, we had to multiply by 5.

The last parameter we looked at was the number of trees. The higher the number of trees, the better the results become, but the performance after a certain amount of trees stays somewhat constant. We combine this parameter with early stopping parameter. So what happens is that it calculates errors for different number of trees until the error does not change for a certain amount of rounds. The final number of trees is then equal to that. The maximum number of trees we give the model is 2500.

3.4.2. External Validation of the RF model

The RF model that was made was externally validated on the external data set. The performance of the model was estimated using the ROC curve. This can be seen in the figure below.

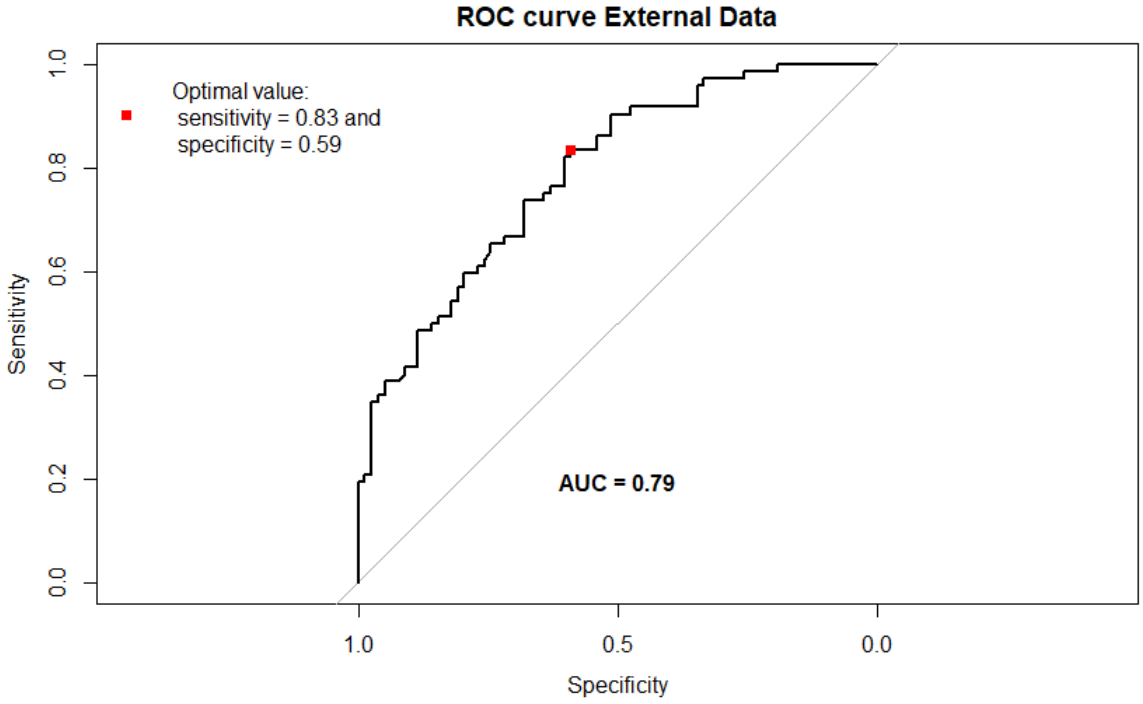


Figure 8: ROC curve for the Random Forest model on the external data.

In Figure 8 it can be seen that the model has an AUC of 0.79, which indicates that the model performs reasonable on the external data. To get a better insight into the model, the performance of the model at different thresholds is looked at.

Thresholds	Specificity	Sensitivity	NPV	PPV	Misclassification
0.50	0.95	0.36	0.62	0.87	0.33
0.45	0.95	0.36	0.62	0.87	0.33
0.40	0.91	0.42	0.63	0.81	0.33
0.37	0.86	0.50	0.65	0.77	0.31
0.35	0.81	0.54	0.66	0.72	0.32
0.30	0.72	0.65	0.69	0.68	0.31
0.25	0.60	0.76	0.73	0.64	0.32
0.235	**0.59**	**0.83**	**0.79**	**0.65**	**0.29**
0.20	**0.50**	**0.90**	**0.85**	**0.62**	**0.31**
0.15	0.35	0.94	0.87	0.57	0.37
0.10	0.22	0.99	0.94	0.54	0.41
0.05	0.05	1.00	1.00	0.49	0.49

Table 4: The RF model on the external data with different thresholds.

In Table 4 it can be seen that the model has a reasonable specificity and high sensitivity, a threshold between 0.20 and 0.25. This is backed up by our internal validation, which showed that a threshold of 0.20 to 0.30 produced similar outcomes. This means if we take a threshold of 0.20 we have the best threshold for internal and external validation. At the threshold of 0.20, the specificity is a bit worse, in the external data, but it is still reasonable. This indicates that overall this threshold will do well, on different data sets.

3.5. Logistic regression

The logistic regression model was built using the same predictor variables as the RF model. The same train and validation set was used as in the RF model. For this model, we also looked at the ROC curve.

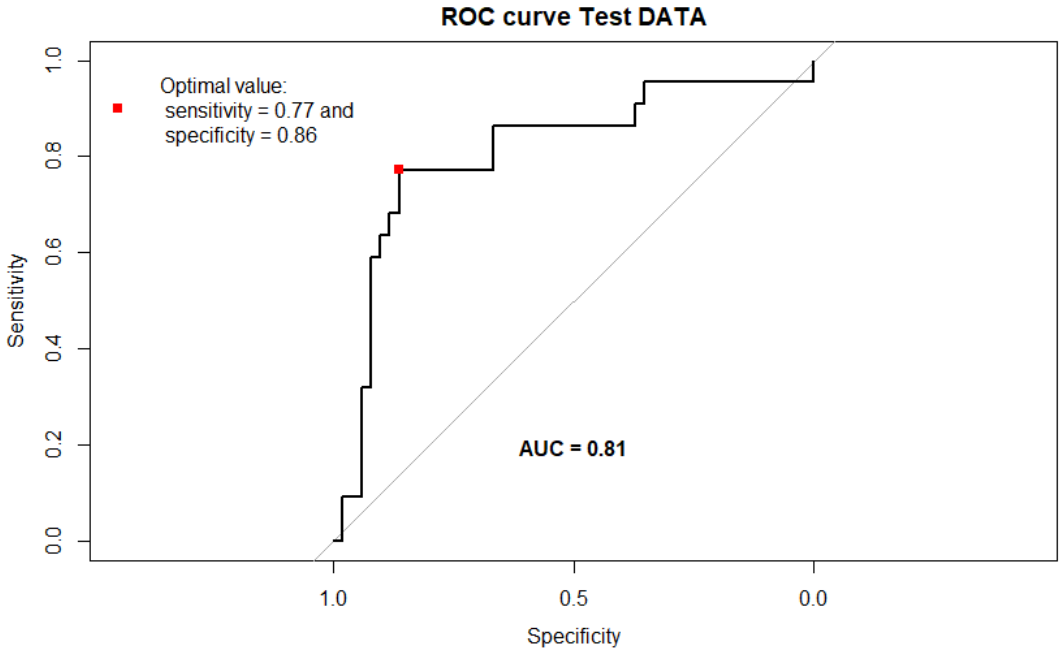


Figure 9: ROC curve for the logistic regression model.

If we look at Figure 9 then by the Youden statistic we have a sensitivity of 0.77, specificity of 0.86. The AUC is 0.81 which is also reasonable.

The same stratification process was followed as mentioned in subsection 2.5.2. In Table 5 we can see our logistic model with the odds ratios (OR), confidence intervals (CI) and p-values.

Predictors	**Odds Ratios**	**Confidence Interval**	**P-value**
(Intercept)	6.97	1.66 – 30.23	0.008
Serum Creatinine day 0	0.24	0.12 – 0.42	<0.001
Age	0.96	0.94 – 0.98	<0.001
Sex: Female	0.45	0.24 – 0.85	0.015
Amoxicillin	12.58	2.44 – 74.70	0.003
Cefotaxim	13.84	5.49 – 38.35	<0.001
Ceftazidim	0.47	0.02 – 3.22	0.510
Cefuroxime	6.25	2.47 – 16.97	<0.001
Flucloxacillin	10.86	2.84 – 46.90	0.001
Meropenem	2.64	0.92 – 7.69	0.072
Piperacillin/tazobactam	13.20	3.04 – 60.38	0.001
Observations	303		

Table 5: Logistic regression model on the train set.

We must first note that the reference category for type of antibiotic was Ceftriaxone, and the reference category for sex was male. These two are represented in the intercept of the model. These were chosen as the reference category because they appeared most frequently in their respective predictor variables. The ORs can be interpreted in the following manner. We find that for serum creatinine, the OR is 0.24. This means that for each 1 mg/dL increase of the serum creatinine on day 0, the individual becomes 0.24 times less likely to achieve target non-attainment. The OR for Amoxicillin is 12.58, which implies that an individual receiving Amoxicillin has 12.58 times higher odds of target non-attainment than someone receiving Ceftriaxone. The other ORs can be interpreted in the same way.

The DCA curve was made for this model to see at which threshold this model had a net benefit. The DCA curve showed a net benefit for thresholds below 0.66. This means that the model has clinical relevance for

thresholds below 0.66.

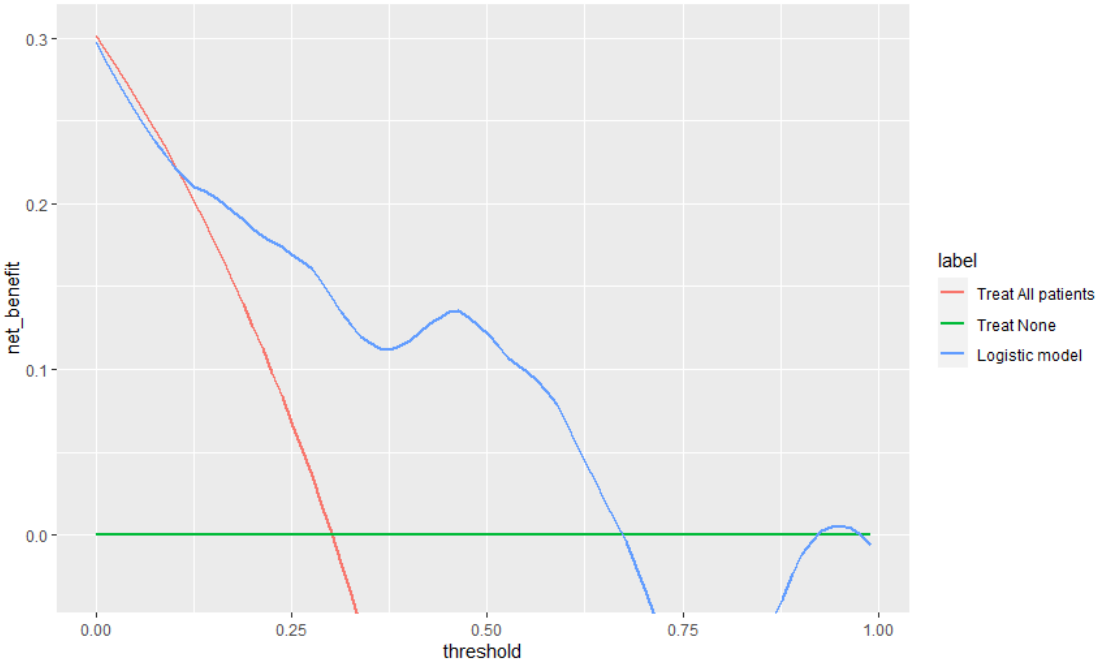


Figure 10: DCA curve for the logistic regression model.

After performing 1000 times repeated CV, we found the following average metrics. The average AUC was equal to 0.84. For each of the 1000 iterations, we also calculated the Youden statistic. So the average sensitivity we get from this is 0.77, and the average specificity is 0.83. The average probability threshold that we got was a probability of 39 %. The average NPV was 0.90, and the average misclassification was 0.19.

We wanted to make use of different thresholds, to see how the model did for all these different thresholds we just looked at how all the 1000 models performed at these different thresholds and took the average. This can be seen in Table 6 below.

Threshold	**Specificity**	**Sensitivity**	**NPV**	**PPV**	**Misclassification**
0.50	0.90	0.60	0.84	0.72	0.19
0.45	0.85	0.65	0.85	0.67	020
0.40	0.83	0.71	0.85	0.64	0.21
0.39	**0.82**	**0.72**	**0.87**	**0.63**	**0.21**
0.35	0.77	0.75	0.88	0.59	0.24
0.30	**0.71**	**0.79**	**0.89**	**0.55**	**0.26**
0.25	0.66	0.83	0.90	0.51	0.29
0.21	**0.60**	**0.85**	**0.91**	**0.48**	**0.32**
0.20	0.59	0.86	0.91	0.48	0.33
0.15	0.52	0.89	0.92	0.45	0.37
0.10	0.37	0.95	0.95	0.39	0.46
0.05	0.21	0.99	0.98	0.35	0.56

Table 6: Table with all different thresholds and the average performance.

If we take into account the probability threshold of 0.39, which is attained by taking the average over all Youden statistics. It can be seen that the average sensitivity is 0.82, and the average specificity equals 0.72. While if we would take the prevalence, which is 0.30, as our cut-off, we find that the average sensitivity is 0.79 and the average specificity equal to 0.71. We must note that we are interested in people who do not attain the target. This means that we are still interested in a high sensitivity but with reasonable specificity. This implies

that for this model to achieve our purpose, a probability threshold of 0.30 to 0.20 would be an excellent choice, because the sensitivity at these thresholds are above 0.8 and specificity around 0.60. The threshold of 0.21 is the ideal choice for the LR model, at this threshold, we have a sensitivity of 0.85 and specificity of 0.60.

3.5.1. External Validation for logistic regression model

In this section, we will externally validate our logistic regression model. This will be accomplished in the same manner as the RF model.

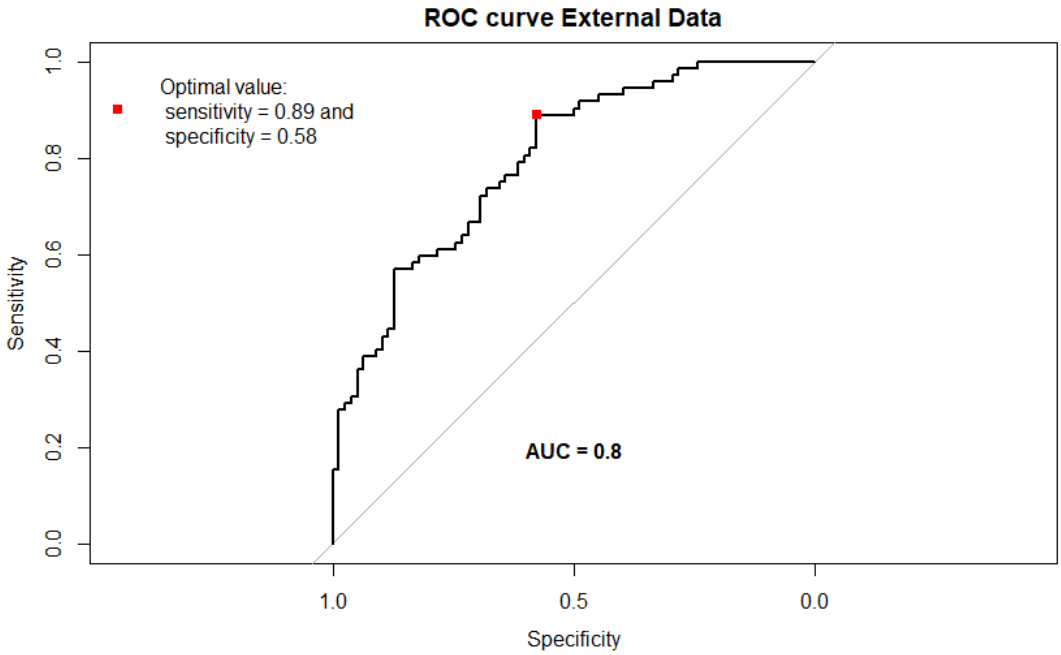


Figure 11: ROC curve of the logistic regression model on the external data.

Figure 11 shows that the logistic regression model does a good job at predicting target attainment, with an AUC of 0.80. Table 7 shows the model performance at different thresholds. The model has a reasonable specificity, with a high sensitivity at thresholds between 0.20 until 0.275. This again seen in the internal validation, where we have the same results for thresholds between 0.20 and 0.30. This means that if we take a threshold of 0.21 we have our ideal threshold for the logistic regression model in the internal and external data set.

Thresholds	Specificity	Sensitivity	NPV	PPV	Misclassification
0.50	0.90	0.42	0.62	0.79	0.33
0.45	0.87	0.54	0.67	0.80	0.29
0.40	0.83	0.58	0.68	0.76	0.29
0.39	0.79	0.60	0.68	0.73	0.30
0.35	0.73	0.64	0.69	0.69	0.31
0.30	0.69	0.72	0.73	0.68	0.29
0.275	0.62	0.76	0.74	0.65	0.31
0.25	0.58	0.85	0.80	0.65	0.29
0.247	**0.58**	**0.89**	**0.85**	**0.66**	**0.27**
0.21	**0.50**	**0.90**	**0.85**	**0.62**	**0.31**
0.20	0.49	0.92	0.86	0.62	0.31
0.15	0.38	0.94	0.88	0.59	0.35
0.10	0.28	0.97	0.92	0.56	0.39
0.05	0.17	1.00	1.00	0.53	0.43

Table 7: The logistic regression model on the external data with different thresholds.

3.6. Naive Bayes Model

In this section, we will look at the performance of the Naive Bayes model on our data. First, we are going to look how the model performed on the same validation and train set as for the RF model using the ROC curve. Again, the same predictor variables were used as for the RF model.

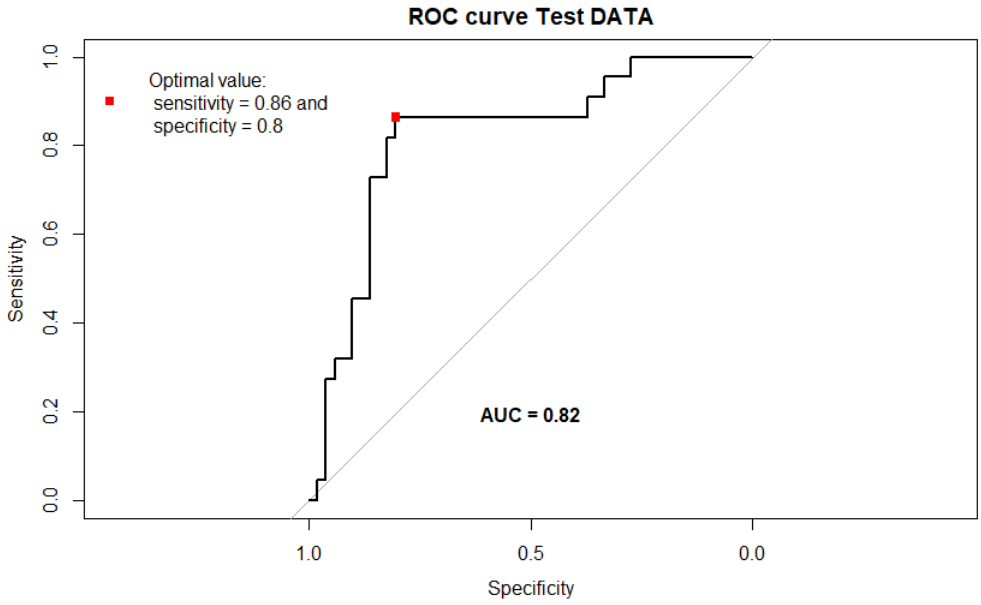


Figure 12: ROC curve of the Naive Bayes model.

Figure 12 shows the ROC curve with sensitivity of 0.86, specificity of 0.8, and AUC of 0.82. The AUC indicates that the model performs reasonable well on our data.

To make use of the different thresholds, we want to know if these thresholds have a net benefit. This will be seen in the DCA curve below.

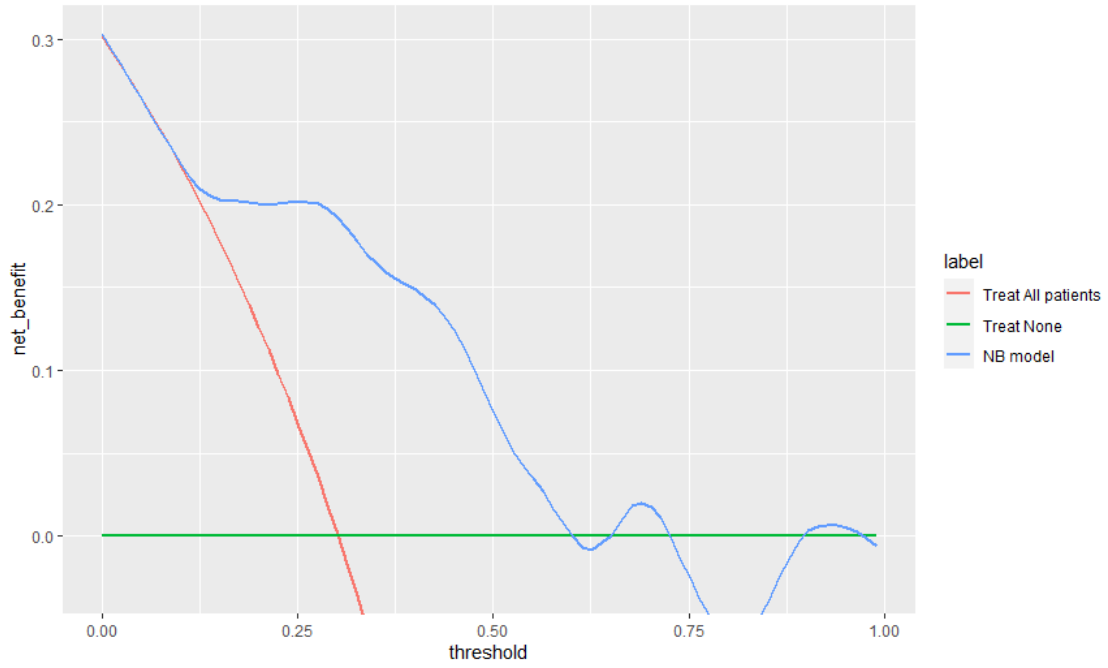


Figure 13: DCA curve of the Naive Bayes model.

In Figure 13 it can be seen that the model has a net benefit until a probability threshold of 0.59. This means that we will not take our probability threshold above this level.

Finally, we used the stratification procedure to look at different train and validation sets. We estimated the ROC for each of these, using various train and validation sets, and then averaged the results. This provided us the average threshold (cut-off point) of 33 %. Furthermore, the average AUC was around 0.83, with average sensitivity and specificity of 0.80 and 0.81, respectively. It's worth noting that the average sensitivity and specificity are taken for the best probability threshold for each model (Youden J statistic) rather than the average probability threshold (0.33).

We wanted to utilize various thresholds, so we just averaged the metrics of all 1000 models at the given threshold. Table 8 below illustrates this.

Thresholds	**Specificity**	**Sensitivity**	**NPV**	**PPV**	**Misclassification**
0.50	0.89	0.55	0.82	0.68	0.21
0.45	0.87	0.62	0.84	0.67	0.21
0.40	0.85	0.69	0.86	0.66	0.20
0.35	0.81	0.75	0.88	0.64	0.21
0.33	**0.80**	**0.77**	**0.89**	**0.62**	**0.21**
0.30	**0.79**	**0.78**	**0.89**	**0.60**	**0.22**
0.25	0.73	0.81	0.90	0.57	0.25
0.20	0.67	0.83	0.90	0.53	0.28
0.17	**0.62**	**0.85**	**0.90**	**0.50**	**0.31**
0.15	0.57	0.87	0.91	0.47	0.34
0.10	0.39	0.93	0.93	0.41	0.45
0.05	0.11	0.99	0.99	0.33	0.70

Table 8: Naive Bayes model average performance at different thresholds.

We can see that the average sensitivity is 0.77 and the average specificity is 0.80 if we use the probability threshold of 0.33, which is obtained by averaging all Youden statistics. While using the prevalence (0.30) as our cut-off, we discover that the average sensitivity is 0.79 and the average specificity is 0.78. We must emphasize that we are mostly interested in persons who have not attained the target. This means that we want a threshold with a sensitivity above 0.80 while maintaining a specificity around 0.60. This means that a probability threshold of 0.30 to 0.15 would be a great choice for this model to achieve our purpose. The optimal threshold for the NB model is 0.17. The sensitivity at this threshold is 0.85 while the specificity is around 0.62.

3.6.1. External validation of the NB model

In this section, the Naive Bayes model will be validated on the external data. First we will show the ROC curve of the NB model, then we will give the performance of this model at different thresholds.

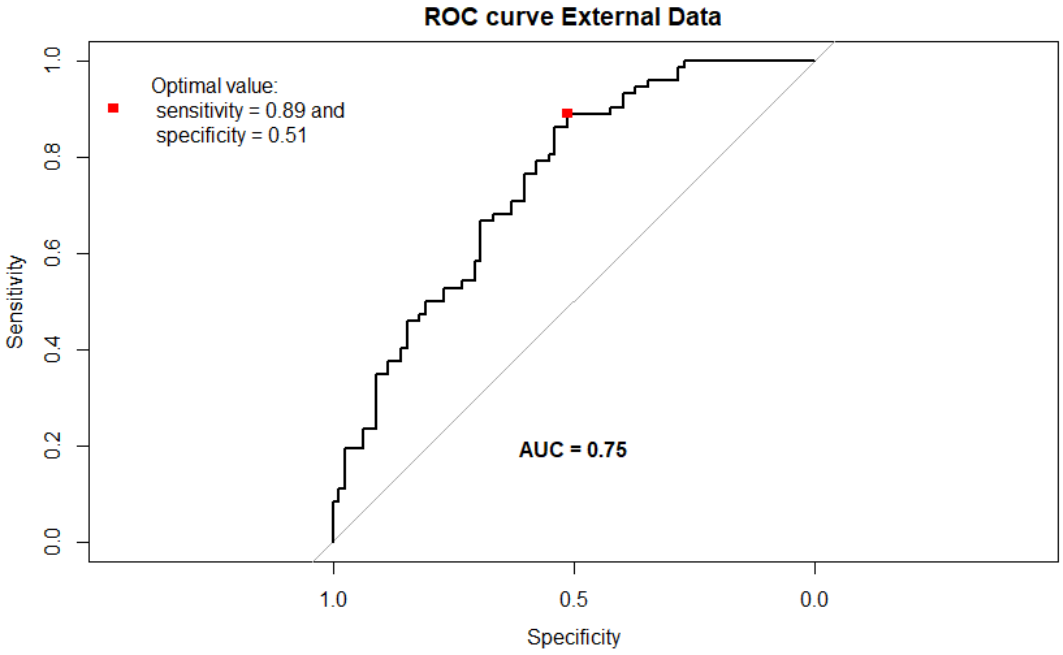


Figure 14: ROC curve for the logistic regression model on the external data.

With an AUC of 0.75, the NB model does a good job of predicting target attainment, as shown in Figure 14. The model performance at various thresholds is shown in Table 9. The model has a high sensitivity at thresholds between 0.17 and 0.235, with a reasonable specificity. The internal validation supports this, with the same findings in the internal validation for thresholds between 0.15 and 0.30. This suggests that a threshold of 0.17 is the best for the NB model. At this threshold, the model performs well in the external as the internal data set.

Thresholds	Specificity	Sensitivity	NPV	PPV	Misclassification
0.500	0.86	0.38	0.60	0.71	0.37
0.450	0.85	0.46	0.63	0.73	0.34
0.400	0.82	0.47	0.63	0.71	0.35
0.350	0.74	0.53	0.63	0.66	0.36
0.330	0.71	0.56	0.63	0.63	0.37
0.300	0.68	0.67	0.69	0.66	0.33
0.250	0.62	0.71	0.70	0.63	0.34
0.235	0.60	0.75	0.72	0.64	0.33
0.200	0.56	0.79	0.75	0.63	0.33
0.173	**0.51**	**0.89**	**0.83**	**0.63**	**0.31**
0.17	**0.47**	**0.89**	**0.82**	**0.61**	**0.33**
0.150	0.41	0.90	0.82	0.59	0.35
0.100	0.29	0.96	0.88	0.56	0.39
0.050	0.00	1.00	1.00	0.48	0.52

Table 9: The NB model on the external data with different thresholds.

3.7. Comparison of the three models

In this section, we will compare our models internally and externally.

3.7.1. Internal Validation of the three models

First, we are going to look at the combined ROC plot of the three models. This is just the combination of Figure 5, 9 and 12.

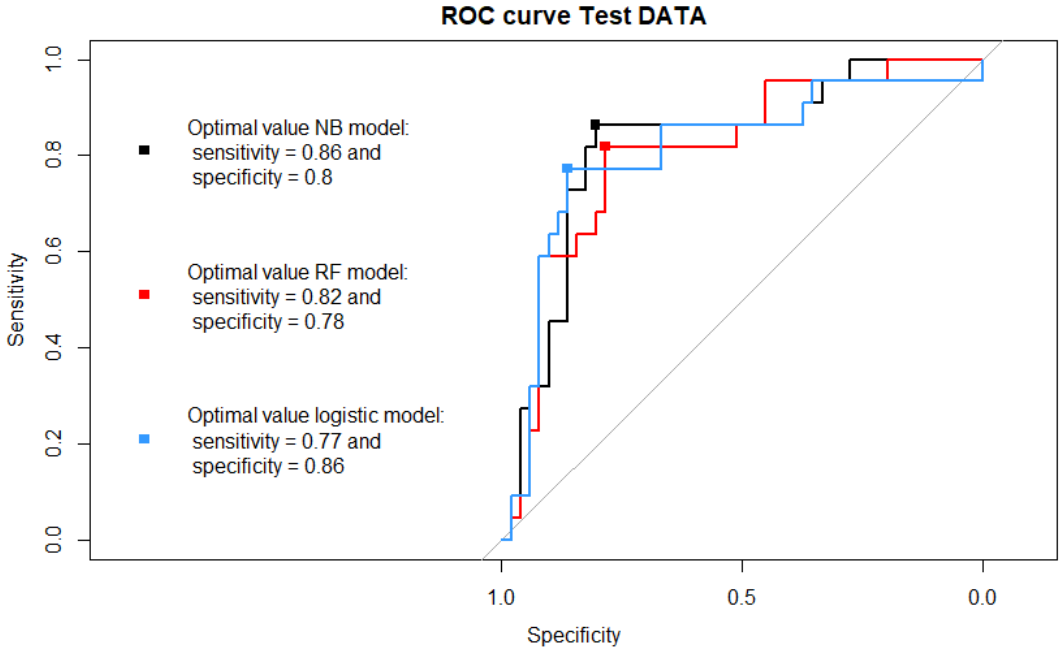


Figure 15: ROC curve of all three models.

If we look at this plot, we see that all three models do a reasonable job. All 3 models performed well on the data, with an AUC between 0.81 and 0.82. But the other two models are not very far off the NB model. This is the performance of the three models on one random train and validation set.

To see how the models performed overall, we looked at the average performance of the models. This can be seen in the table below.

Model	Specificity	Sensitivity	NPV	PPV	Misclassification	AUC
NB model	0.80	0.81	0.91	0.66	0.19	0.83
RF model	0.84	0.76	0.89	0.68	0.19	0.83
LR model	0.83	0.77	0.90	0.68	0.19	0.84

Table 10: Average performance for all the models. The averages are taken at the best probability threshold for each model (Youden J statistic), meaning that for each CV fold the Youden J statistic is calculated. Then the average over these 1000 iterations is taken.

If we look at Table 10 we see that the Logistic regression model has the best AUC, which indicates that it would work the best on our data. The other two models do not differ a lot, as they have an AUC of 0.83, which is also reasonable. If we would only look at the sensitivity, then the NB model would be the best choice as it has the highest sensitivity. The RF model and logistic model do not differ a lot in terms of sensitivity and specificity. So there is no clear best model, as the differences are minimal between the 3 models.

Lastly, we also compared the models at their average thresholds. This is seen in Table 11.

Model	Specificity	Sensitivity	NPV	PPV	Misclassification	AUC	threshold
NB model	0.80	0.77	0.89	0.62	0.21	0.83	0.33
RF model	0.83	0.71	0.87	0.64	0.21	0.83	0.37
LR model	0.82	0.72	0.87	0.63	0.21	0.84	0.39

Table 11: Average performance of the models at their average thresholds.

If we look at this, we find that the NB model overall performed the best at its average threshold. Again, if we wanted the best sensitivity, we find that the NB model had the highest sensitivity.

Finally, we compare the three models at the threshold of 0.30, this is the prevalence of target non attainment in our data.

Model	Specificity	Sensitivity	NPV	PPV	Misclassification	AUC	threshold
NB model	0.79	0.78	0.89	0.60	0.22	0.83	0.30
RF model	0.78	0.76	0.88	0.60	0.23	0.83	0.30
LR model	0.71	0.79	0.89	0.55	0.26	0.84	0.30

Table 12: Average performance of the models at a threshold of 0.3 (prevalence).

In Table 12 it can be seen that the logistic regression model has the highest sensitivity, while the NB model has the highest specificity. All the three models perform comparably at this threshold.

From these comparisons, it can be said that all three models have their benefits. However, we should externally validate all the models to know which model performs the best. The metrics we will find will give us a better indication of which model will perform the best. Furthermore, it can be seen that all three models have an AUC above 0.8, which indicates that the models will perform reasonably well [8]. Because each model performs differently at various thresholds, the threshold we will set for each one will be different. In the table below, we can see the performance of the three models at their respective ideal threshold.

Model	Specificity	Sensitivity	NPV	PPV	Misclassification	AUC	threshold
NB model	0.62	0.85	0.90	0.50	0.31	0.83	0.17
RF model	0.65	0.83	0.90	0.51	0.29	0.83	0.20
LR model	0.60	0.85	0.91	0.48	0.32	0.84	0.21

Table 13: The average performance of the models at their ideal threshold.

In Table 13 it can be seen that the three model perform similarly at their respective ideal threshold. The NB model has the highest sensitivity, while the RF model has the highest specificity.

3.7.2. External Validation of the three models

First, we are going to show how the different models performed on the external validation set. Then we will show their performance on the external validation set for different thresholds. The same models for which we built the ROC and DCA curves in the previous sections will be used.

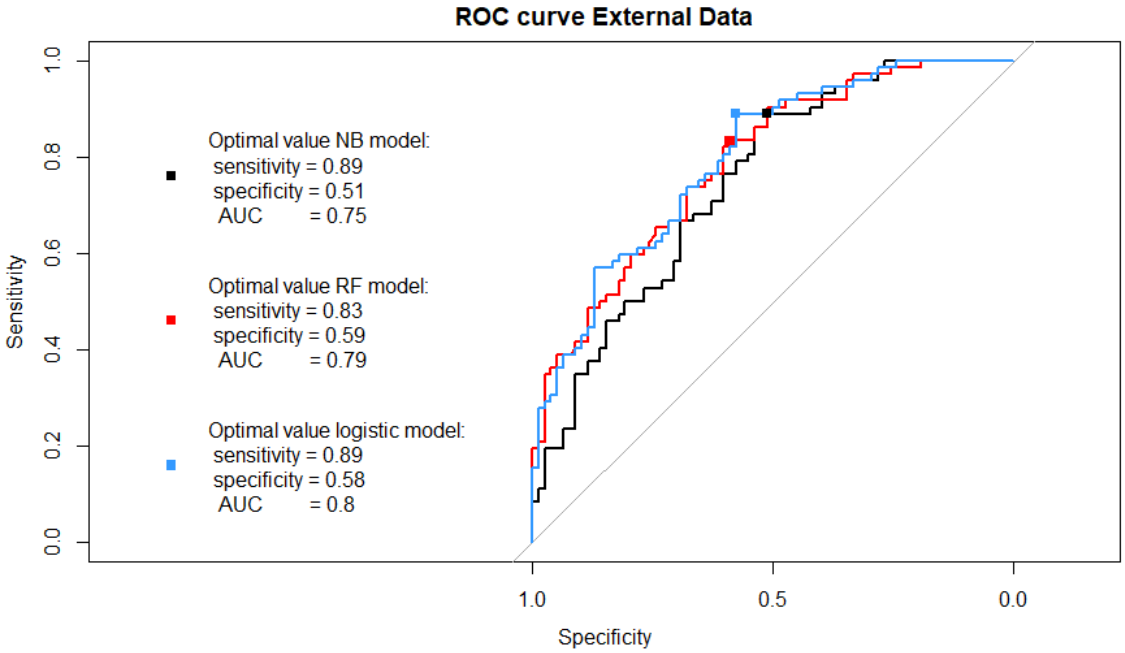


Figure 16: ROC curve of all three models on the external data.

In Figure 16, it can be seen that the logistic regression model and the RF model both perform reasonably well with an AUC of 0.80 and 0.79 respectively. The Naive Bayes model still performs reasonably, but still worse than the other two models, with an AUC of 0.75.

If we look at the Youden statistic's threshold for each of the models, we find the following table.

Model	Specificity	Sensitivity	NPV	PPV	Misclassification	AUC	threshold
NB model	0.51	0.89	0.83	0.63	0.31	0.75	0.173
RF model	0.59	0.83	0.79	0.65	0.29	0.79	0.235
LR model	0.58	0.89	0.85	0.66	0.27	0.80	0.247

Table 14: The performance of the models on the external data at their Youden statistic thresholds.

The Table shows that the logistic regression model has the highest AUC, while the best specificity is for the RF model. If we take the ideal threshold that was chosen in the internal validation and look at the performance of the three models, we find the following table.

Model	Specificity	Sensitivity	NPV	PPV	Misclassification	AUC	threshold
NB model	0.47	0.89	0.82	0.61	0.33	0.75	0.17
RF model	0.50	0.90	0.85	0.62	0.31	0.79	0.20
LR model	0.50	0.90	0.85	0.62	0.31	0.80	0.21

Table 15: The performance of the models on the external data at the ideal threshold chosen in the internal validation.

The logistic regression and RF models are the best choices if we pick a model because they have reasonably high sensitivity (above 0.8) and perform reasonably well on the external data (AUC of 0.80 and 0.79 respectively).

3.7.3. Models on the Internal & External data at their ideal threshold

In this section, we will show the models on the external and internal data at their ideal thresholds.

Internal Data								
	Model	**Specificity**	**Sensitivity**	**NPV**	**PPV**	**Misclassification**	**AUC**	**threshold**
	NB model	0.62	0.85	0.90	0.50	0.31	0.83	0.17
	RF model	0.65	0.83	0.90	0.51	0.29	0.83	0.20
	LR model	0.60	0.85	0.91	0.48	0.32	0.84	0.21
External Data								
	Model	**Specificity**	**Sensitivity**	**NPV**	**PPV**	**Misclassification**	**AUC**	**threshold**
	NB model	0.47	0.89	0.82	0.61	0.33	0.75	0.17
	RF model	0.50	0.90	0.85	0.62	0.31	0.79	0.20
	LR model	0.50	0.90	0.85	0.62	0.31	0.80	0.21

Table 16: Performance metrics of the three models at their ideal threshold in the internal data set and external data set

It can be seen that the RF model and LR model have the highest sensitivity in the external data set. The specificity of all the three models are worse in the external data set, but can still be called reasonable. Lastly, we see that the NB model performs worse on the external data set, a possible explanation could be the difference in serum creatinine in the external data set compared to the data on which the model was built. From this, we can conclude that both the LR and RF model are good models to use to predict target attainment.

3.8. Proper scoring rules & our own scoring rule

The most common proper scoring rule in statistics is log-loss, with the following partial losses and expected loss:

$$L_1(1-p) = -\log(p), \quad L_0(q) = -\log(1-p),$$
$$S(q|p) = -q\log(p) - (1-q)\log(1-p),$$

where $q_n = \phi(\mathbf{b}^T\mathbf{x}_n)$ and $\phi()$ is the logistic link. Log-loss is sometimes called Kullback-Leibler loss or the cross-entropy term of the Kullback-Leibler divergence.

Another proper scoring rule is the squared error loss:

$$L_1(1-p) = (1-p)^2, \quad L_0(p) = p^2, \tag{7}$$
$$S(q|p) = (1-q)p^2 + q(1-p)^2 \tag{8}$$

In the literature on proper scoring rules, squared error loss is known as Brier score [20].

3.8.1. Example of our own Scoring rule

In this section, our own scoring rule will be introduced. The purpose of a scoring system is to filter out whether or not someone has attained the target or not, by just looking at the score that a particular patient received. This is far easier to implement compared to the other models we mentioned before because it is less computationally expensive and it can also be easily memorized by practitioners. This will help to make a faster diagnosis of the patient.

Notice that from the RF model we found that age, sex, creatinine and type of antibiotic were important predictors for target non-attainment. We will build 4 classification trees with all of our data, for each of these trees we take a min bucket & min split equal to 30. We chose this because 30 is around 10% of our data. Furthermore, we use the information criterion as our loss function. In Figure 17 we can see one of the decision trees after following the instructions in section 2.4.1.

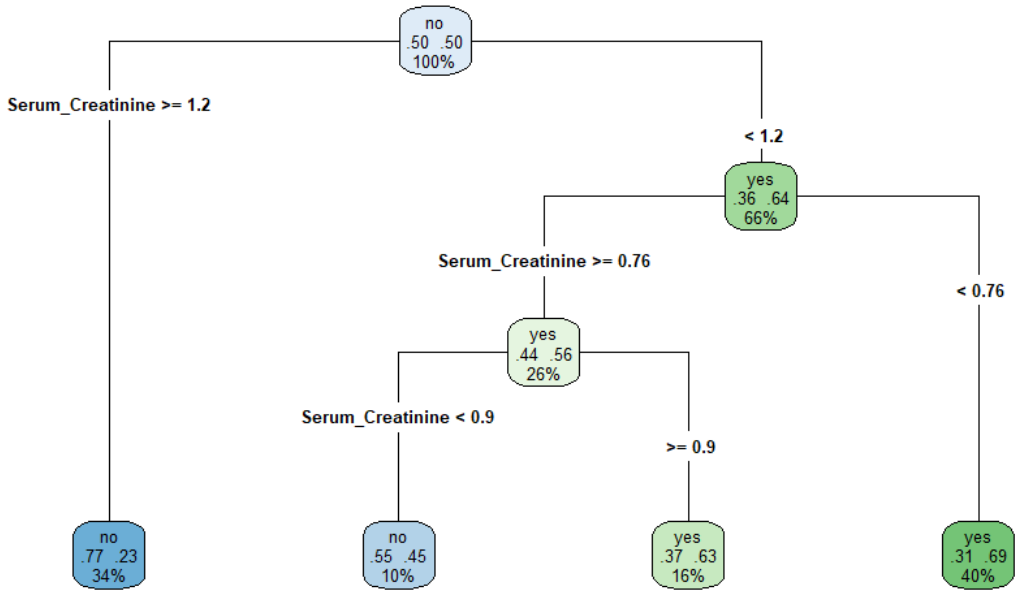


Figure 17: Decision tree of Serum Creatinine.

If we look at the Table 17 and Figure 17 we can see that the scoring system was made by choosing the yes (target non-attainment) class weight times 100 times 1. The prior that we took was equal to 0. So for example, we see that for a serum creatinine greater equal than 1.2 we give a score of 23.

Sample size	conditions	Points
Age	Age $\geq$ 77	5
	65 $\leq$ Age $<$ 77	44
	58 $\leq$ Age $<$ 65	58
	54 $\leq$ Age $<$ 58	39
	Age $<$ 54	69
Sex	Male	53
	Female	45
Serum Creatinine (SC)	SC $\geq$ 1.2	23
	0.9 $\leq$ SC $<$ 1.2	45
	0.76 $\leq$ SC $<$ 0.9	63
	SC $<$ 0.76	69
Type of Antibiotic	Amoxicillin	70
	Cefotaxim	70
	Ceftazidim	21
	Ceftriaxon	21
	Cefuroxime	55
	Flucloxacillin	70
	Meropenem	44
	Piperacillin/tazobactam	70

Table 17: Example of a scoring system using our rules without normalization.

Now in Table 17 we have the non normalized score, after normalization we get the following scoring system:

Sample size	conditions	Points
Age	Age ≥ 77	0
	$65 \leq$ Age < 77	23
	$58 \leq$ Age < 65	32
	$54 \leq$ Age < 58	20
	Age < 54	38
Sex	Male	5
	Female	0
Serum Creatinine (SC)	SC ≥ 1.2	0
	$0.9 \leq$ SC < 1.2	24
	$0.76 \leq$ SC < 0.9	13
	SC < 0.76	28
Type of Antibiotic	Amoxicillin	29
	Cefotaxim	29
	Ceftazidim	0
	Ceftriaxon	0
	Cefuroxime	20
	Flucloxacillin	29
	Meropenem	14
	Piperacillin/tazobactam	29

Table 18: Example of a scoring system using our rules with normalization and round off.

In Table 18 we have the normalized scoring system. It's important to note that when we normalized the scores for this scoring system, we rounded them off, so it's not exactly the same as the previous scoring system in Table 17. But nevertheless, for practical implementation, the normalized score is more intuitive because the score goes from 0 to 100.

To see how good the scoring system performed, we made use of the ROC curve and DCA curve. Furthermore, we used the Youden statistic to find the most optimal sensitivity and specificity. To attain an accurate ROC curve and DCA curve, we put the results of our scoring system in logistic regression, with predictors of our score and outcome target non-attainment. This gave us probabilities, which we then linked with the individual scores. Meaning that, for example, a score of 65 got a probability of 0.30 in the logistic regression and so on.

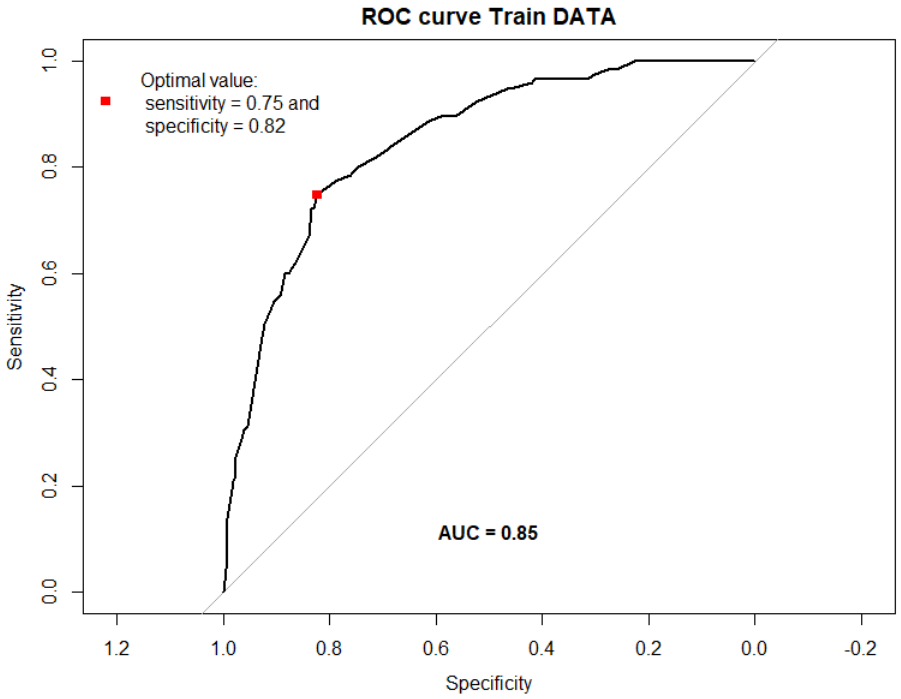


Figure 18: ROC curve of the normalized scoring system.

From the ROC curve, we found that the score of 67 was the most optimal score, with a sensitivity of 0.75 and specificity of 0.82.

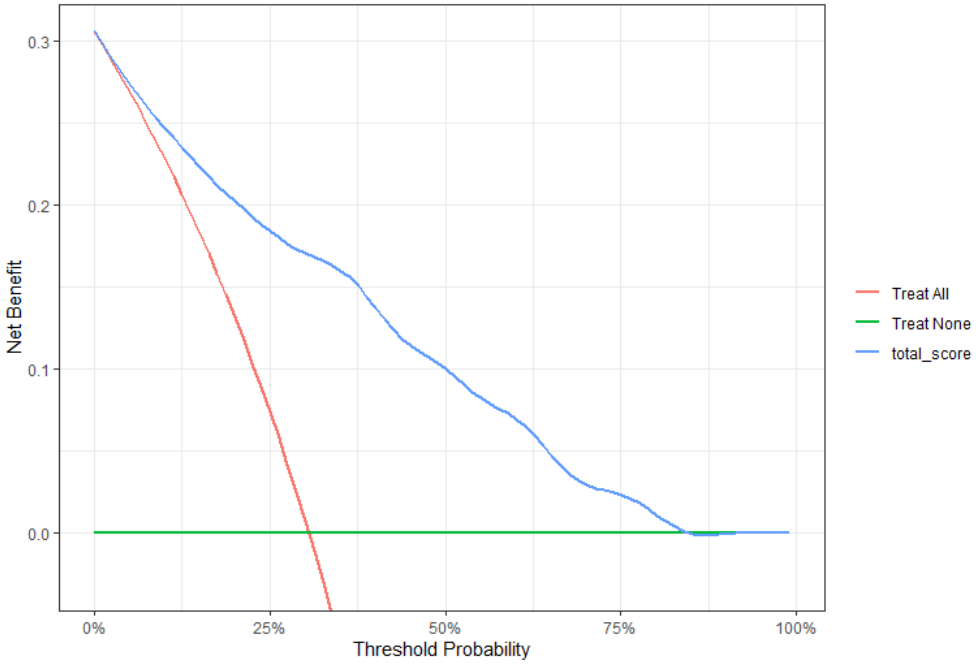


Figure 19: DCA curve of the normalized scoring system.

From the DCA curve, we see that we have a net benefit until a probability threshold of 0.80. The prevalence of target non-attainment in our data is 30%, which has a very good net benefit in the DCA curve. If we chose this cut-off, we found that the optimum score would be 65 with a sensitivity of 0.78 and specificity of 0.77.

Furthermore, if we would look at the scoring system performance over different score thresholds, we find that a score between 57 and 62 has a sensitivity above 0.80 and a specificity above 0.60. So a score between these thresholds would be ideal for a high sensitivity and reasonable specificity. At a score of 57 we had a sensitivity of 0.89 and specificity of 0.61. This would be the best cut off for this scoring system.

One thing we must note is that this scoring system was made using all the data. So it should still be externally validated. But the initial results look promising.

3.8.2. External Validation

The scoring system will be externally validated on the external data. In this section, we look at the performance of our own scoring system on this data.

The ROC of this system is given in Figure 20, where the red dot is at the score of 62. The scoring system gives us an AUC of 0.73, sensitivity of 0.83 and specificity of 0.50. The ideal score found in the internal validation by the Youden statistic was a score of 67. But we also found that a score between 57 and 62 gave us a sensitivity above 0.80 for the internal validation set. In the external data, the score of 57 gave us a sensitivity of 0.86 and specificity of 0.44.

This means that the score of 57, is a good cut-off point for our scoring system, because it works reasonably well for both the internal and external validation set.

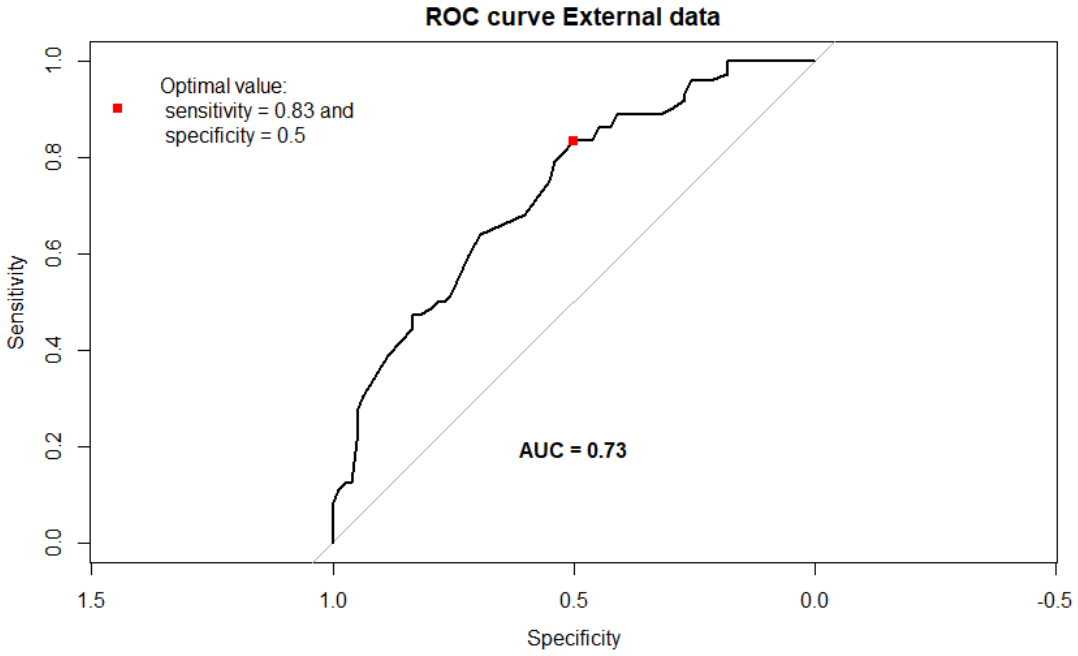


Figure 20: ROC curve of the normalized scoring system on the external data.

4. Discussion & Conclusion

The main objective was to make a model that could predict the antibiotic's target attainment. The random forest model, Naive Bayes model, and Logistic regression model were chosen. We used the predictors' sex, age, serum creatinine day 0 and type of antibiotic for all of these models. These were found using the Boruta algorithm and our brute force approach.

The main advantage of the RF model is that it can detect non-linear relations in the predictor variables. The parameters were optimized using the 10-fold twice repeated CV, which gave us $mtries = 1$, minimum rows equal to 0.3, and sample rate equal to 0.75. Furthermore, $ntree = 2500$ and the $weights$ were chosen as in Equation (6). Using all these parameters, we found that the model performed reasonably well. This was further confirmed by training the model on different training and validation sets using the stratification process. This process was done 1000 times and gave us an average NPV of 0.89, average sensitivity of 0.76 and an average specificity of 0.84,

indicating reasonably good performance. The external validation found that the model performed reasonably well, with an AUC of 0.79. The threshold of 0.20 was chosen as the ideal threshold for this model, because in the internal data, the model had a sensitivity above 0.8 and specificity around 0.6. In the external data, the model had a worse specificity (0.5) but still a sensitivity above 0.8. This threshold shows to work reasonably well for both data sets.

The logistic model also did reasonably well on the data. It had the highest AUC (0.84) out of the three models, with an average sensitivity of 0.77 and specificity of 0.83. The external validation gave an AUC of 0.80, a sensitivity of 0.89 and a specificity of 0.58. The ideal threshold for this model can be chosen around 0.21. The internal and external validation at this threshold showed a sensitivity above 0.8. The specificity was a bit worse in the external data set. A possible explanation for this could be the difference in serum creatinine in the two data sets. The model performs well at this threshold for both internal and external validation.

The Naive Bayes model had the best average sensitivity (0.81) out of all the models in the internal validation set. This model performed similarly to the other two on the internal data, with an AUC of 0.83. This model performed the worst out of the three models on the external data, with an AUC of 0.75. However, this was still reasonable. The ideal threshold for this model was around 0.17.

Our second objective was to make a scoring system based on predictors of the RF model. This was made using decision trees for each individual predictor. These trees were pruned, and the prediction weights were used as the score. We found that our scoring system was proper if we took the prior as the mean of our Bernoulli distributed target variable. This way of making a scoring system on our own data gave us an AUC of 0.85. We calculated the Youden statistic of the ROC curve for this model and found a sensitivity and specificity of 0.75 and 0.82, respectively, at a score of 67. The ideal cutoff that we chose after our analysis was We must note that this scoring system was built on all the data, so the AUC could be overestimated. In the external validation, the scoring system has an AUC of 0.73, with an ideal score of 62. It was shown that in the internal data set, the model had a high sensitivity for scores between 57 and 62. In the external data, we find that at the cutoff of 57, the model has a sensitivity of 0.86 and specificity of 0.44. The specificity is a bit worse than in the internal data. A possible explanation for this could be the different serum creatinine in the internal and external data sets. The cutoff of 57 is chosen as the ideal cutoff.

Out of the three models, the logistic and Random Forest models performed the best on both the internal and external data. This means that both models can be used to predict target attainment. One must also notice that the external data has a statistically different serum creatinine than on which the models are built. This could also indicate that the models will perform worse on the external data. However, we still see reasonable performance of the models, which is a good sign.

The RF model is more computationally intensive than the logistic regression and NB models. The RF model can be used to create a scoring system, whereas the logistic regression model provides us with p-values and OR. So, in terms of interpretability, logistic regression is the model to choose. The RF model, on the other hand, is required to create a scoring system. The simplest and least computationally intensive model is the Naive Bayes model. All three models have advantages and disadvantages, but they can all be utilized depending on the need.

The scoring system will be a tool which could be easily implemented in the existing framework of the hospital software, which would save time and money for the hospital. This will help the practitioners assess if someone will not achieve target attainment, saving time and money. In the literature [21], there are models built for one specific antibiotic. Our model predicts target attainment for a wide array of beta-lactam antibiotics. It can be seen that the models perform reasonably, this means that they can be used in practice.

For future research, we could add more data on the other antibiotics, such as Ceftazidime. This could make the models better. The three models only look at a binary target variable, we can also expand these models for a non-binary outcome. Because in practice, we have three groups, target attained, target not attained and toxic patients. This means we can also expand the scoring systems for non-binary outcomes.

Finally, the three models and scoring system work reasonably well with only four readily available predictors. The scoring system can be easily implemented in hospitals. In conclusion, the models and scoring system will aid practitioners in spotting target non-attainment and saving lives.

References

[1] A. Abdulla *et al.*, "Failure of target attainment of beta-lactam antibiotics in critically ill patients and associated risk factors: A two-center prospective study (EXPAT)," *Crit Care*, vol. 24, no. 1, p. 558, Dec. 2020, ISSN: 1364-8535. DOI: 10.1186/s13054-020-03272-z.

[2] J. A. Roberts *et al.*, "DALI: Defining antibiotic levels in intensive care unit patients: Are current beta-lactam antibiotic doses sufficient for critically ill patients?" *Clinical Infectious Diseases*, vol. 58, no. 8, pp. 1072–1083, Apr. 15, 2014, ISSN: 1058-4838. DOI: 10.1093/cid/ciu027.

[3] P. J. Simner, J. H. Medicine, and L. Miller, "Understanding pharmacokinetics (PK) and pharmacodynamics (PD)," vol. 3, no. 1, p. 4, 2018.

[4] A. Abdulla *et al.*, "The effect of therapeutic drug monitoring of beta-lactam and fluoroquinolones on clinical outcome in critically ill patients: The DOLPHIN trial protocol of a multi-centre randomised controlled trial," *BMC Infect Dis*, vol. 20, no. 1, p. 57, Dec. 2020, ISSN: 1471-2334. DOI: 10.1186/s12879-020-4781-x.

[5] M. Gijsen *et al.*, "Meropenem pharmacokinetics and target attainment in critically ill patients are not affected by extracorporeal membrane oxygenation: A matched cohort analysis," *Microorganisms*, vol. 9, no. 6, p. 1310, Jun. 16, 2021, ISSN: 2076-2607. DOI: 10.3390/microorganisms9061310.

[6] M. Gijsen *et al.*, "Meropenem target attainment and population pharmacokinetics in critically ill septic patients with preserved or increased renal function," *Infect Drug Resist*, vol. 15, pp. 53–62, 2022, ISSN: 1178-6973. DOI: 10.2147/IDR.S343264.

[7] R Core Team, *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria, 2020.

[8] D. W. Hosmer, S. Lemeshow, and R. X. Sturdivant, *Applied Logistic Regression*. John Wiley & Sons, Apr. 1, 2013, 528 pp., ISBN: 978-0-470-58247-3.

[9] G. A. F. Seber and A. J. Lee, *Linear Regression Analysis*. John Wiley & Sons, Jan. 20, 2012, 585 pp., ISBN: 978-1-118-27442-2.

[10] G. I. Webb, "Naïve bayes," in *Encyclopedia of Machine Learning and Data Mining*, C. Sammut and G. I. Webb, Eds., Boston, MA: Springer US, 2017, pp. 895–896, ISBN: 978-1-4899-7687-1. DOI: 10.1007/978-1-4899-7687-1_581.

[11] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct. 2001. DOI: 10.1023/a:1010933404324.

[12] A. Buja, W. Stuetzle, and Y. Shen, *Loss functions for binary class probability estimation and classification: Structure and applications*, 2005.

[13] T. Gneiting and A. E. Raftery, "Strictly proper scoring rules, prediction, and estimation," *Journal of the American Statistical Association*, vol. 102, no. 477, pp. 359–378, Mar. 2007, ISSN: 0162-1459, 1537-274X. DOI: 10.1198/016214506000001437.

[14] J. Bröcker, "Reliability, sufficiency, and the decomposition of proper scores," *Q.J.R. Meteorol. Soc.*, vol. 135, no. 643, pp. 1512–1519, Jul. 2009, ISSN: 00359009, 1477870X. DOI: 10.1002/qj.456.

[15] T. Hastie, R. Tibshirani, and J. H. Friedman, *The elements of Statistical Learning: Data Mining, Inference, and prediction*. Springer, 2017.

[16] M. B. Kursa and W. R. Rudnicki, "Feature selection with the **Boruta** package," *J. Stat. Soft.*, vol. 36, no. 11, 2010, ISSN: 1548-7660. DOI: 10.18637/jss.v036.i11.

[17] K. H. Zou, A. J. O'Malley, and L. Mauri, "Receiver-operating characteristic analysis for evaluating diagnostic tests and predictive models," *Circulation*, vol. 115, no. 5, pp. 654–657, Feb. 6, 2007, ISSN: 0009-7322, 1524-4539. DOI: 10.1161/CIRCULATIONAHA.105.594929.

[18] A. Niculescu-Mizil and R. Caruana, "Predicting good probabilities with supervised learning," in *Proceedings of the 22nd international conference on Machine learning*, ser. ICML '05, New York, NY, USA: Association for Computing Machinery, Aug. 7, 2005, pp. 625–632, ISBN: 978-1-59593-180-1. DOI: 10.1145/1102351.1102430.

[19] E. Scornet, "Tuning parameters in random forests," *ESAIM: Proceedings and Surveys*, vol. 60, pp. 144–162, Dec. 14, 2017. DOI: 10.1051/proc/201760144.

[20] R. L. Winkler *et al.*, "Scoring rules and the evaluation of probabilities," *Test*, vol. 5, no. 1, pp. 1–60, Jun. 1, 1996, ISSN: 1863-8260. DOI: 10.1007/BF02562681.

[21] U. Liebchen *et al.*, "Evaluation of the merorisk calculator, a user-friendly tool to predict the risk of meropenem target non-attainment in critically ill patients," *Antibiotics*, vol. 10, no. 4, 2021, ISSN: 2079-6382. DOI: 10.3390/antibiotics10040468.