Learning to mitigate reliance on features with missing values in interpretable prediction models

Master's thesis in Computer science and engineering

Tianyi Duan

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2024

MASTER'S THESIS 2024

Learning to mitigate reliance on features with missing values in interpretable prediction models

Tianyi Duan

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2024

Learning to mitigate reliance on features with missing values in interpretable prediction models
Tianyi Duan

Supervisor: Fredrik D. Johansson and Lena Stempfle, Computer Science and Engineering Department
Examiner: Simon Olsson, Computer Science and Engineering Department

Master's Thesis 2024
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 73 658 2385

Typeset in LaTeX
Gothenburg, Sweden 2024

Learning to mitigate reliance on features with missing values in interpretable prediction models
Tianyi Duan
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

In the healthcare area, it is common for datasets to contain observations that are missing for the corresponding features. Predicting outcomes with such datasets in supervised learning tasks often results in outcomes that are heavily influenced by these missing values. This thesis modifies two original machine learning algorithms and introduces two novel models: the Least Absolute Shrinkage and Selection Operator Mitigating Reliance (LASSOMR) and the Decision Tree Mitigating Reliance (DTMR). Both models are designed to reduce dependency on features with missing values during predictions. This reduction is achieved by penalizing features that have missing values, thereby decreasing the model's reliance on these features. The synthetic dataset and real-world dataset are used to explore that DTMR and LASSOMR models give a larger penalty to the features that have larger missing ratios. As a result, the coefficient value of the features becomes less leading to the goal of relying less on features having missing values. Additionally, real-world datasets with missing values evaluate the performance of these models against baseline methods, confirming that the models perform comparably while effectively mitigating reliance on missing value features.

Keywords: Machine Learning, Supervised Learning, Missing Values, Healthcare

Acknowledgements

I want to express my thanks to Fredrik Johansson and Lena Stempfle for their full support and assistance with my Master's Thesis. Through each conversation with Fredrik Johansson, I learned how to conduct research, understand underlying principles, perform experiments, and explain phenomena, from which I gained a great deal. I am also grateful to Lena for her detailed guidance. When I was still unfamiliar with the topic, she provided many suggestions on how to better grasp the entire principle through experiments, which also gave me a clearer understanding of my Master's Thesis. Also, she gave me a lot of experience in conducting research, starting with small tasks, gradually breaking down and dividing the issues, and then analyzing and solving the problems. For the writing part, she gave me lots of guidance and helped me on hand with academic writing. Additionally, I want to thank the members of the Healthy AI Lab; attending the weekly group meetings allowed me to learn about other research areas. I would like to thank the examiner Simon Olsson for his guidance on my Master's Thesis, and for providing advice and suggestions on my report and potential issues in the experiments. This research experience has paved the way for my future research, and I thoroughly enjoyed this time.

Tianyi Duan, Gothenburg, August 6, 2024

Contents

List of Figures

List of Tables

1
Introduction

The vast majority of machine learning models are widely valued in predictive tasks due to their interpretability, a particularly vital feature in fields with high demand for model interpretability such as healthcare and financial areas [1, 2]. Variables missing or unobserved are common to encounter at the time of prediction in these domains (due to data loss, privacy issues, dataset fusions, etc) [3]. Various methodologies have been explored to address the challenge of model prediction with incomplete data. Imputation [4] is an effective way to deal with missing value problems, but it requires a strong assumption of the missing value [5]. It often assumes that the data is missing at random (MAR) [6]. Also, several limitations exist in the imputation method: altering the variance of the dataset, distorting the distribution of the dataset, and not applicable to specific cases [3, 7].

Other methods include Bayesian modeling [8], the application of fallback default rules [9, 10], weighted estimating equations [11], prediction with missingness indicators [12]. A limitation of these methods is that: they either lack interpretability (estimation weighting) or rely on features with missing values (imputation methods, fallback rules, missingness indicators).

The methods that deal with missing values either lack interpretability or rely on features with missing values. To overcome these limitations, I modify two methods relying less on features having missing values with high interpretability. The ideas are inspired by the Least Absolute Shrinkage and Selection Operator (LASSO) method and the Decision Tree method. By modifying the LASSO method and the Decision Tree method, I created the LASSO Mitigating Reliance (LASSOMR) and Decision Tree Mitigating Reliance (DTMR) method. These methods aim to maintain high interpretability and reduce dependence on features with missing values. My method not only relies less on features having missing values but also has a comparable performance with the method Decision Tree with imputation and the LASSO method with imputation. To more directly compare the degree of dependency on missing values across different models, the concept of reliance is defined. It is used to measure the extent to which the model's predictions depend on missing values. If the value of reliance is high, it has a high degree of dependency on missing values. This thesis defines reliance differently in the linear and tree models to accommodate their distinct structural characteristics. However, they both reflect how the linear model and tree model rely on features with missing values when they make a prediction.

The LASSOMR method has the function of variable selection. It penalized the features with missing values based on the missingness. This can clearly explain how the LASSOMR model mitigates reliance on features with missing values. The DTMR method modifies the original Decision Tree by incorporating a penalty mechanism during the information gain calculation. When choosing a candidate node as the decision node, a penalty value is given to the feature that has missingness. The algorithm makes the process of choosing a candidate node not only consider the reduction of entropy but also counts the reliance on missing values. The DTMR model chooses candidate nodes based on the information gain calculation which maintains the interpretability. The LASSOMR model and DTMR model address a critical aspect often ignored in traditional approaches. They provide a novel perspective on mitigating reliance on missing values that aligns with the increasing demand for interpretability.

Consider an example of a classification task in Figure 1.1. The ’Smoker’, ’Weight’, and ’Age’ are the features. The target of ’Risk’ is the risk of respiratory diseases. In this Figure 1.1, the first record of this dataset is for the feature ’Smoker’, the observation value is ’Yes’. The decision path veers left if the observation ’Smoker’ is true. Then this record arrived in the decision node ’Age $<$ 45’. For the feature ’Age’, the observation is missing. This means that to classify this record to a specific classification, it relies on the missing value. This example illustrates what is missing in the Decision Tree method.

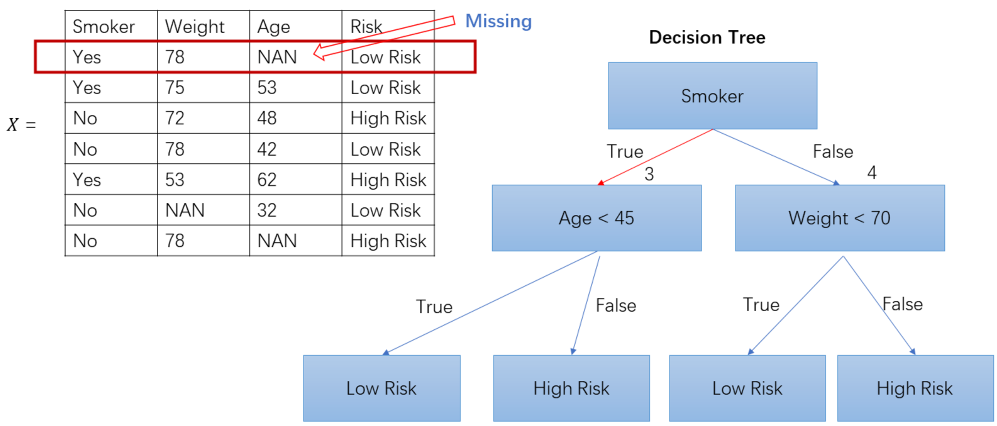


Figure 1.1: For the record on dataset X in row one, to classify this record to a specific classification, it relies on the feature ’Age’. Since the observation for the feature ’Age’ has missing values.

The outcomes of this research have the potential to significantly advance the field, offering new insights and techniques for robust model development to overcome the limitations of traditional methods that deal the dataset having missing values

1.1 Contributions

The contributions are summarized as follows: 1) I have developed a modified version of the LASSO model, termed the LASSOMR model. It can penalize the features with missing values based on the degree of missingness. 2) By integrating the penalty term into the information gain calculation in the original Decision Tree method, I propose an improved method which is DTMR, it not only considers the feature importance but also considers the missingness in a feature. 3) I introduced the concept of reliance and a formula for calculating dependency on missing values in linear models and decision trees. This provides a systematic approach to quantifying the impact of missing data on model predictions. 4) The newly developed model maintains high interpretability while minimizing dependence on missing data. They achieve prediction performance comparable to that of the traditional Decision Tree and LASSO models, demonstrating their effectiveness in practical applications.

2

Background

This study specifically addresses the challenge of limitations that previous methods have when dealing with datasets having missing values. Previous methods have limitations: they either have less interpretability or rely on features with missing values. To overcome these limitations, I modified two models that aim for good prediction, meaning low prediction errors and minimizing the empirical risks. The two models have high interpretability and aim to rely less on features that have missing values. The reliance is defined as the degree of dependence on features with missing values within the linear model and tree model in this thesis. Also, the modified models provide interpretability on how the models reduce the reliance on features having missing values.

Considering the output $Y \in \mathbf{R}^d$, based on a vector of input features $\tilde{X} = [x_1, x_2, x_3, \ldots, x_d]^T \in \mathbf{R}^d$, missing values may be present during training or test time. Missingness in the dataset is characterized by a random binary mask $M = [m_1, m_2, m_3, \ldots, m_d]^T \in \{0,1\}^d$. Let $m_{i,j} = 1$ when $\tilde{x}_{i,j}$ is missing, and $m_{i,j} = 0$ when $\tilde{x}_{i,j}$ has a value. The dataset containing missing values can be effectively reconstructed using a missingness mask in Formula 2.1.

$$\tilde{X_{ij}} = \begin{cases} \tilde{X_{ij}} & \text{if } M_{ij} = 0 \\ \text{NAN} & \text{if } M_{ij} = 1 \end{cases} \tag{2.1}$$

An example of representing the missingness in $\tilde{X}$ is in Figure 2.1. The input dataset X has a missing value. Then, the binary mask matrix M records the information of missing values in the input matrix $\tilde{X}$. The value 1 represents that the dataset has a missing value and 0 means that the dataset doesn't have a missing value. The missingness matrix M can help assisting in calculating the reliance r.

The occurrence of missingness can be various but can be categorized into three main categories [4]: missing completely at random (MCAR), missing at random (MAR), and missing not at random (MNAR). To define the missingness mechanism, both $\tilde{X}$ and M are modeled as random variables with probabilistic distribution $P_{\tilde{X}}$ and P_M respectively. I parametrize the missingness distribution P_M by a parameter ϕ. The observation can be said to be missing completely at random if an observation is missing and is independent of the variables and the value of the observation itself: the probability that an observation is missing does not depend on variables and

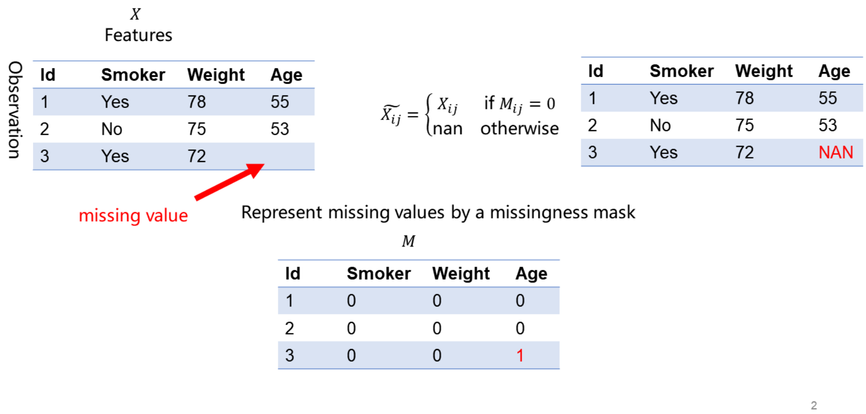


Figure 2.1: An example of a missingness matrix corresponding with a dataset with missing values. M is the missingness matrix that is built from the dataset X. The value 1 in matrix M means that dataset X has a missing value. Then reconstruct the dataset by filling NAN value if the value in the dataset is missing.

observations.

$$\mathbb{P}_M(M \mid \tilde{X^{obs}}, \tilde{X^{mis}}; \phi) = \mathbb{P}_M(M) \qquad \forall\, \phi \tag{2.2}$$

The observation can be said to be MAR if the missingness $\tilde{X^{mis}}$ dependents on the observation$\tilde{X^{obs}}$. The formula can be written as follows [13]:

$$\mathbb{P}_M(M \mid \tilde{X^{obs}}, \tilde{X^{mis}}; \phi) = \mathbb{P}_M(M \mid \tilde{X^{obs}}; \phi) \qquad \forall\, \phi,\, \forall\, \tilde{X^{mis}}. \tag{2.3}$$

The missingness is MNAR if missingness depends on the value of missingness itself and potentially depends on the observations.

Imputation is an effective method for solving missing value problems. However, imputation has its disadvantages. Mean value imputation often leads to biased statistical inference, by underestimating the variance of the dataset and potentially distorting the mean value of the dataset[14]. This underestimation is due to the mean imputation not counting for the uncertainty of the missing value mechanism, thereby reducing the variability of the dataset[15]. Even sophisticated methods like multiple imputation can underestimate the variance if not properly tuned to reflect the underlying mechanism[16]. These limitations highlight the critical need to mitigate reliance on the imputation method for the missing value problem.

This study is dedicated to achieving the target of reducing the dependence on the imputation method using interpretable models. The solution mitigates reliance on imputation and can improve interpretability over black-box methods. The methodology of my research begins by modifying the LASSO (Least Absolute Shrinkage and Selection Operator) method. The LASSO method gives the same penalty term for each coefficient. In my method, I modify the penalty term by giving different penalty terms which are decided by the degree of missingness in its corresponding features. Then, I modify the information gain calculation in the Decision Tree by adding a penalty mechanism to the features that are missing.

2.1 LASSO Method

The Least Absolute Shrinkage and Selection Operator (LASSO) method, introduced by Robert Tibshirani [17], represents a significant advancement in the field of regression analysis and variable selection. LASSO is a regularization technique that can enhance the prediction accuracy and interpretability of the statistical models. It is applied by constraining the size of the coefficients. This constraint involves adding a penalty equivalent to the absolute value of the magnitude of the coefficients, controlled by a tuning parameter λ. The reason to choose the LASSO method is that the LASSO method has the function of variable selection. The formula is shown in Formula 2.4. The penalty term λ is used to penalize the features. The normal LASSO method penalizes each feature with the same penalty term.

$$\min_{\theta} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{d} \theta_j \times x_{i,j} - y_i \right)^2 + \sum_{j=1}^{d} \lambda \left| \theta_j \right| \right\} \tag{2.4}$$

The LASSO method gives the inspiration that the penalty term can be modified to penalize the features based on the degree of missingness.

2.2 Decision Tree Method

Decision Tree is a fundamental and highly versatile machine learning algorithm, used extensively for both classification and regression tasks. It is a graphical representation of possible solutions to a decision based on certain conditions. Introduced in their modern form by Breiman in their seminal work "Classification and Regression Trees" [18], Decision Trees operate by breaking down a dataset into smaller subsets based on the feature values, while an associated tree is incrementally developed. The result is a tree-like model of decisions, where each node represents a feature, each branch represents a decision rule, and each leaf node represents an outcome. One of the key advantages of Decision Trees is their interpretability and simplicity; the decision-making process is easy to understand and visualize, making them particularly useful for business decisions[19]. It is composed of nodes that represent tests on attributes, edges, or branches that correspond to the outcomes of these tests, and leaf nodes that represent class labels or decision outcomes. The primary goal of a decision tree is to model decision-making by sequentially splitting data into increasingly specific subsets, which is analogous to asking a series of questions that can divide the dataset into discernable attributes leading to a final decision [19]. The key idea of decision tree is to reduce the maximum entropy when choosing a decision node. Before choosing a decision node, the entropy can be calculated. Then, it calculates each entropy after choosing a candidate node as a decision node. In this way, the candidate node that can decrease the maximum entropy can be chosen. Each branch from this node represents the possible answers, and this process is repeated at each subsequent node until a leaf node is reached, which provides the final decision or classification.

The decision tree makes the decision process intuitive, interpretable, and easy to understand. The decision tree gives an inspiration that when using a decision tree to make a decision, each row of observations in the dataset will follow the path of the tree. When deciding on a certain row of observations, some of the decision nodes do not participate in the decision-making process. If some of the nodes do not participate in the decision-making process, even if the observation has missingness in these nodes, it will not influence the decision-making process.

$$\text{Info Gain}(S, J) = -\sum_{i=1}^{n} p_i \log p_i - \sum_{j \in J} \left(-\frac{|S_j|}{|S|} \sum_{k=1}^{m} q_k \log q_k \right) \tag{2.5}$$

p_i is the probability of an element in S belonging to class i, n is the number of different classes in the dataset. q_k represents the probabilities of class k in subset S_j, and m is the number of different classes in the subset dataset S_j. log represents the logarithm, measuring the information in bits. S_j is the subset of S for each value j that attribute J can take, $|S_j|$ is the number of elements in subset S_j, $|S|$ is the total number of elements in S, The summation is across all unique values j of attribute J.

3

Methods

In this study, I propose two methods that mitigate reliance on features with missing values when the models make a prediction. 1) The LASSOMR method gives different penalties to the coefficients based on the degree of missingness. 2) The DTMR method penalizes the features with missingness in the information gain calculation. Compared with the traditional methods, the two methods have the advantage of mitigating reliance on features with missing values and high interpretability. Then this chapter introduces the definition of reliance in the LASSOMR method and DTMR method. Reliance is used to measure the extent to which a model's predictions depend on missing values.

3.1 LASSOMR Method

The LASSO method adds a penalty λ equivalent to the absolute value of the magnitude of coefficients (L1 regularization). The penalty term is modified to λ_j to achieve the target of penalizing more on features having more missing values. The missing ratio of a certain feature j determines the value of λ_j. When a feature j has a large value of the missing ratio, the penalty term λ_j is also larger. As a result, it gives a large penalty to this feature to shrink its coefficient value to be small. As a result, it contributes less to the final prediction, achieving the target of mitigating reliance on features with missing values. Likewise, when a feature has a small value of missing ratio, the penalty term λ_j is smaller. It is defined λ_j as Formula 3.1.

$$\lambda_j = [\frac{\sum_{i=1}^{n} \mathbb{1}[x_{i,j} = NA] + 1}{n}] \times \lambda \tag{3.1}$$

$\sum_{i=1}^{n} \mathbb{1}[x_{i,j} = NA]$ is the number of missing values in feature j. n is the number of observations in feature j, and λ is the penalty term parameter that can be set. The reason for $\sum_{i=1}^{n} \mathbb{1}[x_{i,j} = NA] + 1$ plus one is that if one feature does not have missingness, the value of $\sum_{i=1}^{n} \mathbb{1}[x_{i,j} = NA]$ equals zero. Since it needs to give a penalty to the feature that does not have missingness, the value of $\sum_{i=1}^{n} \mathbb{1}[x_{i,j} = NA] + 1$ can avoid becoming zero. By using the penalty term λ_j, the LASSOMR method penalizes the features based on the missingness. When a feature has large missingness, the λ_j increases, giving a large penalty to the feature, shrinking the coefficient value for this feature, and decreasing the contribution to the prediction for the feature having large missingness. The λ_j achieves the goal of mitigating reliance on features based on the missing ratio. This LASSOMR formula is defined as:

$$\min_{\theta}\left\{\frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=1}^{d}\theta_j\times x_{i,j}-y_i\right)^2+\sum_{j=1}^{d}\lambda_j\left|\theta_j\right|\right\} \tag{3.2}$$

$\lambda_j=\left(\frac{\sum_{i=1}^{n}\mathbb{1}[x_{i,j}=\mathrm{NA}]+1}{n}\right)\times\lambda$ is the regularization parameter that controls the amount of shrinkage applied to the coefficients. θ_j is the coefficient value for feature j. $x_{i,j}$ is the observation, n is the number of observations, d is the number of features. Training a model with distinct penalty terms, λ_j, as specified in Formula 3.2, presents challenges. The standard LASSO implementation in the scikit-learn library applies a uniform penalty term across all features. The scikit-learn library does not provide methods that penalize each feature differently. To make the LASSOMR model easy to train and implement, it can use math transformation to transform the Formula 3.2 to a LASSO-like problem.

$$\begin{aligned}
\textbf{Step 1: }&\min_{\theta}\left(\frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=1}^{d}\theta_j\times x_{i,j}-y_i\right)^2+\sum_{j=1}^{d}\lambda_j|\theta_j|\right)\\
\textbf{Step 2: }&\min_{\theta}\left(\frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=1}^{d}\theta_j\times x_{i,j}-y_i\right)^2+\sum_{j=1}^{d}\lambda_j\times\frac{\lambda}{\lambda_j}\times\frac{\lambda_j}{\lambda}\times|\theta_j|\right)\\
\textbf{Step 3: }&\min_{\theta}\left(\frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=1}^{d}\theta_j\times x_{i,j}-y_i\right)^2+\sum_{j=1}^{d}\lambda_j\times\frac{\lambda}{\lambda_j}\times|\frac{\lambda_j}{\lambda}\theta_j|\right)\\
\textbf{Step 4: }&\min_{\beta}\left(\frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=1}^{d}\beta_j\times\frac{\lambda}{\lambda_j}\times x_{i,j}-y_i\right)^2+\sum_{j=1}^{d}\lambda|\beta_j|\right)
\end{aligned} \tag{3.3}$$

For the math transformation, in step 2, it rewrites the λ_j as $\lambda_j\times\frac{\lambda}{\lambda_j}\times\frac{\lambda_j}{\lambda}$. Since the $\frac{\lambda_j}{\lambda}$ is positive, in step 3, it can be put in the absolute value with θ_j. Then replace $\frac{\lambda_j}{\lambda}\theta_j$ with β_j by letting $\beta_j=\frac{\lambda_j}{\lambda}\theta_j$, which is the new coefficient that needs to estimate. Since $\beta_j=\frac{\lambda_j}{\lambda}\theta_j$, it can replace $\beta_j\times\frac{\lambda}{\lambda_j}$ with θ_j due to $\theta_j=\beta_j\times\frac{\lambda}{\lambda_j}$. The formula in step 4 becomes a LASSO-like problem.

Reweight the input dataset $\tilde{X}$ with $\frac{\lambda}{\lambda_j}$, then use the standard LASSO package to train the model, it can obtain the same effect in step 1 the formula that modified. As a result, it can achieve the goal of the model penalizing more for the features with a larger missing ratio.

3.2 The Definition Of Reliance In Linear Model

In the LASSOMR method, the reliance r is used to measure the reliance on features with missing values. The definition of reliance differs between the LASSOMR and the DTMR methods, due to the different structures in the linear and tree models. However, the principle included in the definition is similar. The larger the reliance

r, the more the model depends on missing values. Reliance r occurs if there exists a feature j such that: 1) $x_{i,j} = NA$ (the value is missing), 2) and the coefficient β_j in a linear model is non-zero. Then use a binary indicator $r(x_{i,j}) \in \{0, 1\}$ to denote reliance. For instance, consider an observation where variable i represents Age = 73, and feature j corresponds to Age. If $x_{i,j} = NA$ (Age is missing) and β_j is non-zero, the binary indicator $r(x_{i,j}) = 1$ signifies reliance. The reliance for a feature j can be defined as:

$$r_j = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}[x_{i,j} = NA] \mathbb{1}[\beta_j \neq 0], \tag{3.4}$$

r_j means the reliance in feature j, n is the number of observations in feature j, $x_{i,j}$ is variable i in observation j, β_j is the coefficient in feature j, $\mathbb{1}[\beta_j \neq 0]$ and $\mathbb{1}[x_{i,j} = NA]$ represents an indicator function. The indicator function is a mathematical function used primarily in probability and statistics that takes the value 1 when its argument (the condition inside the brackets) is true, and 0 otherwise. Using formula 3.4 can calculate the reliance for each feature. The Equation 3.5 shows the average reliance for the LASSOMR model.

$$r = \frac{1}{d} \sum_{j=1}^{d} r_j \tag{3.5}$$

3.3 DTMR Method

To effectively mitigate reliance on features with missing values, the DTMR method introduces a penalization mechanism for features having missing values. The new information gain formula is shown in Formula 3.6.

$$\begin{aligned} \text{Info Gain}(S, J) = &- \sum_{i=1}^{n} p_i \log p_i - \sum_{j \in J} \left(-\frac{|S_j|}{|S|} \sum_{k=1}^{m} q_k \log q_k \right) \\ &- \lambda \times \left(-\frac{m_{s,j}}{n_s} \log \left(\frac{m_{s,j}}{n_s} \right) \right) \end{aligned} \tag{3.6}$$

$m_{s,j}$ is the number of data that remains undetermined when choosing a feature j as the decision node implying that the data has missing values in feature j. $n_{s,j}$ is the number of data that arrives in the node s. The first half of the formula definition before the penalty term $-\lambda \times \left(-\frac{m_{s,j}}{n_s} \log \left(\frac{m_{s,j}}{n_s} \right) \right)$ is the same as Formula 2.5. This penalization is directly proportional to the extent of a feature's reliance on missing values; features heavily dependent on missing values incur greater penalties. This strategy is crucial in adjusting the selection criteria for features within Decision Tree algorithms.

Incorporating penalties fundamentally alters the decision-making process for selecting features as nodes within a decision tree. By integrating a penalty system, the method extends this criterion to also account for a feature's dependency on missing values. This dual consideration ensures that the construction of the decision tree not only aims for optimal entropy reduction but also minimizes reliance on missing values. Such an approach promotes the development of models that are both robust and less dependent on potentially biased or inaccurate imputations, leading to more reliable and interpretable decision-making processes.

3.4 The Definition Of Reliance In Tree Model

In the DTMR method, the definition of reliance r is similar to the definition in the LASSOMR method, the smaller the reliance r, the less the model depends on missing values. It is defined as the dependence on features with missing values for an observation $x_{i,j}$. Reliance r is calculated if there exists a feature j such that: 1) $x_{i,j} = NA$ (the value is missing), 2) and feature j is used as a decision node for observation $x_{i,j}$. A binary indicator $r(x_{i,j}) \in \{0, 1\}$ denotes reliance. For instance, consider an observation where variable i represents Age = 73, and feature j corresponds to Age. If $x_{i,j} = NA$ (Age is missing) and Age is used as decision node, the binary indicator $r(x_{i,j}) = 1$ signifies reliance. Firstly, define $\delta(i, j)$ as follows in Formula 3.7. $\delta(i, j)$ equals to one meaning that the observation $x_{i,j}$ for feature j is missing. When it equals zero, meaning that the observation $x_{i,j}$ for feature j is not missing.

$$\delta(i,j) = \begin{cases} 1 & \text{if Feature } j \text{ is used as the decision node for } x_{i,j} \\ 0 & \text{otherwise} \end{cases} \tag{3.7}$$

Then, the feature can be expressed as:

$$r_j = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}[x_{i,j} = \text{NA}] \cdot \delta(i,j) \tag{3.8}$$

Where n is the number of nodes in decision node j, $x_{i,j}$ is the observation for feature j. $\mathbb{1}[x_{i,j} = NA]$ is the indicator function. By using Formula 3.8, it can calculate the reliance for the decision node. Then it can calculate the reliance for the whole decision tree by using:

$$r = \frac{1}{\zeta} \sum_{j=1}^{m} r_j \tag{3.9}$$

where ζ is the number of decision nodes.

4

Experiment Setup

This chapter outlines the preparatory setup required before experimentation. Firstly, this chapter introduces how to implement the LASSOMR method and DTMR methods. Subsequently, it describes the metric employed to measure the performance. The final section describes the real-world dataset and the missing ratio of its features.

4.1 Implementation

This thesis proposes two methods LASSOMR and DTMR that can mitigate reliance on features with missing values. In the data processing part, zero imputation fills in the zero value with the NAN value. Then label encoding is used for the category data. The third step is to use the standard scaler method in scikit-learn to preprocess the data. In the next step, reweight the dataset with $\frac{\lambda}{\lambda_j}$. The data preprocessing step is finished after these steps. For the model part, use the scikit-learn lasso package to train the model. The parameter term λ was tuned, for values from 10^{-2} to 10^{4}, using a 10-fold cross-validation on the training portion. Then the LASSOMR method can get a temporal coefficient β. By multiplying a parameter $\frac{\lambda}{\lambda_j}$, it can reweight the coefficient β to get a true parameter θ. For the classification tasks, LASSOMR uses logistic regression with the l1 penalty term in the scikit-learn package to train the model.

The DTMR method implements the information gain differently compared with the Decision Tree method. The DTMR method penalizes the candidate node if the observation for this feature in this node is missing. The information gain calculation is defined in Formula 3.6. Then use 10-fold cross-validation to tune the parameter max_depth, and penalty term λ. The parameter max_depth means the max depth in the decision tree which is selected in the range from 4 to 20. The λ is chosen from 10^{-2} to 10^{2}.

For the regression task, the metric that is used to measure the performance is Mean Squared Error(MSE), R^2. MSE provides a concrete measure of the model's error in terms of squared units of the outcome variable. This helps in understanding the magnitude of the error directly. R^2 is useful for comparing the explanatory power of different models on the same dataset. The mean squared error (MSE) is calculated using the 'mean_squared_error' package, which requires both the predicted values and the ground truth values as inputs. R^2 is calculated as $R^2 = 1 - \frac{S_{res}}{S_{tot}}$. S_{res} is the sum of squares of the residual errors (difference between actual and predicted values). S_{tot} is the total sum of squares (the difference between actual values and the mean of actual values).

For the classification task, the metric is F1 score, Area Under the ROC Curve(AUC). F1 Score helps to balance the trade-off between precision and recall, making it a better measure than accuracy in cases where class imbalance exists. A higher AUC value would indicate better model performance in distinguishing between the positive and negative classes. The F1 score is implemented for the package scikit-learn for the function 'f1_score'. The AUC score is implemented for the package scikit-learn for the function 'auc_score'. For small datasets, the training dataset and test datasets are randomly split 10 times. Then train the model and measure the average performance in test datasets. The purpose of splitting the dataset many times is to avoid the small test dataset influencing the measurement of performance. Randomly splitting the dataset many times can prevent random splitting from influencing the performance in small datasets.

4.2 Dataset

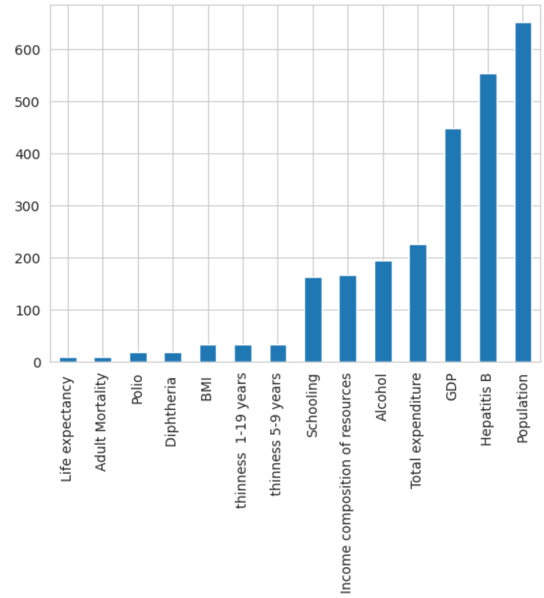
To measure the effect of the model, the synthetic dataset is used to research the change of the coefficient compared with the LASSOMR model and LASSO model. It is also used to measure the structure of decision tree change by giving different penalty values λ between the DTMR method and the Decision Tree method. To measure the effect of the model, 3 real-world datasets with missingness are used to measure the performance. The life expectancy dataset is from the source World Health Organization. Support2 dataset[20] and bone marrow transplant for children dataset [21] are from the UC Irvine Machine Learning Repository. All three datasets exhibit missingness. The life expectancy dataset is employed for regression tasks, while the support2 and bone marrow transplant for children datasets are utilized for classification tasks.

4.2.1 Synthetic Dataset

The synthetic dataset is formulated as $\theta^T X + \epsilon$ to assess the model's ability to achieve the target that is penalized more for the features having a large missing ratio. Given that θ is a 5×1 parameter vector and X is a 1000×5 feature matrix, where 1000 denotes the number of observations and 5 is the number of features. ϵ is the noise distributed according to a Gaussian distribution. The synthetic dataset $\theta^T X + \epsilon$ tests the model's ability to penalize features with a higher missingness ratio.

4.2.2 Life Expectancy Dataset

The life expectancy dataset is from the Global Health Observatory (GHO) data repository under the World Health Organization (WHO) of all countries. The target for this dataset is life expectancy meaning how many years that people live. It is used as a regression task. It contains health-related features and factors around 193 countries which are related to health. The dataset considers data from 2000-



(a) Bar chart of the number of missing values in the life expectancy dataset

	Missing Ratio
Population	22.191967
Hepatitis B	18.822328
GDP	15.248468
Total expenditure	7.692308
Alcohol	6.603131
Income composition of resources	5.684139
Schooling	5.547992
BMI	1.157250
thinness 1-19 years	1.157250
thinness 5-9 years	1.157250
Polio	0.646698
Diphtheria	0.646698
Life expectancy	0.340368
Adult Mortality	0.340368

(b) The missingness ratio in life expectancy dataset

Figure 4.1: Analysis of missingness in the life expectancy dataset.

2015 to show the factors that can influence the dataset in the regression task. It can be seen that 13 features have missing values for the life expectancy dataset. The missingness mechanism is shown in Figure 4.1. When the LASSOMR model and DTMR model are used to mitigate reliance on features with missingness, it aims to mitigate reliance on these 13 features with missingness. It can be seen that the feature 'Population', 'Hepatitis', and 'GDP' has the largest missingness, the percentage of the three features is all above 10%. For the feature 'Total expenditure', 'Alcohol', 'Income composition of resources', and 'Schooling', the missingness ratio of them is between 5% to 7%. The remaining features have missingness below the 2%.

4.2.3 Support2 Dataset

The support2 dataset comprises data from 9,105 critically ill patients across five medical centers in the United States. It encompasses 42 features and spans two collection periods: 1989-1991 and 1992-1994. Each entry corresponds to the records of hospitalized patients who satisfied the inclusion and exclusion criteria for nine disease categories: acute respiratory failure, chronic obstructive pulmonary disease, congestive heart failure, liver disease, coma, colon cancer, lung cancer, multiple organ system failure with malignancy, and multiple organ system failure with sepsis. This dataset is primarily utilized for classification tasks to predict hospital mortality rates. Understanding these rates is crucial for hospitals and medical professionals, particularly in addressing the national concern regarding patients' loss of autonomy toward the end of life.

Significant missingness affects several features within the dataset. A detailed analysis reveals that the attributes ’adlp’, ’urine’, and ’glucose’ exhibit the highest levels of missing data, each exceeding 49%. The features ’bun’, ’totmcst’, ’alb’, ’income’, ’adls’, ’bili’, ’pafi’, and ’ph’ show missingness rates ranging from 25% to 47%. The attributes ’prg2m’, ’edu’, ’prg6m’, ’sfdm2’, ’totcst’, ’wblc’, ’charges’, ’avtisst’, and ’crea’ have less than 20% missing data. Overall, the support2 dataset contains 20 features with significant missingness. The description of these features and a detailed table of these ratios are provided in Table 4.1.

Table 4.1: Analysis of missingness in the support2 dataset

Feature	Description	Missing ratio
adlp	Index of Activities of Daily Living (ADL) of the patient,filled out by the patient. Higher values indicate more chanceof survival, measured at day 3.	61.95
urine	Urine output measured at day 3.	53.40
glucose	Glucose levels measured at day 3.	49.42
bun	Blood urea nitrogen levels measured at day 3.	47.80
totmcst	Total micro cost.	38.17
alb	Serum albumin levels measured at day 3.	37.03
income	Income of the patient.	32.75
adls	Index of Activities of Daily Living (ADL) of the patient.	31.49
bili	Bilirubin levels measured at day 3.	28.57
pafi	The ratio of arterial oxygen partial pressure (PaO2 in mmHg) to fractional inspired oxygen (FiO2 expressed as a fraction).	25.54
ph	Arterial blood pH.	25.09
prg2m	Physician’s 2-month survival estimate for patient.	18.11
edu	Years of education.	17.95
prg6m	Physician’s 6-month survival estimate for patient.	17.94
sfdm2	Level of functional disability of the patient in a 1-5 scale,with 5 being the highest severity.	15.38
totcst	Total ratio of costs to charges (RCC) cost.	9.75
wblc	Counts of white blood cells (in thousands) measured at day 3.	2.33
charges	Hospital charges.	1.89
avtisst	Average TISS score, days 3-25, where Therapeutic Intervention Scoring System (TISS) is a method for calculating costs in the intensive care unit (ICU) and intermediate care unit (IMCU).	0.90
crea	Serum creatinine levels measured at day 3.	0.74

4.2.4 Bone Marrow Transplant For Children Dataset

The bone marrow transplant children dataset aims to predict the success of marrow transplants for pediatric patients. It contains 187 instances and 36 features. This dataset is from the UCI Machine Learning Repository[21]. It is an open dataset resource. It describes who were subject to the unmanipulated allogeneic unrelated donor hematopoietic stem cell transplantation. This dataset is collected to help doctors improve the success rate of marrow transplants. It helps people understand the factors that contribute to the success of bone marrow transplant children. The dataset has some features with missing values. The number of missingness and ratio are counted. The number of datasets that have missingness in this dataset is twelve. The missingness mechanism is shown in Figure **??**. The feature 'extcGvHD' ,'CMVstatus','RecipientCMV' has the three largest missing ratios. The feature 'extcGvHD' has the largest ratio of missingness. It has a missing ratio of 16.5%. The feature 'CMVstatus', and 'RecipientCMV' has the missing ratio close to 7%. The feature 'CD3dCD34', 'CD3dkgx10d8', 'RecipientRh', 'DonorCMV', 'Rbodymass' has a lower missing ratio compared with 'extcGvHD','CMVstatus','RecipientCMV'. Their missingness ratio is in the range between 1% and 2%. The feature 'RecipientABO', 'ABOmatch', 'Antigen', 'Allele' has the least missing ratio. Their missing ratio is below 1%. They have the same missing ratio which is close to 0.5%.

Table 4.2: Analysis of missingness in the bone marrow transplant for children dataset.

Feature	Description	Missing ratio
extcGvHD	Development of extensive chronic graft versus host disease.	61.95
CMVstatus	Serological compatibility of the donor and the recipient of hematopoietic stem cells according to cytomegalovirus.	53.40
RecipientCMV	Presence of the Rh factor on recipients red blood cells ('+' - 1, '-' - 0).	49.42
CD3dCD34	CD3+ cell to CD34+ cell ratio.	47.80
CD34kgx10d6	CD34+ cell dose per kg of recipient body weight.	38.17
RecipientRh	Presence of the Rh factor on recipients red blood cells ('+' - 1, '-' - 0).	37.03
DonorCMV	resence of cytomegalovirus infection in the donor of hematopoietic stem cells prior to transplantation (present, absent).	32.75
Rbodymass	Body mass of the recipient of hematopoietic stem cells at the time of transplantation.	31.49
RecipientABO	ABO blood group of the recipient of hematopoietic stem cells (0 - 0, 1, A, B=-1, AB=2).	28.57
ABOmatch	Compatibility of the donor and the recipient of hematopoietic stem cells according to ABO blood group(matched - 1, mismatched - 1).	25.54
Antigen	In how many antigens there is difference beetwen the donor nad the recipient (-1 - no differences, 0 - one difference,1 (2) - two (three) differences).	25.09
Allele	In how many allele there is difference beetwen the donor nad therecipient {-1 no differences,0 - one difference, 1 (2) (3) - two, (tree, four) differences).	18.11

5

Experimental Results

The experiment section presents the experimental results of LASSOMR and DTMR. The LASSO and Decision Tree methods are used as the baseline methods. The synthetic dataset is used to search for the coefficient value change for the features that are missing in the LASSOMR and LASSO methods. It is anticipated that features with higher levels of missingness will incur great penalties compared to those with less missingness in the LASSOMR method. For the DTMR method, I demonstrate how the structure of the decision tree changes by applying different penalty terms λ. The DTMR method is inclined to select a decision node with no missing values when a large λ value is applied. With a smaller λ value, the DTMR method tends to select decision nodes characterized by significant feature importance. Then the three real-world datasets are used to measure the performance of the LASSOMR and DTMR models compared with the baseline models. The life expectancy dataset shows the model can maintain a great performance while mitigating reliance on features having missing values. The Support2 dataset and the Marrow Transplant for Children dataset illustrate that the LASSOMR and DTMR models reduce reliance on features with missing values compared to the traditional LASSO and Decision Tree methods. Although performance may be somewhat compromised, these findings offer valuable insights for further refinement in future research.

5.1 LASSOMR Experimental Results

For the LASSOMR method, the synthetic dataset with missing values is used to search for the coefficient change. The methods LASSOMR penalize the features more which have more missingness. It can be observed that when a feature has many missing values, it will be heavily penalized. When a feature has a few missing values, it will be penalized less. This aligns with the initial design when proposing the LASSOMR model.

5.1.1 Coefficient Change

The synthetic dataset is created as $y = \theta^T X + \epsilon$. Setting the coefficient θ can control the feature importance. Figure 5.1 sets feature 1 as the largest feature importance with 10% missingness. In LASSO, the coefficient value of feature 1 decreases to zero lastly since the coefficient decreases based on the feature importance without missing value influence. However, in the LASSOMR method, the coefficient change is different from the LASSO method. Since λ_j gives a penalty based on the missingness, with the increase of λ, it can be seen that the coefficient of feature 1 decreases to 0 first. It means that feature 1 with 10% missingness has been penalized to the greatest extent compared with other features. Even if feature 1 has the largest feature importance, it will be penalized and reduced to zero first.

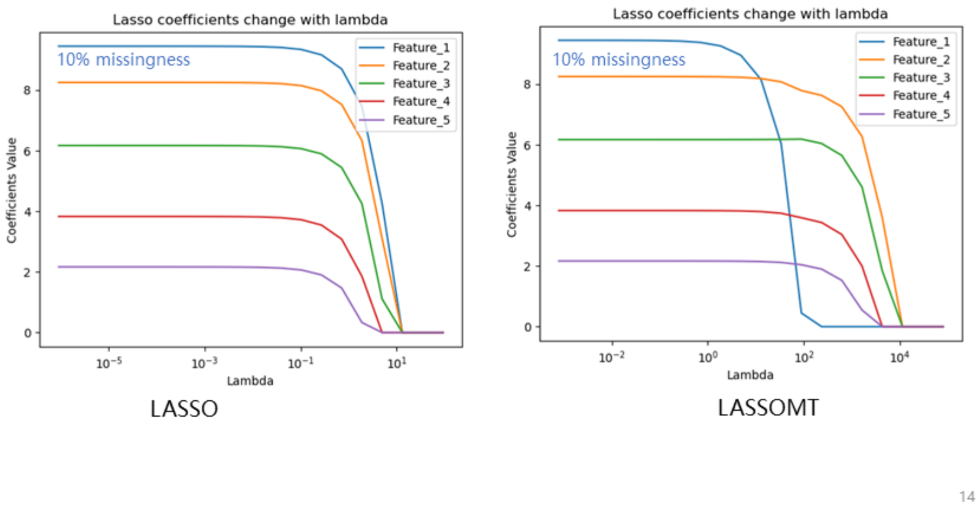


Figure 5.1: Coefficient change for a dataset having missingness in the LASSO model and LASSOMR model. Feature 1 has the largest feature importance with 10% missingness. This figure shows the coefficient change with the increase of λ value.

Then, the synthetic dataset creates five features with the same feature importance. It gives feature 1 10% missing ratio. Assigning a 20% missing ratio to feature 2, a 30% missing ratio to feature 3, gradually increasing to a 50% missing ratio for feature 5. Setting different missing rates of the feature from 10% to 50%, it aims to explore how the coefficients change for the features having different missing ratios as the lambda increases. In Figure 5.2, the dataset sets feature 1 10% missingness, feature 2 20% missingness, feature 3 30% missingness, feature 4 40% missingness, and feature 5 50% missingness.

When training the model using LASSO, the coefficient uniformly decreases to zero. Since the features have the same feature importance, it is penalized with the same penalty term. The LASSOMR method penalizes the feature based on the degree of missingness. With the increase of λ, it can be observed that the order of coefficient decreases to zero based on the degree of missingness. Feature 5 has the largest missing ratio. Therefore, it is penalized the most severely. Feature 1 has the fewest missing values, therefore it is penalized the least. The emergence of this result aligns with the initial thought of this method. The LASSOMR model penalizes the feature depending on the degree of missingness.

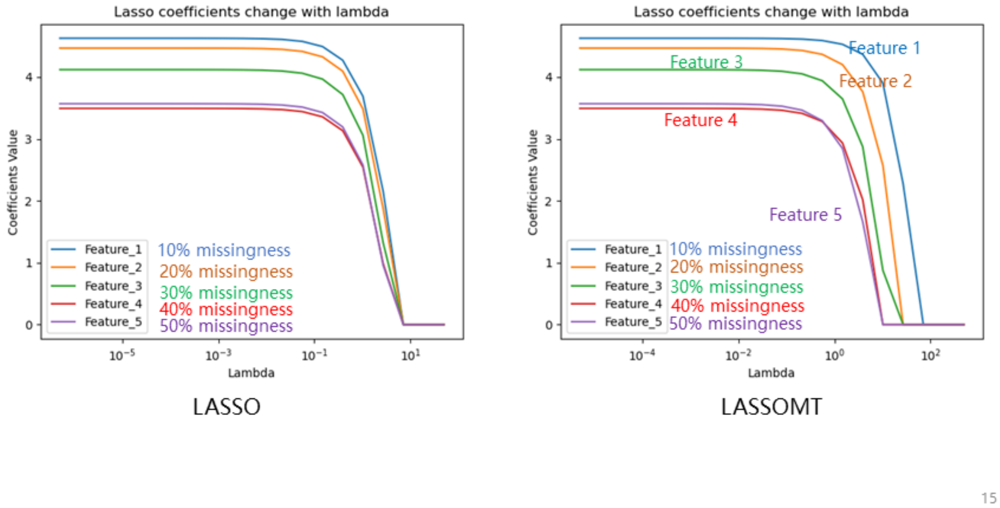


Figure 5.2: Coefficient change for a dataset having missingness in the LASSO model and LASSOMR model. All the feature has the same feature importance. From feature 1 to feature 5, the missingness ratio gradually decreases. This figure shows the coefficient change with the increase of λ value.

In the next step, set a correlation with two features. It is designed to explore how these two features change if one of them has missing values. In Figure 5.3, feature 1 and feature 2 exhibit a positive correlation. Introduce a 10% missingness in feature 1, while feature 2 remains complete. Given the positive correlation between feature 1 and feature 2, an increase in λ results in a penalization of the coefficient for feature 1, leading to a decrease in its value. As the value of the coefficient for feature 1 decreases, the coefficient in feature 2 increases at the same time. Feature 2 compensates for the contribution of feature 1 to the prediction results. This observation is important and inspiring. In real world dataset, if one feature exhibits missing values, another feature has less missing ratio can compensate for the contribution to the predicted values. As a result, the LASSOMR model achieves the goal of mitigating reliance on features with missing values.

In Figure 5.4, the dataset creates a negative relationship with feature 1 and feature 2, one of which has missingness. Feature 1 has a 10% missing ratio. When the coefficient value for feature 1 decreases with the increase of λ, the absolute value of the coefficient in feature 2 increases since it compensates for the contribution of feature 1 to the prediction result.

It can be concluded that when two features are correlated, the presence of missing data in one can be compensated by the other feature, thereby sustaining the integrity of the prediction. As a result, the LASSOMR model effectively reduces reliance on features with missing data by shifting dependency toward features that have no missing values.

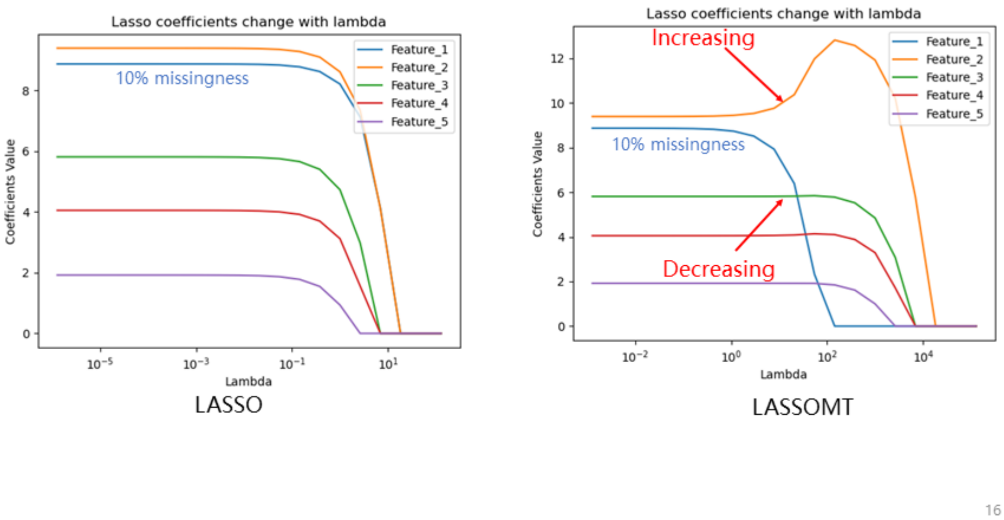


Figure 5.3: Coefficient change for a dataset having missingness in the LASSO model and LASSOMR model. Feature 1 has the largest feature importance with 10% missingness. Feature 2 has a positive correlation with feature 1. This figure shows the coefficient change with the increase of lambda value.

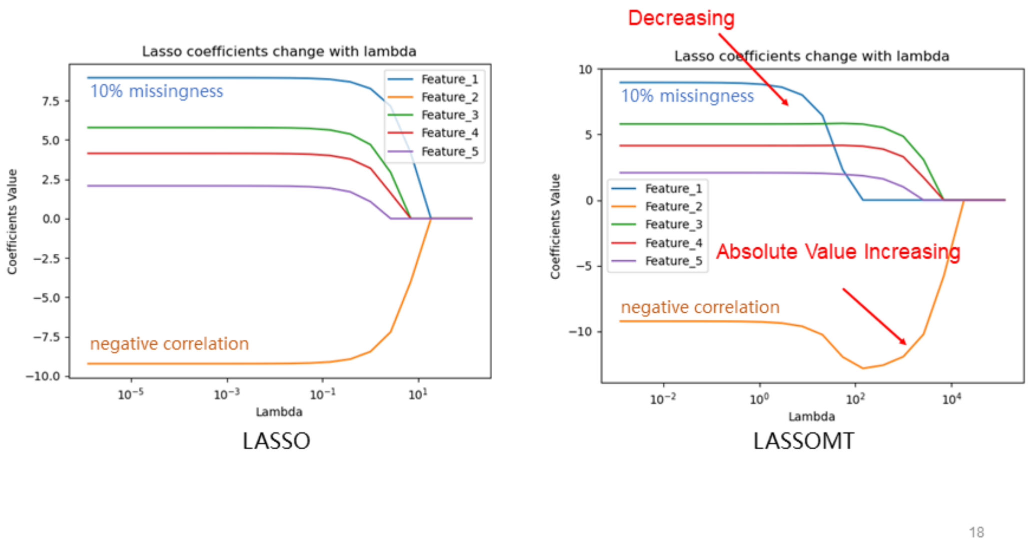


Figure 5.4: Coefficient change for a dataset having missingness in the LASSO model and LASSOMR model. Feature 1 has the largest feature importance with 10% missingness. Feature 2 has a negative correlation with feature 1. This figure shows the coefficient change with the increase of *lambda* value.

5.1.2 Real World Dataset

Three real-world datasets measure the performance and reliance between the LASSOMR and DTMR models. The baseline models are LASSO and Decision Tree methods. The life expectancy dataset is used in the regression task. Support2 and bone marrow transplant for children datasets are used in the classification task. In regression tasks, performance is assessed using the Mean Squared Error(MSE) and the coefficient of determination(R^2). In the classification task, the F1 score and AUC are utilized as the performance metrics.

5.1.2.1 Life Expectancy Dataset

The first dataset is the life expectancy dataset. For the LASSOMR model, the lambda value λ is adjusted to a small λ to 0, a suitable λ close to 0.1, and a large λ 1400 to mitigate reliance to zero. The suitable λ 0.1 is chosen since it has a lower reliance, a higher R^2, and a lower MSE value compared with the LASSO method. A large λ 1400 mitigates reliance to zero. The objective is to determine the performance of the LASSOMR model when all features are penalized to zero due to missingness.

Model	**Reliance (r)**	λ	$\mathbf{R^2}$	**MSE**
LASSOMR	0.043	0	0.79	18.89
LASSOMR	0.037	0.1	0.79	18.89
LASSOMR	0	1400	0.50	44.63
LASSO	0.038	0.01	0.78	18.95

Table 5.1: Performance between the LASSOMR and LASSO on life expectancy dataset. The LASSOMR model shows that with the increasing λ value, the reliance gradually decreases. By using cross-validation, choose the optimal λ for LASSO and use it to compare with the LASSOMR model.

According to Table 5.1, it can be seen that with the increase of λ, the reliance can decrease to zero gradually, meaning that the LASSOMR model mitigates reliance on features having missingness. When setting the λ to 0.1, LASSOMR has a less reliance value compared with the LASSO method, meaning that the model relies less on features with missing values. Comparing the performance with LASSO, it has a low MSE and a large R^2 meaning that it has a comparable or even better performance. Then, after increasing the λ to be a large value 1400, the reliance decreases to zero, the performance comprises that the MSE decreases to 44.63 and the R^2 becomes 0.5. For the life expectancy dataset, it is found that with the increasing of λ, the reliance on the LASSOMR model gradually decreases to zero. The LASSOMR has the function of mitigating reliance on features with missing values. Compared with the LASSO model, LASSOMR has a lower reliance. For the performance part, the LASSOMR model has a higher R^2 and lower MSE value meaning that the LASSOMR model has a comparable or even better performance.

Then the next step is to explore with the increase of λ, what is the coefficient value change in the life expectancy dataset. It can be seen that by increasing the value of λ, the value of the coefficient can be penalized based on the feature's missing ratio. In Figure 5.5, when setting the λ to be a really small value of 0.01, it is found that no features have been penalized to zero. When λ is set to a low value, features with missing ratios face minimal penalization.

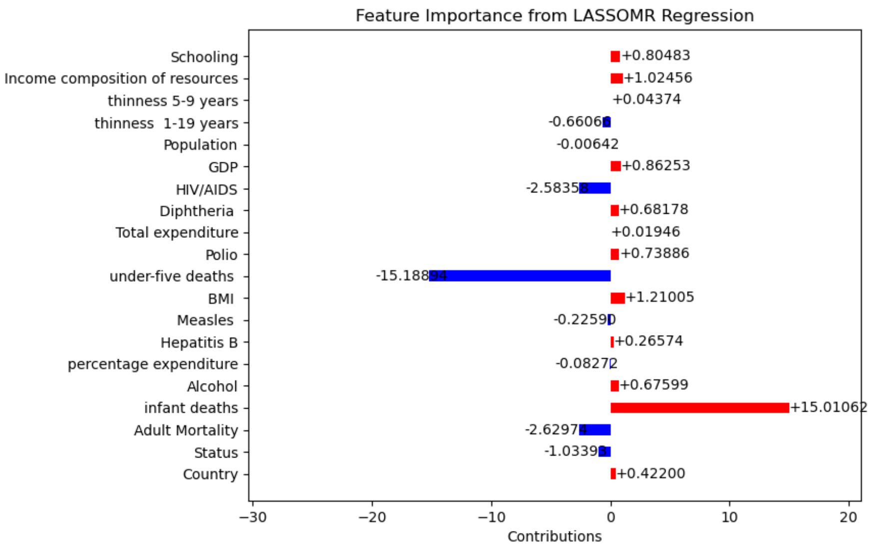


Figure 5.5: The coefficient value for each feature in the life expectancy dataset when setting the λ to 0.01.

In Figure 4.1, it is observed that the three features with the most missing ratios are 'Population', 'Hepatitis B', and 'GDP'. Then increase the value of λ to 0.1 to see the coefficient change for these three features. In Figure 5.6, it can be found that with the λ value increasing to 0.1, the coefficient value of feature 'Population' having the largest missingness has been reduced to zero. This fulfills the initial design target of the LASSOMR model, when increasing the value of λ, it penalizes most severely for the feature having the largest missingness and squeezes the value of its coefficient to zero.

Then increasing the λ to 0.99 in Figure 5.7, compared with Figure 5.6, it penalizes the value of the coefficient that has a missing ratio for features 'Population', 'Hepatitis B' to zero. When increasing the penalty term λ, it will penalize the features more severely. In this case, the feature that has the largest missingness will be penalized to zero.

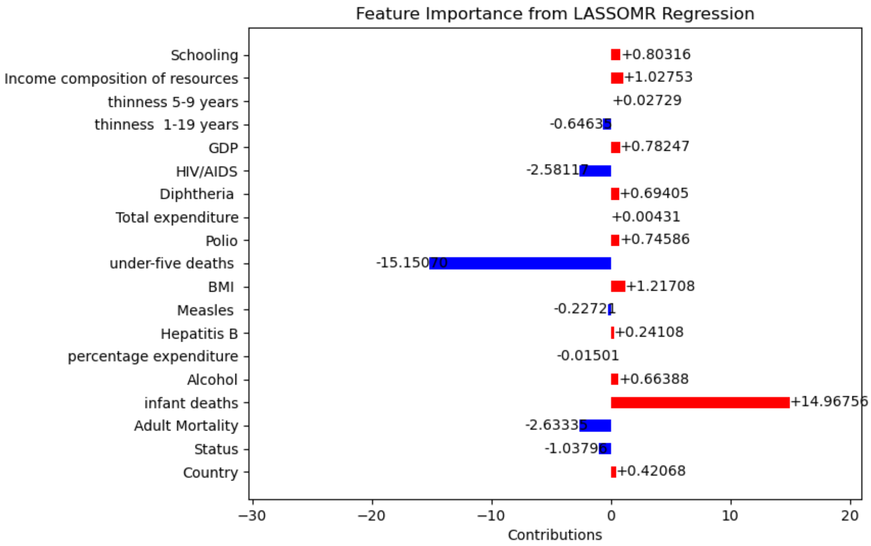


Figure 5.6: The coefficient value for each feature in the life expectancy dataset when setting the λ to 0.1. It aims to show the change of the coefficient value when increasing the λ.

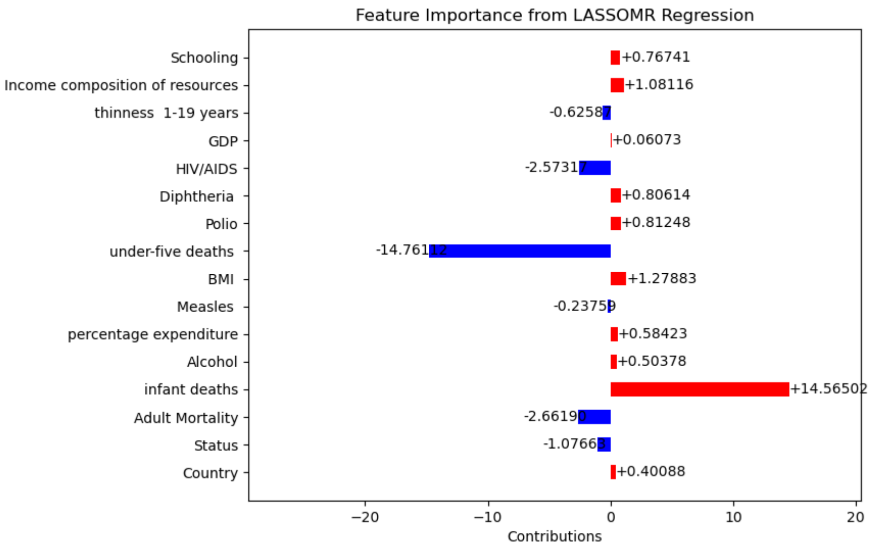


Figure 5.7: The coefficient value for each feature in the life expectancy dataset when setting the λ to 0.99.

When it increases the λ value to a large value of 1400 in Figure 5.8, the features having missingness will be penalized to zero. Only features without missingness remain.

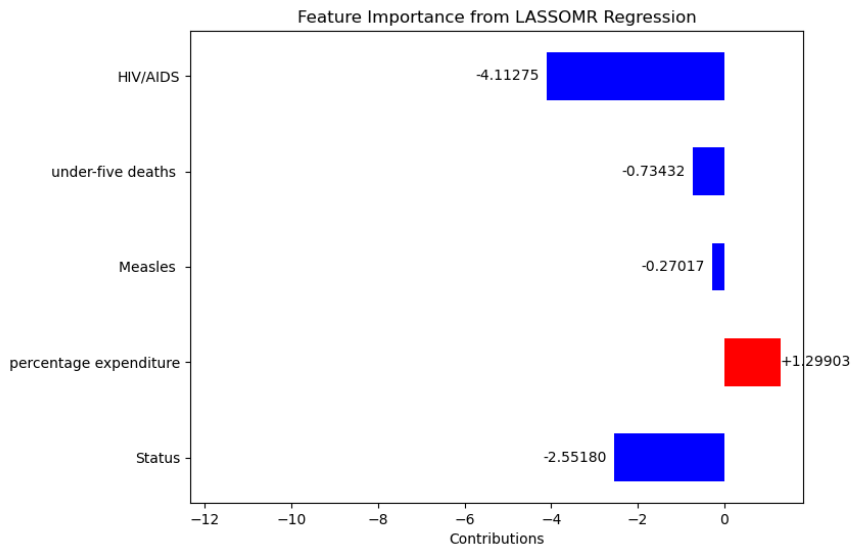


Figure 5.8: The coefficient value for each feature in the life expectancy dataset when setting the λ to 1400.

5.1.2.2 Support2 Dataset

The second dataset is from the UC Irvine Machine Learning Repository. Initially, train the dataset using the standard LASSO model. It uses cross-validation to pick the optimal value of λ. It picks the optimal value of λ 0.11 for the LASSO method. The reliance is calculated as 0.12. The AUC value is 0.78, and the F1 score is 0.72. By employing the LASSOMR method and adjusting the λ parameter to 100, reliance is significantly reduced to a minimal value. Even though the AUC value has a trade-off of 0.69, the F1 score remains consistent with that of the LASSO model. In this case, it is found that the DTMR method can mitigate reliance on features with missing values. The method's performance is relatively weak when compared to the results obtained from the life expectancy dataset. By analyzing the factors, the reason is that when a feature has a large feature importance and missingness, the method penalizes this feature severely. It decreases the contribution to the prediction which can lower the metric of AUC and F1 score. This phenomenon guides future work to better trade-off between the reliance and performance. In future research, the LASSOMR method could be enhanced by incorporating a standard penalty term λ. This modification would squeeze the penalty value gap between the features having missingness and the features that are complete. It allows for a more effective balance between reducing dependence on features with missing values and performance.

Then, after increasing the λ to be a large value 1400, the reliance decreases to zero, the performance comprises that the AUC decreases to 0.68 and the F1 score becomes 0.5. When all the features having missing values do not contribute to the final prediction, the performance will be influenced.

Model	Reliance (r)	λ	AUC	F1
LASSOMR	0.12	0	0.69	0.73
LASSOMR	0.00047	100	0.69	0.73
LASSOMR	0	10000	0.68	0.5
LASSO	0.11	0.12	0.78	0.72

Table 5.2: Performance between the LASSOMR and LASSO on support2 dataset. The LASSOMR model shows that with the increasing λ value, the reliance on the features having missing values gradually decreases. By using cross-validation, choose the optimal λ for LASSO and use it to compare with the LASSOMR model.

Then, using a figure of coefficient value to search with the increasing of λ, the change of coefficient value for the features having missingness. For missingness analysis in Table 4.1, it is found that the features'adlp',' urine', and 'glucose' have the largest missingness beyond 49%. The description of features can be known for Table 4.1. For the feature 'bun','totmcst' 'alb','income','adls', 'bili', 'pafi' 'ph' the missingness ratio is between 25% to 47%. When setting the λ value to be 101, for the Figure 5.9, it is found that it has penalized the largest missingness feature 'adlp' to zero. The coefficient value equal to zero will not be shown in this figure. Then continuing to increase the value of λ, it aims to search whether the model can mitigate reliance on features 'urine', 'glucose' to zero.

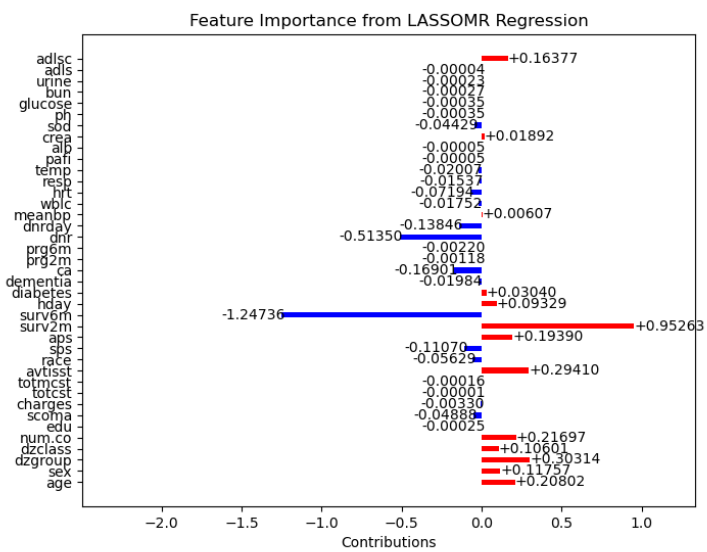


Figure 5.9: The coefficient value for each feature in the support2 dataset when setting the λ to 101.

When increasing the λ value to 625, it can be observed that the features 'adlp', 'urine', and 'glucose' have a large missing ratio, all of them have been penalized to zero. For the features 'totmcst', 'alb',' income','adls', 'bili', 'pafi', and 'ph',

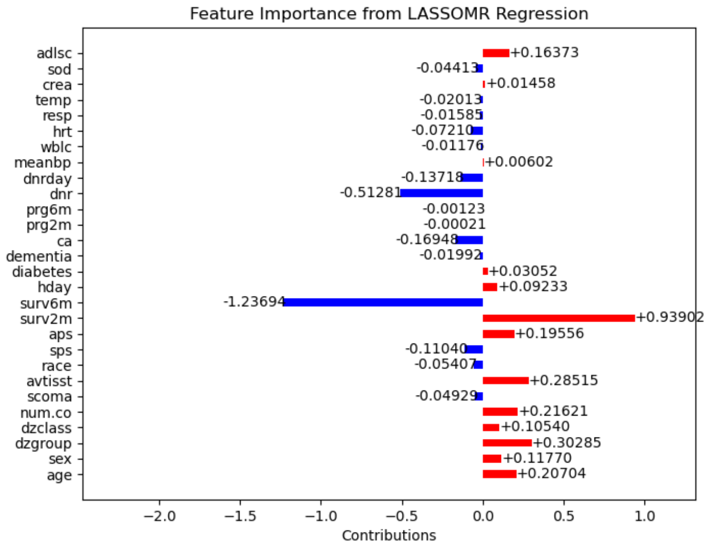


Figure 5.10: The coefficient value for each feature in support2 dataset when setting the λ to 625.

the reliance value is between 25% to 47%. Their values of the coefficient are also penalized as zero. Then continuing to increase the value of λ, it is found that it penalized all the coefficients for features having missingness to zero in Figure 5.11. In the support2 dataset, with the increasing of λ, the coefficient value with features having missing values has been penalized to zero gradually. When increasing the value of λ to be a large value, the LASSOMR model can penalize the coefficient value to zero for all the features having missingness.

5.1.2.3 Bone Marrow Transplant For Children Dataset

The third dataset is from the UC Irvine Machine Learning Repository which is the bone marrow transplant for children dataset. Initially, train the dataset using the standard LASSO model. Using cross-validation, it picks the optimal lambda value to 1.69. It gets the reliance to 0.008, the AUC value is 0.81, and the F1 score is 0.81. By using the LASSOMR method, when changing the λ to 100, it mitigates reliance to a really small value of 0.004. The AUC value has a trade-off of 0.69, the F1 score is reduced to 0.73. The dataset may exhibit relatively weak performance compared with the LASSO method because the number of observable instances is limited. Since it only has 137 instances for all datasets, a small number of instances may influence performance.

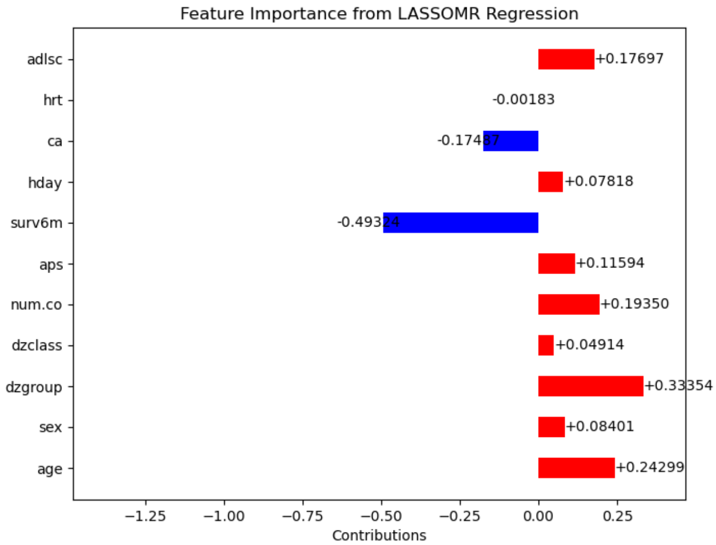


Figure 5.11: The coefficient value for each feature in support2 dataset when setting the λ to mitigate reliance to 0. For all the features having missing values, their coefficient value has been penalized to zero.

Model	Reliance (r)	λ	AUC	F1
LASSOMR	0.01	0	0.65	0.64
LASSOMR	0.004	100	0.69	0.73
LASSOMR	0	10000	0.68	0.5
LASSO	0.008	1.69	0.81	0.81

Table 5.3: Performance between the LASSOMR and LASSO on bone marrow transplant for children dataset. The LASSOMR model shows that with the increasing λ value, the reliance gradually decreases. By using cross-validation, choose the optimal λ for LASSO and use it to compare with the LASSOMR model.

Then continuing to search by increasing the value of λ, the change of the coefficient for each feature in bone marrow transplant for children dataset. When setting the value of λ to be 1, for Figure 5.12, it is found that when setting a λ to be a small value, most coefficients have not been penalized to zero. For Table 4.2, it can be found that 'extcGvHD' ,'CMVstatus','RecipientCMV' have the largest feature importance. Their missingness ratio is between 7% to 16%. Then increase the λ to see the coefficient change of each feature.

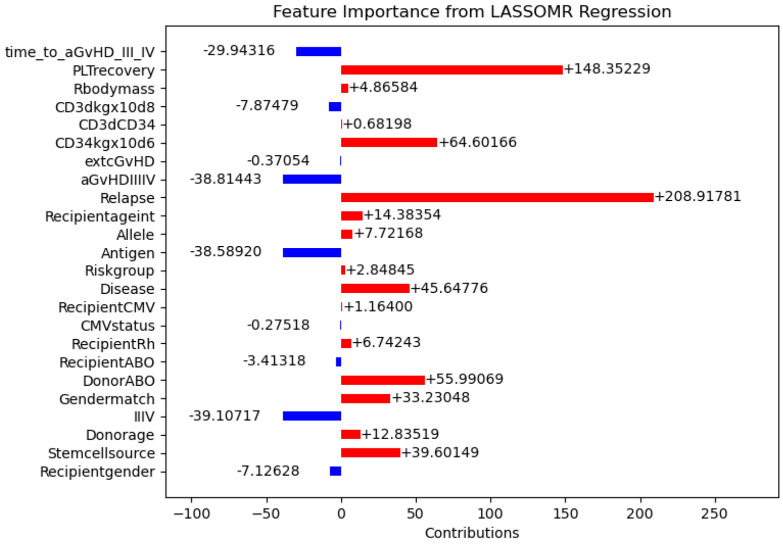


Figure 5.12: The coefficient value for each feature in bone marrow transplant for children dataset when setting the λ to 1.01.

When increasing the value of λ to 5, it can be observed that the feature 'CMVstatus' having the large feature missingness has been penalized to zero. In the bone marrow transplant for children dataset, it penalizes the feature 'CMVstatus' having the largest missing ratio to zero. The LASSOMR model penalizes more for the feature having more missing ratio.

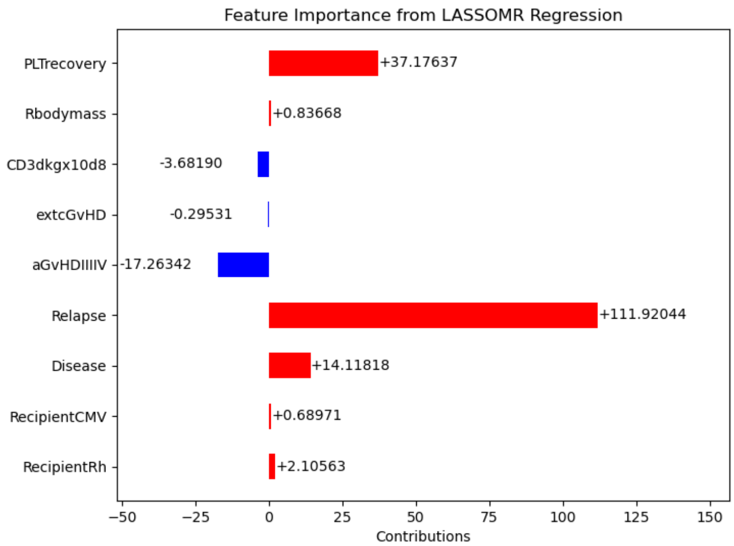


Figure 5.13: The coefficient value for each feature in bone marrow transplant for children dataset when setting the λ to 5.

5.2 DTMR Experimental Results

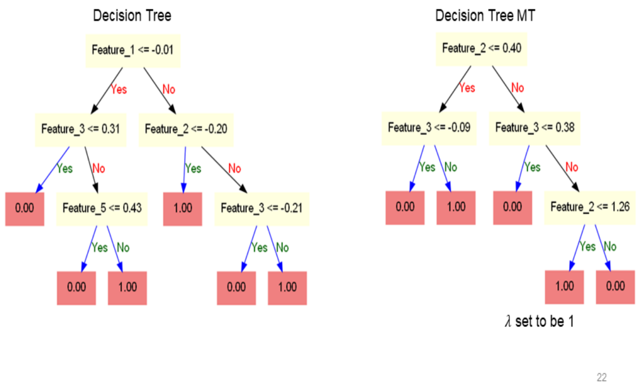
For the DTMR method, the synthetic dataset with missing values is used to figure out the tree change between the Decision Tree and the DTMR method. It is expected to be seen that the DTMR method can trade off between reliance and feature importance when building the decision tree. Consistent with the expectation, when setting the penalty term to be a large value, considering a decision node is mainly based on the missingness. When setting the penalty term to a middle value, it has a trade-off between the feature importance and missingness. When setting the penalty term to be a really small value, it almost only considers the feature importance as the decision node.

5.2.1 Decision Tree Change

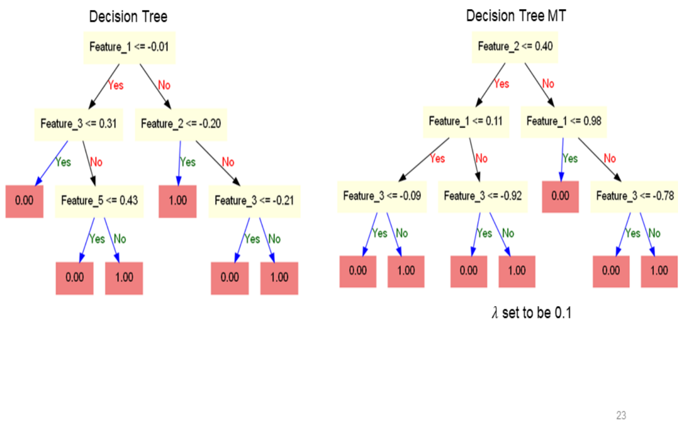
The synthetic dataset is used to explore the change of λ leading to the change of the decision tree. It creates dataset as $Probabilities = sigmoid(bx + \epsilon)$, $Target = (probabilities >= 0.5).astype(int)$. Assigning the highest feature importance to feature 1, the second highest to feature 2, and so forth, with feature 5 having the least feature importance. Set feature 1 10% percent missing ratio.

In Figure 5.14, it can be observed setting the value of λ to a different value has an impact on decision trees. First, setting the λ to be a large value equal to 1 penalizes severely for a feature having missingness. Observing the Figure 5.14 in part (a), the Decision Tree method has the root node 'Feature 1 < 0.01'. It illustrates that feature 1 has the strongest ability to assign the dataset to a specific class. Using the DTMR method to train on the dataset, it can be found that feature 1 with the largest feature importance will not be chosen as the decision node. The reason is that when calculating the information gain using Formula 3.6, feature 1 has been penalized severely to reduce the information gain. In this case, when setting the λ to be a large value, the most important factor in choosing a candidate node is whether the features have missingness.

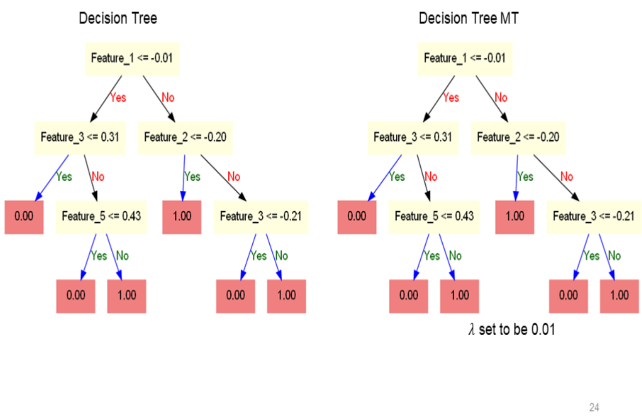
When set λ to a middle value of 0.1, feature 1 is in the second layer using the DTMR method. In this case, setting the value of λ to 0.1 has a trade-off between the feature importance and reliance on features with a missing value. When setting λ to be a really small value of 0.01 in part c, DTMR has built a tree the same as the Decision Tree model. It means that when using Formula 3.6 to calculate information gain, the penalty value in the decision tree is really small, it will not influence the value of information gain. In this case, the DTMR method and the Decision Tree will build the same decision tree. This example illustrates that by setting the different values of the λ, DTMR builds the different decision trees. When constructing decision trees, setting λ value to be a large value, the DTMR method tends to favor decision nodes with no missingness. When setting the λ value to be a small value, the DTMR method prefers to choose the feature having the maximum ability to reduce the entropy, the largest feature importance. Controlling the value of the penalty term has a trade-off between the feature importance and reliance on features with missing value.



(a) Build the decision tree using the Decision Tree and DTMR method. For DTMR method, setting the λ value to be 1.



(b) Build the decision tree using the Decision Tree and DTMR method. For the DTMR method, setting the λ value to be 0.1.



(c) Build the decision tree using the Decision Tree and DTMR method. For the DTMR method, setting the λ value to be 0.01.

Figure 5.14: Setting the λ value to be 1, 0.1, and 0.01, draw 3 different decision trees in DTMR compared with Decision Tree.

5.2.2 Real World Dataset

Three real-world datasets are employed to evaluate the performance and compare the reliance of the DTMR model with the baseline models. The life expectancy dataset is utilized to evaluate regression tasks, while the support2 and bone marrow transplant for children datasets are employed for classification tasks. MSE and R^2 are used to measure the performance in regression tasks. AUC score and $F1$ score are used to measure the performance in classification tasks. Reliance is a metric that can be compared with the reliance on features with missing values.

5.2.2.1 Life Expectancy Dataset

DTMR and Decision Tree(DT) models are used to train the model for the regression task. When the $\lambda = 0$, according to Formula 3.6, the penalty term is zero indicating that node selection is based on the maximum reduction of the entropy, consistent with the traditional Decision Tree model. Cross-validation is implemented for the hyperparameter, selecting a maximum depth of 7. As shown in Table 5.4, when the $\lambda = 0$, the model exhibits the reliance to 0.013. The MSE value is 7.83 and R^2 is 0.913

Increasing the λ to 0.1 has a lower reliance compared with the DT method. Also, it has a lower MSE and higher R^2 value. The DTMR method, when compared with the DT approach, demonstrates superior performance: it achieves a lower MSE and a higher R^2 value, showing a better performance.

Furthermore, setting the λ to a higher value further reduces reliance, minimizing it significantly to 0.0039, while maintaining an MSE of 7.94 and an R^2 of 0.91. It demonstrates that the DTMR method mitigates reliance on a small value while sustaining excellent performance.

Model	**Reliance (r)**	λ	**MSE**	R^2
DT	0.013	0.0	7.83	0.913
DTMR	0.010	0.1	7.69	0.915
DTMR	0.0039	1.9	7.94	0.911

Table 5.4: The performance between the DTMR and DT in life expectancy dataset. Setting λ value to be 0.1 to get a lower reliance than DT method, and have a higher R^2. Setting the λ value to be 1.9 to reduce the reliance to a really small value and get an MSE value not so large and R^2 not so low.

5.2.2.2 Support2 Dataset

The support2 dataset is used for classification tasks. Use cross-validation to tune the parameter for max depth in the Decision Tree model. The maximum depth is 7. The reliance is 0.086, and the AUC and F1 scores will be 0.69 and 0.75. Then when the DTMR increases the λ to 0.1, it can mitigate reliance to 0.018. Observing the value of the AUC and F1 score, it can maintain a comparable performance compared with DT. Then when changing the λ to 1.7, the reliance becomes a really small value close to zero, and the AUC and F1 Score can still have a great performance. when setting the λ to be a really large value, it can mitigate reliance on features and reduce the reliance to a really small value.

Model	Reliance (r)	λ	AUC	F1 Score
DT	0.086	0	0.69	0.75
DTMR	0.018	0.1	0.69	0.74
DTMR	0.0035	1.7	0.69	0.74

Table 5.5: The performance between the DTMR and DT in the support 2 dataset. Setting λ value to be 0.1 to get the same AUC value as the DT method, and have a comparable F1 score. Setting the λ value to be 1.7 to reduce the reliance to a really small value and get the same AUC value R^2 not so low.

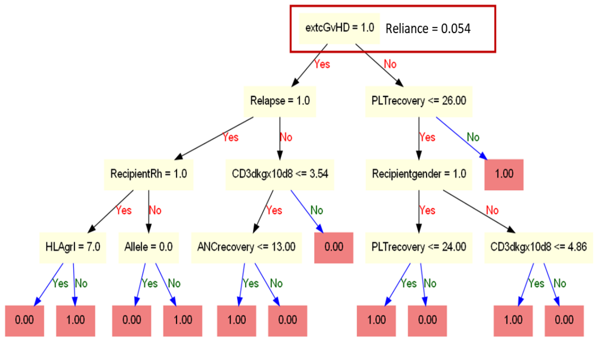
5.2.2.3 Bone Marrow Transplant For Children Dataset

The third dataset is the marrow transplant for children dataset. By using the decision tree method, the reliance is 0.039. The AUC and F1 scores are 0.75 and 0.76. When changing the λ to 0.3, it mitigates reliance to 0.021, and the AUC score is 0.73 and 0.74. It decreases reliance to a lower value compared with the Decision Tree method even though the AUC and F1 score is lower compared with the Decision Tree method. The F1 score and AUC value are still close to the DT method. Then when continuing to increase the λ to 0.6, the reliance becomes 0. The F1 score and AUC values are 0.61 and 0.60. In this case, the DTMR model chooses all of the decision nodes having no missing values. To better improve the performance, it is inspired to combine penalty thought with the Random Forest in future work. Since the Random Forest algorithm creates more decision trees. Some of the trees focus on performance while others mitigate reliance on features with missing values. By integrating all of the results together, it is anticipated to be a better trade-off between reliance and performance.

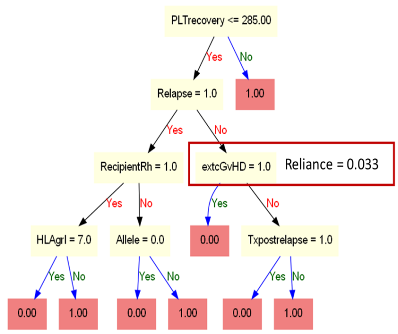
Model	Reliance (r)	λ	AUC	F1 Score
DT	0.039	0.0	0.75	0.76
DTMR	0.021	0.3	0.73	0.74
DTMR	0	0.6	0.60	0.61

Table 5.6: The performance between the DTMR and DT in the bone marrow transplant children dataset. Setting λ value to be 0.3 to get the AUC value close to DT method, and have an F1 score close to DT. Setting the λ value to be 0.6 to reduce the reliance to 0.

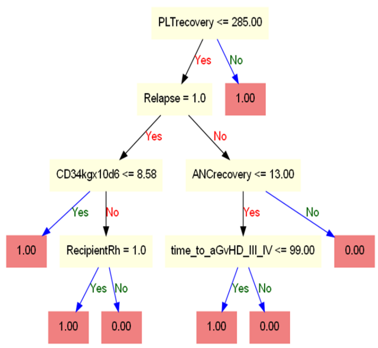
This part searches for with the increasing λ value, what is the change of the decision tree in bone marrow transplant for children dataset. Figure 5.15 shows that when using the decision tree to train the model, the feature 'extcGvHd' is the root node because it has the largest feature importance. Even if it has missingness, it does not get penalized. The reliance for this node is 0.054. When increasing the λ to a large value of 0.3, since the feature 'extcGvHd' has a really large value of missing ratio, even if it still has the large feature importance, it will not be chosen as the decision node in the first layer. It holds the position of the third layer, and the reliance of this node is reduced to 0.033. From this change, it can be seen that the penalty term mitigates reliance on features with missingness. Then increasing the λ to be a large value to 0.6, and the reliance on this feature goes down to 0. Since it has a really harsh penalty for the feature having missingness. DTMR model builds the tree choosing the features having no missingness. This change in the decision tree process clearly illustrates the increasing λ value leads to the change of the decision tree.



(a) DT Method for the bone marrow transplant children dataset.



(b) DTMR method for the marrow transplant children dataset setting λ to 0.3.



(c) DTMR method for the marrow transplant children set λ to be 0.6.

Figure 5.15: Build three different trees in bone marrow transplant children dataset. The method that is used is the DT method, the DTMR method sets λ to be 0.3, and the DTMR method sets the λ to be 0.6.

6

Discussion

6.1 Experimental Results

This thesis aims to solve the limitation of previous methods that have dealt with datasets having missing values. Previous methods have limitations: they either lack interpretability or rely on features with missing values. By modifying the LASSO and Decision Tree methods, I create the LASSOMR method and the DTMR method to overcome these limitations. The methods that I modified rely less on features with missing values and have high interpretability. The synthetic dataset demonstrates that the LASSOMR method imposes more severe penalties on features as their missingness increases. The experiment results display that with the increasing of penalty term λ, the coefficient value for the features having more missingness decreases to zero earlier compared with the features having less missingness. The features with more missingness contribute less to the prediction. Therefore, the LASSOMR model mitigates reliance on features with missing values. For the DTMR method, the synthetic dataset shows that when giving a large penalty λ, it prioritizes that the decision nodes have no missing values. When giving a small penalty λ, the DTMR selects decision nodes by considering the feature importance first. By setting different values of λ, the DTMR model can trade-off between the performance and reliance on features having missing values. The real-world dataset is used to measure the performance between the LASSOMR, DTMR method, and LASSO, Decision Tree method. The life expectancy dataset shows that the LASSOMR method and DTMR method can mitigate reliance on the missing features and maintain better performance than baseline models. Support2 and bone marrow transplant for children dataset shows that the LASSOMR method and DTMR method can mitigate reliance compared with the LASSO and Decision Tree method. Even if its performance trades off, this experience guides future work that can better improve the performance. Observing the coefficient value of real-world datasets shows that the LASSOMR model penalizes the features more by decreasing the value of the coefficient. It aims to reduce the contribution of the features that have large missingness. The DTMR model shows that when using the DTMR model in real-world datasets, the features with missing values are often not selected or are located at a position where fewer datasets would arrive. As a result, the decision tree built by the DTMR model makes decisions relying less on features with missing values. In summary, the LASSOMR model and DTMR model have an effect that mitigates reliance on features having missing values with high interpretability whether in synthetic datasets or real-world datasets.

6.2 Limitation

In this study, one limitation is that the LASSOMR method penalizes the feature with severe missingness. In Formula 3.2, the LASSOMR method penalizes the feature with missingness based on the value of the coefficient. According to the value of λ_j in Formula 3.1 if one feature does not have missingness, the value of penalty term is $\frac{1}{n}\lambda$, a really small value. When one feature exhibits substantial missingness, the penalty term correspondingly increases in value. The limitation of this is that if one feature has a really large missing ratio, even if it also has a large feature importance to the prediction, it will be penalized severely. It reduces the contribution of the result. The LASSOMR method may penalize this feature severely, which can influence the performance of this model.

The second limitation is the reliance definition in Formula 3.4. It means that only the coefficient value for the features having missingness equals zero, meaning that it does not have reliance on this feature. However, when the coefficient value has been penalized for squeezing to a small value but not equal to zero, in this case, it contributes less to the prediction of the result. The model mitigates reliance on this feature having missing values. However, the definition of reliance does not reflect this result directly.

6.3 Future Work

This research has demonstrated the potential of using machine learning algorithms to mitigate reliance on the features with missingness using interpretable model. Several directions for future work could further enhance the effectiveness and applicability of my findings.

Future studies should trade-off between the reliance and performance in the LASSOMR method, meaning that it can mitigate reliance and at the same time have a comparable or even better performance with the baseline models. In the LASSOMR method, I only modify the penalty term to λ_j mitigating reliance on features having missing value. In future studies, it can add another penalty term λ with the missingness penalty term λ_j. By adding another penalty term λ, it penalizes $\lambda + \lambda_j$ to all the features. For the feature that does not have missingness, it gives the penalty term λ. In this way, it penalizes more to the feature having more missingness. However, the penalty term value gap will decrease between the feature having missingness and the feature having no missingness aiming for a trade-off between reliance and performance.

Future work should also consider how to modify the definition of reliance in the linear model. In this thesis, only the value of the coefficient equal to zero will be considered to have no reliance on the feature having a missing value. However, when the value of the coefficient decreases, even if the contribution to the prediction of this feature becomes less, it will not reflect the value of reliance changes. In future research, the concept of reliance can be modified to better reflect the contribution of variables having missing values.

In future work research, it can be considered to modify the Random Forest algorithm by adding the penalty term into it. Random forest algorithms build many decision trees. To mitigate reliance on the Random Forest algorithm, it can be improved by building many decision trees by giving different penalty terms. Then the trees will have a preference to select a node with large feature importance or choose a node that has no missingness. Then combine the results from different trees to get the final result. Since the Random Forest algorithm can build more trees, it is expected to have a better trade-off between reliance and performance.

7
Conclusion

This thesis proposes two methods that can mitigate reliance on features with missingness when making a prediction. The LASSOMR method and DTMR method penalize the features based on their missingness. When running two models in a synthetic dataset and real world datasets, it is shown that the coefficient changes fulfilling the expectation that features with more missing ratio are penalized more compared with the features having less missingness. As a result, the LASSOMR model and DTMR model mitigate reliance on features with missing values.

Three real world datasets are reformulated to determine whether two models can mitigate reliance on features with missing values and have a comparable performance compared with the baseline models. The result of the life expectancy dataset shows that the LASSOMR model can not only mitigate reliance on features having missing values but also have a comparable performance with the LASSO model. The support2 dataset and bone marrow transplant for children dataset also show that with the increasing λ value, the LASSOMR model mitigates reliance gradually to zero. It also paves the way for future research to integrate a standard penalty term λ, alongside the existing penalty term λ_j, into the model. From the theoretical aspect, combining a normal penalty term λ with a missingness penalty term λ_j can give the model a better trade-off between performance and reliance.

In the DTMR model, a penalty term is incorporated during the calculation of information gain to account for features that depend on missingness. When selecting a feature as a decision node, the model not only considers the feature importance but also its reliance on missingness. The life expectancy dataset and support2 dataset show that the DTMR model has a lower reliance and a great performance compared with the Decision Tree method. For the marrow transplant for children dataset, the size of the dataset for observation is small, so the performance is not that comparable with the Decision Tree method. It guides future work that the thought that penalizes the feature with missingness can be extended to the Random Forest method. Random Forest can build more decision trees. Some of the trees have a better performance, and some of the trees mitigate reliance on features with missing values. It combines the results from different trees to get the final result. There is expected to be a better trade-off between reliance and performance.

Bibliography

[1] Dennis Wei et al. "Generalized linear rule models". In: *International conference on machine learning*. PMLR. 2019, pp. 6687–6696.

[2] J. Fürnkranz, D. Gamberger, and N. Lavrač. *Foundations of Rule Learning*. Cognitive Technologies. Springer Berlin Heidelberg, 2012. ISBN: 9783540751977. URL: `https://books.google.se/books?id=JsTXnjWexCEC`.

[3] Roderick J. A. Little and Donald B. Rubin. *Statistical Analysis with Missing Data*. 3rd ed. Hoboken, NJ: Wiley, 2019.

[4] Donald B Rubin. "Inference and missing data". In: *Biometrika* 63.3 (1976), pp. 581–592.

[5] Marine Le Morvan et al. "NeuMiss networks: differentiable programming for supervised learning with missing values." In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 5980–5990. URL: `https://proceedings.neurips.cc/paper_files/paper/2020/file/42ae1544956fbe6e09242e6cd752444c-Paper.pdf`.

[6] Shaun Seaman et al. "What is meant by "missing at random"?" In: (2013).

[7] Stef van Buuren. *Flexible Imputation of Missing Data*. CRC Press, 2012.

[8] Geoffrey I Webb, Eamonn Keogh, and Risto Miikkulainen. "Naıve Bayes." In: *Encyclopedia of machine learning* 15.1 (2010), pp. 713–714.

[9] Bheki ETH Twala, MC Jones, and David J Hand. "Good methods for coping with missing data in decision trees". In: *Pattern Recognition Letters* 29.7 (2008), pp. 950–956.

[10] Tianqi Chen and Carlos Guestrin. "Xgboost: A scalable tree boosting system". In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 2016, pp. 785–794.

[11] Joseph G Ibrahim et al. "Missing-data methods for generalized linear models: A comparative review". In: *Journal of the American Statistical Association* 100.469 (2005), pp. 332–346.

[12] Marine Le Morvan et al. "Linear predictor on linearly-generated data with missing values: non consistency and solutions". In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by Silvia Chiappa and Roberto Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, 26–28 Aug 2020, pp. 3165–3174. URL: `https://proceedings.mlr.press/v108/morvan20a.html`.

[13] Teresa Alves de Sousa and Imke Mayer. *How to Generate Missing Values?* Accessed: insert-access-date-here. 2021.4. URL: `https://rmisstastic.netlify.app/how-to/generate/misssimul`.

[14] R.J.A. Little and D.B. Rubin. *Statistical Analysis with Missing Data*. Wiley Series in Probability and Statistics. Wiley, 2019. ISBN: 9780470526798. URL: `https://books.google.se/books?id=BemMDwAAQBAJ`.

[15] Donald B Rubin. *Multiple imputation for survey nonresponse*. 1987.

[16] Ian R White, Patrick Royston, and Angela M Wood. "Multiple imputation using chained equations: issues and guidance for practice". In: *Statistics in medicine* 30.4 (2011), pp. 377–399.

[17] Robert Tibshirani. "Regression shrinkage and selection via the lasso". In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 58.1 (1996), pp. 267–288.

[18] Leo Breiman. *Classification and regression trees*. Routledge, 2017.

[19] J. Ross Quinlan. "Induction of decision trees". In: *Machine learning* 1 (1986), pp. 81–106.

[20] Frank Harrell. *Dataset from Vanderbilt University Department of Biostatistics*. `https://hbiostat.org/data/`. DOI: https://doi.org/10.24432/C5NP6Z. 2022.

[21] Marek Sikora, Lukasz Wrobel, and Adam Gudys. *Dataset from Vanderbilt University Department of Biostatistics*. `https://hbiostat.org/data/`. DOI: `https://doi.org/10.24432/C5NP6Z`. 2022.