Norwegian University
of Life Sciences

Master's Thesis 2021 30 ECTS
Faculty of Science and Technology

Multi block Analysis of Gastrointestinal Neuroendocrine Tumors Data Using Response Oriented Sequential Alternation (ROSA)

Ghazal Azadi
Master of science in data science

Preface

This thesis is a final work as partial fulfilment for the degree of Master of Science in Faculty of Science and Technology, Norwegian University of Life Sciences. I worked on this study with pleasure and enjoyed carrying out a small research myself.

Firstly, I would like to express my gratitude towards my supervisors Kristian Hovde Liland and Oliver Tomic, Faculty of Science and Technology (REALTEK), NMBU for their perfect guidance and support. I was blessed being supervised by knowledgeable and experienced supervisors who devote their time for guiding us whenever that was needed. I should also thank them teaching us so many applicable concepts through the courses.

Lastly, truth to be told, I could not have achieved my current level of success without the support I received from my family and beloved ones. I want to deeply appreciate their support, understanding and encouragement.

I hope readers enjoy reading this work as much as I did while carrying it out.

Ås, 19th May, 2021

Ghazal Azadi

Abstract

Gastrointestinal neuroendocrine tumours (NETs) are slow-growing tumours. In this type of cancer, survival rate is an important factor. The current study considers the number of survival days as the target variable and tries to spot important features impacting this variable.

Applying preprocessing steps, the dataset was prepared to be used in the machine learning algorithms. Moreover to that, using Repeated Elastic Net Technique (RENT), some of the relatively important features were selected and our relatively wide dataset with high number of features and low number of samples changed into a more stable dataset. However since we wanted to select the features based on a model which was relatively reliable in terms of error (RMSEP) and R^2, we examined three different complementary approaches. In the first approach, we considered our full dataset without any missing items. However RENT models selected features based on average R^2 of -47% and -40% for the first and second block, respectively. In the second approach, we include two more features which caused our dataset to lose 9 samples, since these features include 9 missing items. However this change helped our RENT models' R^2's to experience improvements until 20% and -36%. In the last approach, we excluded some samples causing too much noise. Moreover to that, consulting with experts, we decided to remove some features which we already knew are not important and lastly having a Box-Cox transformation of the target we started working with a normalised response vector which had symmetric distribution. This approach helped us achieving average R^2's of 34% and 21% for the first and second block respectively.

In the last step, multi block method of ROSA (Response Oriented Sequential Alternation) was applied to analyse our dataset obtained from the last steps. Modelling our problem with ROSA, this method gave us an acceptable R^2 of 74% on the cross validated data. ROSA also helped us ordering the features based on their importances.

KEYWORDS: Box-Cox, Cross validated data, Repeated Elastic Net Technique (RENT), Response Oriented Sequential Alternation (ROSA)

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

Gastrointestinal neuroendocrine tumours (NETs) are slow-growing tumours with distinct histological, biological, and clinical characteristics that have increased in incidence during the last decades. [1] This is mostly due to improvements of diagnosis (specially diagnosis of neuroendocrine), including better endoscopy and CT scans and of course better awareness about the tumours. [2] Based on the statistics, around 8000 people in the United States are diagnosed with this type of cancer each year. The most common organs of body that these types of tumours can be produced are small intestine and rectum. It has been studied that around 94% of diagnosed people live at least 5 years after the tumour is found. If we consider our study group as people who do not experience any metastasis and the tumour does not spread in their body, the 5-year survival rate would increase up-to 97% . If the tumour spread to nearby nodes, the percentage decreases to 95% and if the metastasis occurs around distant areas in the body, the survival rate decreases to 67%. This issue proves the importance of survival rate in this type of cancer. The current study also considers the number of survival days as the target. The aim is to find the important features which cause this number to be relatively high.

1.2 Structure of thesis

This study is mainly divided into 4 sections: theory, materials, methods and results. In the theory section (section 2) we describe the theory behind the main multi block modelling method (so-called response oriented sequential alternation- ROSA) and

[1] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7443843/

[2] https://www.cancer.net/cancer-types/neuroendocrine-tumor-gastrointestinal-tract/statistics

all the concepts needed to know in advance to understand this technique. Afterwards, we have a quick review on the dataset and how we are going to make the most use of our samples by implementing some validation techniques followed by an explanation about tasks to have a response variable with more symmetric distribution is also mentioned in this section. In the materials chapter, (section 3) we make a very detailed explanation about the data and features we have. For every single variable in the dataset we have an interpretation, so in the next chapters whenever a feature has been mentioned, its explanation is referred to this chapter. Next chapter (chapter 4) is methods section. In this section we explain all the preprocessing methods used in this thesis along with the repeated elastic net feature selection technique. Preprocessing steps include feature filtering, feature transformation, handling missing values and identifying outliers. The last but not least chapter is the results part. (section 5). In this section we implement all the theories and methods explained in the previous chapters on our dataset and present the results of our findings in this section. It should be noted that the order of preprocessing in this section is the same as order we had in methods section. However, the last step of the multi block analysis of ROSA on our data is presented in the last part of this section.

The last two chapters of this study is about conclusions and future potentials of this research.

Chapter 2

Theory

Machine learning is based on finding a model which fits to the new data based on the information we already had from historical data.[1] In other words, machine learning includes some automatic computation procedures which try to learn form previous examples. [2] Generally, supervised machine learning problems are either classification or regression problems. In supervised machine learning problems, we try to make a model of features in terms of labeled target. [3] In other words, we already know which variable is the target and which are the features. [3] The type of the supervised case is specified based on the target type. If we deal with categorical target variable, our problem is supervised classification and if we have continuous target variable the problem is addressed by supervised regression machine learning techniques. [1] In the beginning of chapter 5 we will explain how changing the problem from classification to regression helped us having more accurate results in our problem.

As been mentioned, this study uses Response Oriented Sequential Alternation (ROSA) to analyse the multi block dataset we have. Therefore in this chapter through the subsequent sections, we will firstly introduce Partial Least Square Regression (PLSR) technique which is a prerequisite for understanding the concept of multi block methods. After that we will have a quick review on some of available multi block methods followed by a detailed explanation about ROSA. In the last section of this chapter, we will discuss about making the most use of the samples when our sample size is low, accompanied by an explanation about uneven target distribution and the solution to tackle this issue.

2.1 Partial Least Square Regression (PLSR)

PLSR is a multivariate statistical technique which is used in situations where we aim to model one or multiple response variables response variables to multiple regressors. [4] PLSR is an improved version of PCR (principal component regression). We first explain the PCR technique and then will show how PLSR improved the PCR.

One of the main problems with multiple liner regression (MLR) was that it could not be used in the cases when number of samples were lower than number of features, so-called wide datasets. [5] Therefore the very first solution which comes to mind is to make the problem in a way that it has lower number of features with the same amount of information in them. Dimension reduction is the main purpose of finding principal components. In other words, instead of using ordinary features, we will use principal components obtained by the orthogonal scores. These components have lower dimension and since they are orthogonal as well, the multicollinearity problem which is also very common in MLR cases would be tackled. As a reminder, scores are a low-dimensional representation of the observations, while loadings are the coordinates of the features when projected onto the scores.[5] Figure 2.1 shows what has been discussed so far. (Assuming **X** as the vector of features with N samples and K variables. Y as the target and **T** as the orthogonal scores)

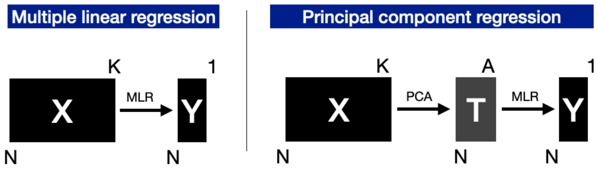

Figure 2.1: *PCR performance visualisation adapted from [5]*

In other words, instead of using the $\mathbf{X}_{N\times K}$ matrix of features, we use lower dimensional $\mathbf{T}_{N\times A}$ matrix of orthogonalised scores to model it on the target. [5] Mathematical expression of the PCR steps is as follows where **T** and **P** are vectors of scores and loadings respectively. [5]

1. $\mathbf{T} = \mathbf{XP}$
2. $\hat{Y} = \mathbf{Tb}$ and can be solved as $\mathbf{b} = (\mathbf{T}'\mathbf{T})^{-1}\mathbf{T}'Y$

Explaining PLSR method, this technique uses the same logic to tackle multicollinearity issue in wide datasets. However PLSR extracts components that maximise the covariance between **X** and Y looking for stable explanations of Y from **X**. [5]

2.2 Multi block problems

Common machine learning cases often consist of vectors of features and responses. Using machine learning tools, we aim to train a model in which the features explain the highest possible proportion of variance in target.[1] This model not only helps us spotting the most significant features impacting the variation of response but also contributes in the prediction of new responses based on the measurements we have about the features.

However, the current problem which this research studies, is a bit different from common machine learning cases. In other words, the features in this research are not defined as single variables and instead, we have blocks of multiple relevant features. Figure 2.2 demonstrates the problem.

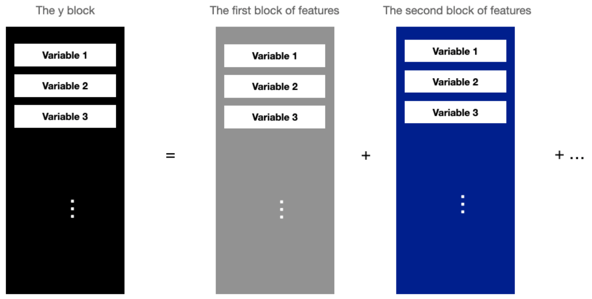

Figure 2.2: *Multi block problem demonstration.*

There are several methods which address these types of problems. During last 30 years, more than 50 different multi block techniques have been proposed. [6] However three most important methods are called Multi Block Partial Least Squares (MB-PLS) [6], Sequential and Orthogonalised Partial Least Squares (SO-PLS) [7] and Response Oriented Sequential Alternation (ROSA) [8]. In the following paragraphs we will explain two methods of MB-PLS and SO-PLS. ROSA which is the technique used in this research, would be discussed in detail in the subsection 2.3.

2.2.1 Multi Block Partial Least Squares (MB-PLS)

The MB-PLS mainly uses Partial Least Square Regression (PLSR) to directly merge the input blocks to the predictor matrix.[8] It has been shown that both MB-PLS and ROSA need variables within a block to be on the same scale. [8] MB-PLS will struggle if the dimensions of the blocks are very different or if the number of underlying components in each block is very different as it extracts the same number of components from all blocks. [8] MB-PLS scales each block by $\frac{1}{\sqrt{J}}$, where J is the number of variables of the blocks before computing PLS on the concatenated (scaled) $\mathbf{X}$ against the response. [8]

2.2.2 Sequential and Orthogonalised Partial Least Squares (SO-PLS)

Sequential and Orthogonalised Partial Least Squares is based on sequential multi block modelling of response variable.[7] In other words, this method tries to separately construct partial least square models using blocks of the features we have in our data in stepwise manner. [7] After making the models, SO-PLS ensures that matrices being used in the stepwise PLS regression models are orthogonalised with respect to each other. [7] In other words, every time a block has been modelled, the information extracted is removed from the following blocks. [7] This helps for problems which have different dimensions within the blocks. [7] SO-PLS is a suitable method when we have wide dataset in which the number of features are more than samples. [7] Experience has shown that this method has interpretational advantages when comparing to the MB-PLS. [8] However, we should bear in mind that SO-PLS is not suitable when we have more than two blocks of features since it will be harder and harder to interpret when more blocks are included. In other words since the later blocks will contribute little to the model in addition to being orthogonalised quite heavily, interpreting the loadings would be difficult.[8]

2.3 Response Oriented Sequential Alternation (ROSA)

As has been already mentioned, the Response Oriented Sequential Alternation (ROSA) is the method being used in this research. This method is specifically very suitable at the times when we have many blocks. [6] In other words, it has been said that the advantage of ROSA over SO-PLS is that ROSA can even be used with large number of blocks. [6]

ROSA mainly uses Partial Least Square Regression [4] to choose components. Therefore, it can be said that ROSA is an extension of PLSR. [8] ROSA has the ”winner takes all” approach in which winner components are being chosen from the blocks that could reduce the error.[8] In other words, firstly the PLS score is being computed for all of the blocks. Then the block which has the smallest error for the PLS model is selected. [6] The important aspect of this method is that consequence manner of block selection in every iteration helps the blocks getting a new chance in each block selection, so they always have this chance to surpass the blocks which had been chosen already in the earlier iterations. [6]

Understanding how this method works precisely, two main steps of ROSA is discussed below.

1) In the first step, a separate PLS regression model is fitted to each of feature blocks. [8] Thus for every block of **X**, we have a local model created by PLSR. Then the winner component is chosen based on the competition between the residual-minimising candidate components computed from each data block.[8]

It should be also mentioned that the block competition rule of ROSA is a forward selection approach where blocks can be used several times (but not excluded after selection). [8]

2) Throughout the second step, after orthogonalising the winner score to the target, the competition between current scores based on residual-minimising approach in the second iteration is formed.[8] ROSA ensures that every block gets a new chance to outperform in every iteration [6] so each winner component in every step, might be either form different blocks or from the same as previous iterations.

We can summarise what had been said in the algorithm in the following table. In this table, m is the block counter and M represents the maximum amount that m can take. Moreover, r is the component counter having R as maximum number.

Algorithm 1: ROSA algorithm extracted from [6]

```
Loop over components, $r = 1, ..., R$        -main loop, similar to PLS
  Loop over blocks, $m = 1, ..., M$          -competition for current component
  $PLS2(\mathbf{X}_m, \mathbf{Y})$           -one candidate component per block
  $\rightarrow \mathbf{t}_m, \mathbf{w}_m$   -scores and weights, both are scaled to unit norm
  End block loop
$\mathbf{t} = argmin_m\{||\mathbf{Y} - \mathbf{t}_m\mathbf{t}_m^t\mathbf{Y}||\}$   -select block that minimises residuals
$\mathbf{t} = \mathbf{t} - \mathbf{TT^t}t$                - orthogonolise on previous scores
$\mathbf{t} = \mathbf{t}/||\mathbf{t}||$                  - normalise scores
$\mathbf{w} = [0, \mathbf{w}_r^t, 0]^t$                   - global weights (0 except winner)
$\mathbf{Y}_{new} = \mathbf{Y} - \mathbf{tt}^t\mathbf{Y}$ - orthogonolise on winning blocks
End block loop
$\mathbf{P} = [\mathbf{X}_1, ..., \mathbf{X}_M]^t]\mathbf{T}$  - global loadings for concatenated X
$\mathbf{Q} = \mathbf{Y}^t\mathbf{T}$                      - global Y loadings
```

$\mathbf{w}_r$ are the weights corresponding to the winning block of component r.

Figure 2.3 also demonstrates how ROSA selects the components for an arbitrary order of blocks and number of components as well. [6] As it has mentioned before the process starts with selecting the winner score based on minimising the distance to residual response of $\mathbf{Y}$. [6] After making the $\mathbf{Y}_{new}$ by subtracting the winning score from it ($\mathbf{Y}_{new} = \mathbf{Y} - \mathbf{t}_r\mathbf{q}_r' = \mathbf{Y} - \mathbf{t}_r\mathbf{t}_r^t\mathbf{Y}$), we repeat the process until reaching the desirable number of components. [6] As an example, in this figure the order of block selections is 2,1,3,1. Obviously this order is completely based on the data. It should also be noted that **P** and **W** are representatives of loadings and weights respectively and they can span all of the subspace spanned by the blocks. [6] We should also consider this issue that block-wise feature selection only allows non-zero weights for blocks per component. [6] These zero weights are shown in white boxes in the following figure.As an explanation of the shaded areas in **P**, these areas are basically footprints of the winning blocks not the real part of **P** we interpret.

Experience has shown that ROSA is relatively fast when comparing to other multi-

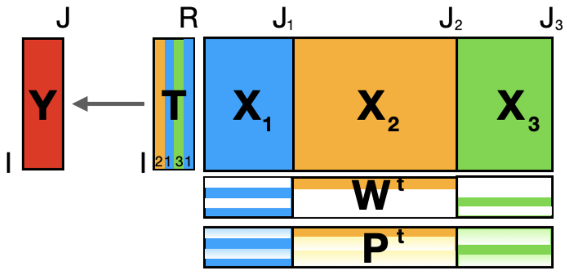

Figure 2.3: ROSA component selection adapted from [6]

block analysis methods. The reason is because ROSA mainly considers all the blocks together at the time and computes the orthogonal scores and loading weights.[8] With SO-PLS one either has to optimise component selection one block at the time (greedy approach) or using all possible component combinations (up to a limit) across blocks (global approach).[8] The former can be sub-optimal, the latter can be very time consuming.[8] Specially, computing candidate scores is a quick process, and the rest of ROSA is almost identical to PLSR, i.e. very quick and simple. In contrast to the global SO-PLS approach, ROSA considers just a single set of components (the selected ones). [8] Another advantage of ROSA is that, this method is stable when you have not scaled your blocks. [8] The reason of this case is that ROSA just uses residuals or prediction errors to select the blocks. Thus block selection does not depend on different scales of the blocks. [6]

ROSA and SO-PLS are both scale invariant. [8] However, ROSA has another advantage when comparing to SO-PLS. This method does not rely on the ordering of the blocks. [8] In other words, SO-PLS choses the components based on the covariance-maximising of components with target in every block so you should take one more step to order the blocks as well, however ROSA considers all the blocks as one block and choses the components from one block. [8] Therefore this method is invariant to block ordering. [8] It is also possible to say that ROSA works like a variable selection technique in which variables are blocks in this method.[8]

We already argued that ROSA is computational effective. [8] The reasons of this declaration is that firstly, without any need for convergence of the optimised solutions, subspaces are directly computed and secondly (which has been already mentioned), ROSA does not need any block ordering implementation in advance.[8] Winners are just components being in one single block at the time. [8]These are all reasons which contribute to lowering the computational time. [8]

Regarding stability, we already know that outliers can influence selection of the components and ROSA is not an exception.[8] Hence it can be wise to apply some outlier detection methods in advance so that you can prevent the final model to be somehow unstable.[8]

2.3.1 Model performance

In order to evaluate the performance of the model we use two criteria:

- R^2 (coefficient of determination)
- RMSEP (Root Mean Square Error of Prediction)

The first measurement takes values between 0 and 1 and basically determines how much of target variance could be explained by the components in the model. The closer this value is to 1, the better model performance is. [1] Formula 2.1 shows the mathematical explanation of this criteria.

$$R^2 = 1 - \frac{Unexplained \quad variation}{Total \quad variation} \tag{2.1}$$

Although R^2 can be so useful in evaluating the model performance, it is not enough and moreover to that we also use another criterion which basically measures the error of the model. This criterion is called root mean square error of prediction and we want the lowest possible amount of it. Formula 2.2 demonstrates that how it can be computed. In this formula y_i is the i'th sample observation and $\hat{y_i}$ is the corresponding prediction using the model. n also is the number of samples. [2]

$$RMSEP = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y_i})^2}{n}} \tag{2.2}$$

2.4 Dataset

Implementing validation techniques especially when the number of samples is not high is necessary. The reason and the method been used for this purpose is discussed in detail in the coming sections.

Moreover to this, the values that our target variable can take is also a vital issue. The distribution of target and how it might affect the model performance is a case which is discussed in detail in subsection 2.4.2.

2.4.1 Cross-validation

Basically using the same data for training and testing the model is a false move. [9] In other words, a model can perfectly work on the prediction of the observations we trained the model with, but has a poor performance on unseen data. [9] Tackling this problem, we can make test and training splits out of the data and train the model based on the training split and test it on the other cut of the test split. [9]

[1] https://www.investopedia.com/terms/r/r-squared.asp

[2] https://en.wikipedia.org/wiki/Mean_squared_prediction_error

However, having one split of test and train is not enough. Cross-validation helps us having several splits of the data, so that every sample in the dataset will have the chance to be at least once in the test dataset. The below flowchart shows how cross validation works. [10] [11] [12] [13]

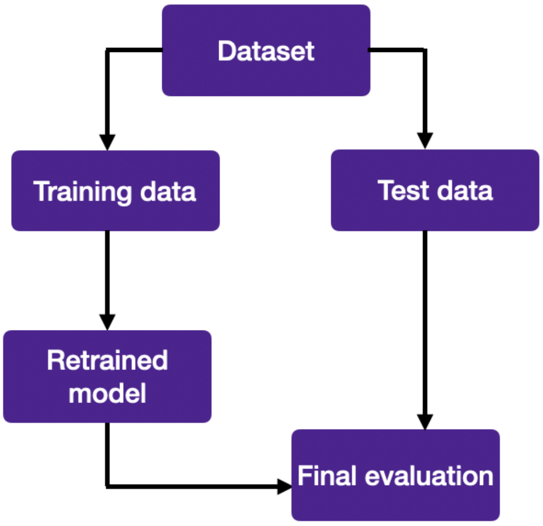

Figure 2.4: *Cross-validation workflow. Figure adapted from [9].*

Cross-validation in ROSA

ROSA is based on greedy algorithm. [6] As a reminder greedy algorithms introduce the solution step-wise, always selecting the best solution locally. [3]. Therefore it is very likely that at each step a different block wins. [6] Due to this fact, having a validation when using ROSA is necessary. Specifically, considering figure 2.4, having several validation and training sets can be considered as the main solution. [6] However since the block selection is part of the problem, we should think of nested cross validation (specifically double cross validation) to help improving the model performance as much as possible.

Double cross-validation

Double cross validation works the same as nested cross validation. In fact, nested cross validation is often used in situations where moreover to training models' errors, the hyper-parameter(s) of the model is also needed to be optimised. [9] In other words, if parameters and complexity (number of components) are optimised in the same loop, overfitting may occur. [9] [6]

[3] https://en.wikipedia.org/wiki/Greedy_algorithm

When implementing the ROSA algorithm, the block selection can be considered as the hyper-parameter needed to be estimated in nested cross-validation. [6] That is to say, since in every iteration different subsets of samples can be chosen, there would be variations in selections of the blocks. [6] As a reminder, ROSA is a greedy algorithm and the block selection is done locally, therefore in every iteration the chosen blocks can be different. [6] Thus double cross-validation is necessary for this problem. Figure 2.5 visualise the workflow of this technique.

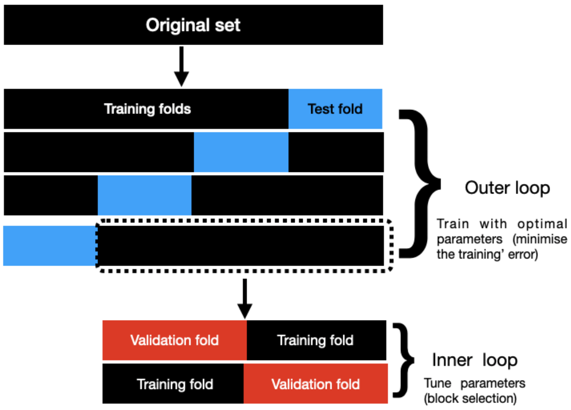

Figure 2.5: Nested cross-validation workflow. Figure adapted from [14]

2.4.2 Study on the target variable

A target with non-symmetric distribution can lead to a model with high error. As an instance, if the target has uneven distribution with many small values and fewer high values, where the high values will dominate the modelling as a contrast to the small ones (like the distribution shown in figure 2.6) , some scaling methods should be applied to the data in order to prepare it for modelling.

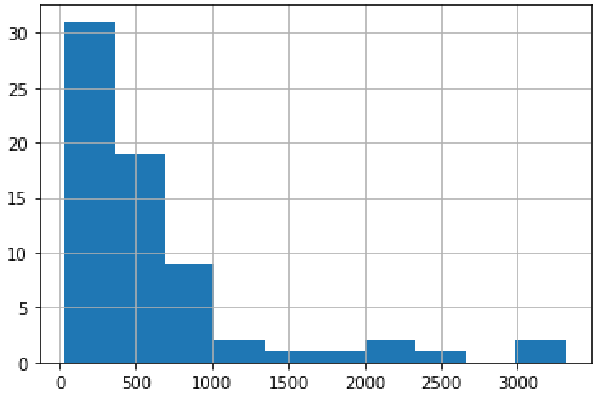

Figure 2.6: An example of uneven target distribution

Box-Cox transformation

This transformation is one of the methods which helps scaling the target to have more symmetric distribution. However this method also helps normalising the target in such a way that the distribution of the response becomes so close to normal distribution. [15]

This transformation is part of a family transformation called power transformation. [15] As it is obvious from the name these types of transformations raise the values to a power. [15] To transform variable y using Box-Cox, we should use the following formula: [15]

$$y(\lambda) = \begin{cases} \frac{y^{\lambda}-1}{\lambda} & \text{if } \lambda \neq 0 \\ \log y & \text{if } \lambda = 0 \end{cases} \tag{2.3}$$

Choice of λ is based on the best symmetric distribution Box-Cox can yield. [15] For the data demonstrated in figure 2.6 *Sickit-learn* package in Python [9] automatically finds the optimal value of λ. The transformed Y has a distribution shown in figure 2.7.

Figure 2.7 shows that the distribution of transformed Y is much closer to a symmetric distribution. In the latest chapter we will see for our case, how transformed vector of target variable could help us having a better model.

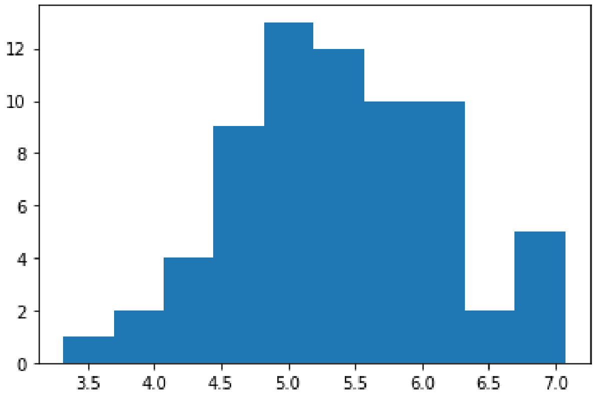

Figure 2.7: *Distribution of transformed Y using Box-Cox*

Chapter 3

Materials

The scope of this study was to model the number of the days between the diagnosis date and the last observation of patient using features in two different blocks. The data has been provided from *Functional and Molecular Imaging Group of Oslo University Hospital.*

Explaining the blocks, the first one includes features about patient clinical properties and the second one has variables about blood values. It should also be noted that in the first block mostly the results of Nordic chemotherapy in poorly differentiated cancer (PDEC) study are included. These results obtained by a survey about patines' habits and sickness history.

In the following section we will discuss every single feature in our research followed by an explanation about the target and how it has been used in the model.

3.1 Features of the first block

As has been mentioned the first block consists of features about patients' clinical properties.This block has 26 variables including different types of numerical, nominal or date features.

This number will increase in number when nominal features in the block are transformed to numerical values to be used in machine learning tasks. The reason of this is that transformation methods make different levels of the nominal variables as separate features. (More explanation about the transformation is given in chapter 4.2.2).

1. Age: varies from 38 to 94 years old.
2. DATEMET - DATEDIAG : this variable is the number of days between the first metastasis and diagnosis of cancer.

3. Sex: male or female

4. PRIMTUM: this is a prognostic factor and basically identifies the primary tumour specially if the patient has several metastases. In the current study this feature has 7 levels of gastric, colon, pancreas, rectum, oesophagus, unknown and other.

5. PRTUMRES: primary tumour resected, that is to say some of the few patients that had surgery might have a better overall survival. This feature has two possible answers of yes or no.

6. OPT: indicates if the patient had any other prior therapy. The levels this variable takes are none, radiotherapy, streptozotocin, sandostatin, interferon or other.

7. SURGMET: if the patient had any surgery metastasis.

8. SMOKHAB: smoking habits of patients. It can take several nominal values of non-smoker, smoker, ex-smoker or unknown.

9. PROTHRCA: indicates if the patient has prior other cancer. The response is either yes or no.

10. MORPH: indicates the morphology (or structure) of tumour. The values are small or large cell carcinoma or other shapes.

11. KI67: this is an indicator of rate of cell growth. KI67 is a protein in cells which increases as the cells prepare to divide. [1].

12. CGA1: cancer associated gene takes 4 values of strongly positive, partly positive, negative and not done.

13. SYNAPTOF: it is a prognostic factor for immunohistochemical marker. Levels of this feature are same as CAG1.

14. OCTREO: indicator of octreo scan. This feature is a type of imaging modality. Different levels include not done, negative, pos<liver and pos>liver

15. SOM-LIVER: if there had been any metastasis at liver in start of chemotherapy. The values it can take is either yes or no.

16. SOM-LYMPHNDS: if there had been any metastasis in lymph nodes at start of chemotherapy. The values it can take is either yes or no.

17. SOM-LUNG: if there had been any metastasis in lung at start of chemotherapy. The values it can take is either yes or no.

18. SOM-BONE: if there had been any metastasis in bone at start of chemotherapy. The values it can take is either yes or no.

[1] https://www.breastcancer.org/symptoms/diagnosis/rate_grade

19. SOM-BRAIN: if there had been any metastasis in brain at start of chemotherapy. The values it can take is either yes or no.
20. SOM-OTHRORGM: if there had been any metastasis in any other organ at start of chemotherapy. The values it can take is either yes or no.
21. PERFSTAT: WHO performance status, for more information see table 5.2.
22. BMI: body mass index which ranges from 18 to 42 in our dataset.
23. HORMSYMP: hormonal symptoms which the patient either has it or not.
24. CARSYNDR: carcinoid syndrome which the patient either has it or not.
25. TIMETOTRM1: the number of days between the first treatment and the diagnosis of cancer.
26. RESPONS1: This is the variable that shows how the patients responded to the treatment. This feature is measured based on CT scans and has several levels of complete response, partial response, progressive disease and stable disease.

3.2 Features of the second block

The second block has variables which define the blood values. This block has 8 nominal variables which will increase in number when transforming them to numerical ones. (More explanation about the transformation is given in chapter 4.2.2). The features of the second block are explained as follows:

1. HIAA: this is a test to help diagnosis of carcinoid tumours. [2] This feature has 4 different levels of HIAA $>$ 2×upper normal limit, normal$<$ HIAA $<$ 2× upper normal limit , normal and not done.
2. CGA2: chromogranin A, it is a feature helping for diagnose and carcinoid tumours and other neuroendocrine tumours. [3] The levels of this variable are the same as values of HIAA.
3. HMGLBN: measure of haemoglobin in blood. It has 3 levels of normal, not done and HMGLBN $<$ 11.0 g/dL.
4. LACTDHDR: prognostic factor of lactale dehydrogenase. This feature has 4 different levels same as HIAA and CGA2.
5. PLATELTS: prognostic factor of platelets. This variable has 3 levels of normal, not done and PLATELTS $>$ 400x10 9/L.

[2] https://labtestsonline.org/

[3] https://labtestsonline.org/tests/chromogranin

6. WHITEBLD: prognostic factor of white cell blood count. It has 3 levels of normal, not done and WHITEBLD $>$ 10x10 9/L.

7. CRETININ: creatinine of blood. It is measured either as normal or $>$ normal.

8. ALKPHSPH: alkaline phosphatase amount in blood. This variable has 4 levels of normal, not done, $3\times$ upper normal limit $<$ ALKPHSPH $<$ normal and ALKPHSPH $>$ $3\times$ upper normal limit

3.3 The response variable

The target in this study is a one dimensional continuous variable which shows the number of days between diagnosis of cancer and the last observation of the patients. Obviously the higher this number is the more days patients could live. Although at first using a binary response had been suggested, the model yielded better results when working with continuous variable.

Chapter 4

Methods

The methodology used in this study can be categorised into three phases: data preprocessing, feature selection and final model implementation. Preprocessing the data includes two main steps: 1) working with features (columns of data set) including either filter or transform them and 2) handling missing data as a part of working with samples (row data). Throughout the first step (sections 4.2.1 and 4.2.2) we mainly work on the variables in our dataset in order to prepare them to be used in the model. The second step works on the row data in order to handle the missing data, either ignore the feature including missing values or impute those items which are being missed. (section 4.2.3) The next phase is about feature selection.(section 4.3.1) Using Repeated Elastic Net Technique to select important features (RENT) [16] we come up with the features which would be used for the final model in the next phase. The last but not least step is implementation of our final multi block model using Response Oriented Sequential Alternation (ROSA) method. (In detail explanation was given in chapter 2.3)

The code used for this study can be found on GitHub at `https://github.com/gazelleazadi/Masters_Thesis/tree/main`.

4.1 Software

This study used Python Version 3.8.3 on an Miniconda platform with *Numpy* Version 1.18.1 and *Scikit-learn* [9] Version 0.22.1 for data preprocessing and feature selection. For the multi block part, RStudio Version 1.3.1093 has helped achieving the results.

4.2 Data preprocessing

Data preprocessing in this study encompasses below steps:

- Feature filtering
- Feature transformation
- Handling missing data
- Identifying outliers

Throughout the subsequent sections, we will explain the aforementioned steps in-detail.

4.2.1 Feature filtering

As [17] has defined, features are numeric representation of the raw data. Relevant features are those which can help having a better model in terms of its performance. In this regard, the number of features is important. [17] If there are few available features, the model can not capture the whole explained variance defined by them and on the other hand if there are many features which mostly are irrelevant, the model will be too complex and consequently too expensive to train. [17] Therefore feature selection plays an important role in preprocessing of data.

Generally speaking, there are three feature selection techniques. *Filtering*, *wrapping methods* and *embedded methods*.[17] Filtering techniques process the features to remove those which are not helpful is explaining the variance of the target. [17]

Experts' knowledge for filtering

As the first step of filtering, we decided to use dominant knowledge of experts to see which features are unlikely to be useful.

Our dataset includes 80 samples. We also have 2 blocks summing into 99 features. After several discussions about the features which obviously can not be helpful, many of the variables in the blocks have been disregarded. This step yields to having only 35 features which might or might not be the ones being used in the final step of modelling our problem.

Features with many missing values

In the next step, there had been some features which contain many missing values which neither could be imputed nor disregarded. As an instance in the third block we have a feature named *DATEPRG1* which is the date of progression of the patients after the first chemotherapy treatment. This feature includes 24 missing values which are not possible to impute since it is about the date of the progression. The missing samples can not be removed as well since we only have 80 samples and disregarding even one sample can lead to underperforming of the final model. Therefore variables of such containing more than 3 missing values were removed.

4.2.2 Feature transformation

In order to prepare the data to be used in machine learning algorithms, all the data samples must have numerical type. However in the real world data it is not always the case and most often the data needs to be transformed in a way that has the numerical type for all of the features. In the next subsections we will introduce two different types of non-numerical data in our data set followed by the solutions we implement to transform them into numerical values.

Features of nominal type

In our dataset there are some variables which describe a ’quality’ or ’characteristic’ of the data. These features which are called *nominal* or *categorical* variables require some specific techniques in order to become ready to use since machine learning techniques accept only numerical values. [18]

For those which only accept two values (for instance sex which is either male or female in our dataset) we simply define 0 as one level of the feature and 1 as the other one. However there are also several variables which take values more than two. For these features we used *OneHotEncoding* [9] to turn them into numerical values. This encoding transformer uses a dummy encoding scheme to make a binary column for each level of the variable.

Features of date type

In our dataset there are some features which are of the date type. These features require some arithmetic calculations in order to be prepared to be used in the machine learning algorithms, as the *Date* type itself is not acceptable to be used in the algorithms. As an instance for a variable like date of birth, we can simply change it to age which takes numerical values.

4.2.3 Handling missing data

Although disregarding features does not seem a good solution when they contain missing values, we decided to use the dominant knowledge of experts to see if the feature with many missing values are important or not. In this respect, features including more than 3 missing values had been removed for the next steps of the research and the rest of the variables with missing data are kept in order to impute the missing values.

Missing data imputation

Sickit-learn [9] imputation package offers several solutions to handle missing values. Generally *Sickit-learn* version 0.24.1 introduces three imputation methods:

- Univariate feature imputation
- Multivariate feature imputation
- K-nearest neighbours imputation

In the following subsections we will explain the methods in detail.

Univariate feature imputation

Univariate feature imputation is a technique of missing values estimation using information of the feature containing the missing value(s). [9] [19] Using this method, we can either replace the missing item by a constant arbitrary value or using statistics (such as mean, median or mode) of the column in which we want to impute the missing values. [9] [19]

Multivariate feature imputation

By contrast, multivariate imputation uses the information of all of the available features in order to estimate the missing value of one variable. [9] [20] For example, if item number i of feature m is a missing sample, multivariate imputation method estimates this value by considering samples which have similar situation in terms of all of the features in the dataset. Let us say, if the missing item is age of a sample which we already know is female, married and data scientist, multivariate uses information of the ages of all the married data scientist females in the dataset to estimate this value.

K-nearest neighbours imputation

This method uses the information of k-nearest neighbours of the missing item using the Euclidean distance. [21] [9] Using values of the k-nearest neighbours around the missing item, we can estimate the sample which is missed. This estimation can be based on the linear or weighted average of the aforementioned information of the k-nearest items. [9]

	f_1	f_2	...	f_M
M_1	β_{11}	β_{12}	...	β_{1M}
M_2	β_{21}	β_{22}	...	β_{2M}
...	...	...	...	...
M_k	β_{K1}	β_{K2}	...	β_{KM}

Table 4.1: *Weights matrix using in RENT feature selection technique*

4.3 Feature selection

Feature selection is a technique used to help reducing the dimension of the dataset, specially when the number of features exceeds the number of samples which is the case in our problem. Thus, having simpler data we may be able to make a more comprehensible model out of selected features. [22] There are several methods proposed to select meaningful features, however in this study we use Repeated Elastic Net Technique (RENT) [16] in order to extract the variables which are more significant than others. This technique is described in detail in section 4.3.1.

4.3.1 Feature selection using RENT

Repeated Elastic Net Technique for feature selection (RENT) [16] is a method which can be used to see which features should be included during the next step of final model. Considering the data matrix with N samples and M features we make some different train and test split.[9] In this way we make sure every sample in the dataset would have the chance to be at least once in the test split. [16] Now, for every splits, we fit K different generalised linear models-(so-called ensemble models)

After fitting the K models we will obtain a matrix of weights for every feature being fitted for every model. Below process shows what we achieve so far: [16]

$$\underset{\text{Input data matrix}}{\begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1M} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{NM} \end{pmatrix}} \Longrightarrow \overset{\text{Models}}{\begin{pmatrix} M_1 \\ M_2 \\ \vdots \\ M_K \end{pmatrix}} \Longrightarrow \overset{\text{Weights}}{\begin{pmatrix} \beta_{11} & \beta_{12} & \cdots & \beta_{1M} \\ \vdots & \vdots & \ddots & \vdots \\ \beta_{K1} & \beta_{K2} & \cdots & \beta_{KM} \end{pmatrix}}$$

For a better clarification, we can also rewrite the weights matrix as in table 4.1, in which f_1,f_2, ... , f_M are representing the M features we had in the dataset and M_1, M_2, ... , M_K are the K generalised linear models [23], we fit to our data.

In the table 4.1, β's are the weights for different features obtained in different models. For instance, β_{kM} is the weight for the M'th feature when fitting the data in the k'th model.

The next step of RENT needs explanation about a concept called regularisation, in advance. Therefore we will have a detailed overview on this concept and then continue to next steps of RENT feature selection.

Elastic net regularisation

One of the common problems in machine learning is overfitting. [14] This problem happens when the model is perfectly fitted to training set, however it does not generalise on the test set. [14] On the other hand, underfitting occurs when the model is not complex enough to capture the pattern in the training set. [14] Figure 4.1 illustrates this issue.

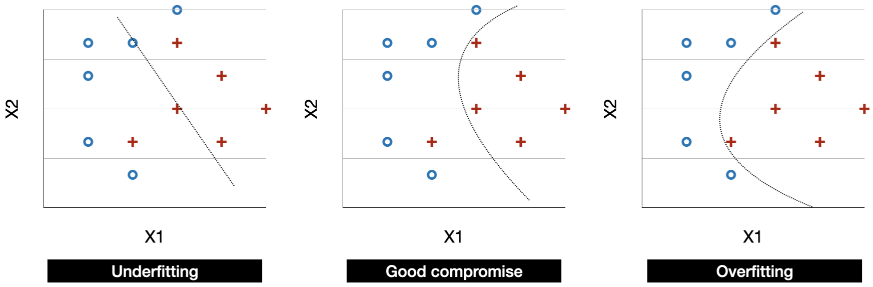

Figure 4.1: Illustration of underfitting, a good compromise and overfitting. Figure adapted from [14]

Tackling the problem of overfitting, we can adjust the model complexity by regularisation. [14] Regularisation mainly excludes noise from the data and also helps handling collinearity. [14]

The most common form of regularisation is called ℓ_2 regularisation. [14] This form is written as follows: [14]

$$\ell_2 : \frac{\lambda}{2}||w||^2 = \frac{\lambda}{2}\sum_{j=1}^{m} w_j^2 \tag{4.1}$$

In equation 4.2, λ is the so-called regularisation parameter, w_j's are estimated features' weights and m is the number of features in the model. [14] In fact by the regularisation parameter of λ we can have control over the model in such a way that how good enough we want it to be. [14] In other words we try to have a tradeoff point which satisfies us as a good compromise: neither too simple nor too complex. It should be mentioned that the higher amount of λ the stronger regularisation we have. [14]

Another approach for tackling the overfitting problem and shrink the complexity of the model is ℓ_1 regularisation. [14]

$$\ell_1 : \sum_{j=1}^{m} |w_j| \tag{4.2}$$

This approach yields to a model in which so many of the features' weights shrink to zero. [14] In the problems when we have so many collinear features, this form of regularisation can help us not only tackle the overfitting issue but also have some sort of feature selection. [14]

ℓ_1 regularisation has this limitation that our dataset should be wide. In other words if we have a dataset with m features and n samples, we can use ℓ_1 if $m > n$. [14]

Now that we learn about concepts of ℓ_1 and ℓ_2 regularisation forms, we can define elastic net. Elastic net is a compromise between ℓ_1 and ℓ_2 regularisation. [14] In fact, elastic net includes ℓ_1 to have sparsity on the features and at the same time having ℓ_2 regularisation helps having control over limitations of ℓ_1. [14]

Now that we know what elastic net is, we can continue with next steps of RENT:

Using the information given in table 4.1, we can have a statistical summary of the weights for every feature in the model. In fact, these summary statistics will help us identifying the most important features in the dataset. [16] Defining three criteria as well as thresholds, we will come up with the selection of the important features. In other words for every criterion we defined we should check for the threshold to see if the feature would be selected or not.

Let us clarify this by representing the following matrix of criteria: [16]

$$\begin{array}{c}\text{Summary statistics}\\ \begin{pmatrix} \tau_1(f_1) & \cdots & \tau_1(f_M) \\ \tau_2(f_1) & \ddots & \tau_2(f_M) \\ \tau_3(f_1) & \cdots & \tau_3(f_M) \end{pmatrix}\end{array}$$

Using this matrix and pre-defined thresholds of t_1, t_2, t_3 we select the feature i that satisfies the following equation: [16]

$$\begin{aligned}\tau_1(f_i) &\geq t_1,\\ \tau_2(f_i) &\geq t_2,\\ \tau_3(f_i) &\geq t_3.\end{aligned} \tag{4.3}$$

Now the question is what are those criteria based on statistics summary of the weights? Answering this question, we say generally for each of the τ's, refer to the definitions in equation 4.4, a feature of f_i is selected in RENT:

1. this feature is selected by elastic net frequently.[16] In other words, when fitting K models, the feature is selected after imposing ℓ_1-norm and ℓ_2-norm penalties which are solutions to detect the irrelevant variables and identify highly correlated ones which have relatively similar regression coefficients, respectively. [24]
2. the weights of the feature are stable.[16] That is to say, if the feature f_i gets weights ranging from negative to positive values across the K models, and the values hops around quite often, it is been concluded that this feature is not stable and can be eliminated.[16]
3. across the K models, the feature's estimations have been calculated as non-zero values with relatively low variance. [16] So, even if the weights of the feature differ from zero with stable behaviour, but the estimation of the itself (the mentioned features) in the models are mostly close to zero, we still do not select the feature since it does not fulfil the third criteria. [16]

Formulating the above conditions into mathematical expressions for arbitrary feature f_i, we come up with the equation 4.4. Noted that $c(\beta_i)$ is the score as for the first criterion which specifies the frequency of selection of the specific feature of β_i. [16]

$$\begin{aligned}\tau_1(\beta_i) &= c(\beta_i),\\ \tau_2(\beta_i) &= \frac{1}{K} \mid \sum_{k=1}^{K} sign(\beta_{i,k}) \mid,\\ \tau_3(f_i) &= P_{K-1}(\frac{\mu(\beta_i)}{\sqrt{\frac{\sigma^2(\beta_i)}{K}}})\end{aligned} \tag{4.4}$$

Where P_{K-1} is the cumulative density function of Student's t-distribution with $K-1$ degrees of freedom, $\mu(\beta_i)$ and $\sigma^2(\beta_i)$ are the feature mean and variance respectively, i.e for feature β_i. [16]

Considering the second criterion $\tau_2(\beta_i)$, the best case is when all the estimated weights of the feature f_i have the same sign. (either positive or negative) However in the real world problems, it is usually not the case and this criterion allows us to define a minimum proportion for the weights which have the same sign. [16]

For the the third criterion $\tau_3(\beta_i)$, considering estimations of the parameter feature f_i across all the K fitted models, we test the average of all the estimations are equal to zero or not. The hypothesis is tested under the Student's t-distribution with $K - 1$ degrees of freedom. [16]

$$H_0 = \mu(\beta_i) = 0$$

Having the thresholds defined in equation 4.3, RENT would tell us if the feature f_i would be selected or not. We should remember that selecting a feature is dependant on the fulfilment of all of the criteria $\tau_1(\beta_i)$, $\tau_2(\beta_i)$, $\tau_3(\beta_i)$ as expressed in equation 4.4. [16]

4.3.2 Validation study regarding the models made by RENT

One of the interesting analysis which RENT makes it possible is to have a feasibility study. Feasibility study helps us assessing practicality of RENT. This study consists of two cases:

The first validation study (so called VS1) tells us if the RENT feature selection is actually better than random selection of features. [16] Let us say we make M models just by randomly taking some features. Then we check if the performance of the model in which the features had been selected by RENT is better than the average performance of M models.[16] If this is the case, we can say the selected features by RENT are meaningful on the test dataset.

The second validation study (so called VS2) is basically done by permutation of test labels. Better to say, we randomly permute the test labels for many times (let us say 1000 times) to see if the performance changes or not. In other words, in the test data we keep the order of the rows in **X** (features' matrix), but permute (change order) of the target values. This means that we break the mapping of **X** to Y. Then, we compute the performances. If RENT feature selection is doing a good job, the performances' distribution of permutations, should be worse than the RENT prediction score. [16]

Take the figure 4.2 into consideration. We have run RENT on a block of features trying to see if this algorithm is doing a proper job of selecting the important features.

In this figure, the red line shows the prediction score of RENT. In other words we take the training data and use RENT to select features. Then using these features we predict classes of test data. Now we compute the test performance which is the red line in the figure. In addition to that, the blue curve is the distribution of performances of the models made by random feature selection. As we can see, the RENT prediction score is almost higher than this distribution. The green curve is the distribution of the second validation study. As we can observe, the prediction score of RENT is better than distribution of permuting the test labels in the green curve, as well. Therefore we can rely on the results of RENT and announce the selected features yielded by RENT as important ones to be used in the next steps of the research.

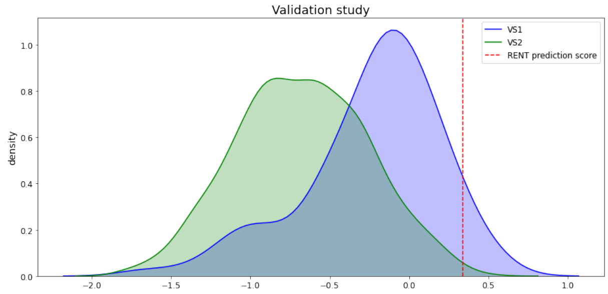

Figure 4.2: *An example of validation study of RENT*

It should also be noted that the performance metric using in the validation study of RENT is MCC (Matthews Correlation Coefficient). [16] This metric is a contingency matrix method between actual and predicted values. [25]

If we consider our case a binary problem, wishing to measure the performance of the machine learning model fitted on the corresponding dataset, we can have below terms, followed by the formula of MCC metric: [25]

- Actual positives that are correctly predicted positives are called true positives (TP);
- Actual positives that are wrongly predicted negatives are called false negatives (FN);
- Actual negatives that are correctly predicted negatives are called true negatives (TN);
- Actual negatives that are wrongly predicted positives are called false positives (FP).

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP) \times (TP + FN) \times (TN + FP) \times (TN + FN)}} \tag{4.5}$$

4.4 Identifying outliers

One of the most important aspects of preprocessing is detecting samples which appear to deviate significantly from other members of the sample. [26] Therefore it is crucial to detect outliers to obtain a better model for analysis.

In this study we used two techniques in order to identify outliers. The first method is mainly based on the principal components of the features and the second method is found on RENT by which we will identify observations that caused the error of predictions to be relatively high. Both of the methods are discussed in detail in the following sections.

4.4.1 Principal Component Analysis method

Principal Component Analysis (known as PCA) is a method by which the components which define the most proportion of the variance of the p features are determined (principal components). If these components would be lower in number comparing to the number of original features, PCA can be used as a dimension reduction technique. [27] However this research aims to use PCA to detect the outliers in the dataset. We should also bear in mind to standardise our data before implementing PCA on it. The reason would be discussed in the following section.

Standardising the data before PCA

Visualising the principal components using PCA, having the same or at least similar measurement scales of features is pre-assumed. [28] In other words, since every variable in our dataset has its own specific scale of measurement (for instance age as year and BMI as unit) we should think of unifying their scales. Standardised features are easier to interpret regardless of their identity as an age, BMI or any other medical measurement scale. The following formula shows how the data is standardised. [28]

$$X_{i,j}(std) = \frac{X_{i,j} - \bar{X}_j}{\sigma_{X_j}}$$

Which literally means subtract the mean value from each feature and then dividing the result by the standard deviation. This will yield a dataset in which all the columns have mean value of zero and standard deviation of 1.

Using PCA to detect the outliers

The principal component analysis for every standardised blocks of the dataset gives us scores and loadings plots. As a reminder, scores plot contains the original data in a rotated coordinate system and loadings can be understood as the weights for each original variable when calculating the principal component. [29] The outliers are those observations which have relatively large deviation from the centre of the subspace spanned by the principal components. [29]

4.4.2 *Hotelling's* T^2 statistic for more than two PC's

As has been discussed, when we have two principal components (PC's), 2D scores plot is sufficient to detect the outliers. [30] However this is not always the case. In other words, when working with real world data, it is very possible that the variance is explained with more than two scores. In such situations, some outliers may not be identified in a 2D plot since there can multiple combinations of two PC's among all the components. [30] Therefore it is suggested [30] to use *Hotelling's* T^2 statistic. This statistic is the multivariate version of *Student's t-test* and can be calculated as the sum of squared scores for each sample and the corresponding largest values are the outliers. [30] Note that this statistic is just a guide and can not be considered as a hard rule to find the outliers. [30]

4.4.3 Using RENT in order to identify the outliers

As already discussed in section 4.3.1, Repeated Elastic Net Technique is basically a method for feature selection. However this method can also help us identifying observations which cause the error to be high. [16] In other words, for every object in the dataset firstly we can see how often this observation had been part of the test set and secondly what is the average absolute error of predictions when this object had been part of this test set. [16]

As a note, in our research not only high number of models in RENT (K different generalised liner models in section 4.3.1) but also using repeated k-fold cross validation (explained in section 4.5) ensure us that every sample would get at least one chance to be in the test set. It should also be reminded that the absolute error is defined as the absolute value of difference between prediction and the true value of the sample. [1]

4.5 Repeated stratified k-fold cross-validation

Cross-validation is a process which ensures every sample in the dataset has at lest one chance to be part of the test set which mainly tests the performance of the model trained by the rest of observations as training set. [9] (More in detail discussion in chapter 2.4.1). In chapter 2.4.1 we introduced one of several approaches can be extracted from cross-validation as double validation. In this section we are about to present another approach, so-called repeated stratified k-fold cross-validation. We use this technique when using RENT in order to ensure that features

[1] `https://mathworld.wolfram.com/AbsoluteError.html`

have been selected are validated enough. This method specially makes a significant contribution in situations where there are not so many samples to work with.

This method works based on the following steps: [9]

- Train the model based on $k - 1$ of folds of the data (as training splits)
- Test the model based on the one split left in the dataset (as the test split)
- Repeat the above steps for several selection of random indices of samples as test or train splits

It should be noted that *stratified* means that this method tries to preserve the distribution of the target in different splits of the dataset. Better to say, if the distribution of the target is imbalanced, *stratified k-fold cross-validation* ensures relative class frequencies have been kept across the different splits. [9]

Figure 4.3 demonstrates one repeat of cross-validation splitting flow (for 5 folds):

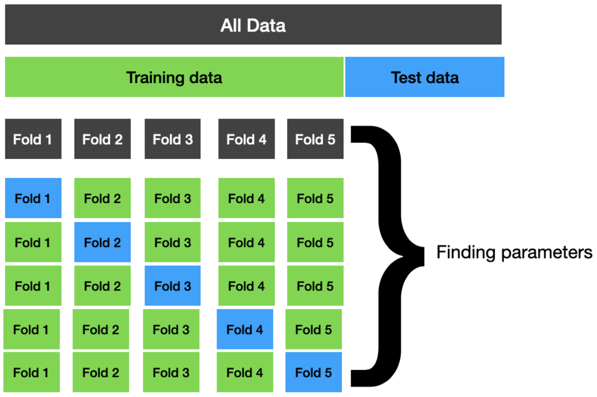

Figure 4.3: *One repeat of cross-validation splitting flow. Figure adapted from [9]*

Chapter 5

Results

In the early phase of the research we decide to define the target as follows:

$$y_i = \begin{cases} 0 & \text{if the patient died at some time during or after treatment} \\ 1 & \text{Otherwise} \end{cases}$$

However we observed that considering a classification problem in which the target is a binary variable being defined as dead or alive does not help us modelling our problem properly. The main reason of this issue was the extremely imbalanced data with respect to the binary target. In simple words, if we consider our response to be dead or alive as a binary variable, almost 90% of the patients in the dataset would be categorised as dead people whereas the rest of 10% are alive. This issue causes so many problems specially when it comes to splitting the data. When splitting data to train models, we would not be able to maintain the target distribution in all the splits being made. In other words, having very low number of alive people does not help assigning them equally in the splits so that in all of them we would have 10% alive people. This obstacle forced us thinking about some solutions to change the target from binary to continuous variable as the number of days of last observation and the diagnosis. This simple transformation of target made a significant contribution to our problem.

Throughout the next step, we preprocess our data by implementing all the methods and techniques being discussed in chapters 2 and 4 on our dataset. The results of these implementations are discussed in detail in the following section. It should be noted that the structure of this chapter is written based on the order of the tasks being done on our dataset: firstly beginning with the preprocessing of the data, in the next step doing the feature selection and finally the main model would be fitted on the processed data.

	# of features before filtering	# of features after filtering
Block 1	34	29
Block 2	18	9
Block 3	34	2
Block 4	5	1

Table 5.1: *Number of features in our dataset*

5.1 Preprocessing the data

As has been mentioned in 4.2, data preprocessing includes five steps as follows:

- Feature filtering
- Feature transformation
- Handling missing data

In the coming parts, we will discuss about the results we obtain by performing the above steps on our dataset across the preprocessing step.

5.1.1 Feature filtering

Having several meetings with experts, we finally decided to use 41 features out of 87 variables. Among the excluded ones, there were also those which encompassed many missing values and it was not possible to impute them due to their large size. Table 5.1 summarises the number of the features in every block that had been decided to use.

It should be noted that we decided to use two of the features in the third block. However since the first variable is defined as if the patients respond to the treatment, we could place it in the first block. The second feature is the date of the first treatment after diagnosis. Considering this feature and subtract this date from the date of diagnosis in the first block, we come up with a new variable as days between diagnosis and the treatment. This feature is also decided to be considered in the first block. The rest of the features in this block had been disregarded due to their high number of missing values. Moreover to this, the only variable being used in the fourth block is *DATELOBS* which helps defining our target as the number of days from the diagnosis until the patient dies. The diagnosis date is accessible in the first block as *DATEDIAG*.

5.1.2 Feature transformation

As has been already discussed in section 4.2.2, two types of nominal and date type features needed to be transformed to numerical values in order to become prepared to be used in our machine learning algorithms. [18]

Degree	Performance Status
0	Able to carry out all normal activity without restriction.
1	Restricted in strenuous activity but ambulatory and able to carry out light work.
2	Ambulatory and capable of all self-care but unable to carry out any work activities; up and about more than 50% of waking hours.
3	Symptomatic and in a chair or in bed for greater than 50% of the day but not bedridden.
4	Completely disabled; cannot carry out any self-care; totally confined to bed or chair..

Table 5.2: *WHO performance status*

Sample number	Performance status value	Encoded performance status			
		WHO0	WHO1	WHO2	WHO3
1	WHO 0	1	0	0	0
2	WHO 0	1	0	0	0
3	WHO 1	0	1	0	0

Table 5.3: *An example of WHO performance status encoding*

Clarifying the *OneHotEncoding* technique to transform nominal types of features, we take the *WHO Performance Status* feature in the first block as an example. This variable takes values based on the information given in table 5.2. [1]

In our dataset this variable takes values ranging from 0 to 3. In order to transform this nominal variable into a numerical one, after using *OneHotEncoding* [9], we end up having 4 features according to 4 levels of the variable takes. For instance 3 samples of 1,2 and 3 have values of WHO1, WHO1, WHO0. After encoding the feature, we will have the results shown in table 5.3 for the mentioned samples as 1 represents accepting the level in which the column is defined.

Hence based on the level numbers of features needed to be encoded, we will create new features and in the final analysis of our model we can decide either a certain level of one feature has significant impact on our response block or not.

As an example of date type feature in our dataset is a variable called *Date of Birth*. This feature can simply be transformed into the *Age* variable which makes it become feasible to work with in making machine learning models. Other example

[1] https://www.nice.org.uk/guidance/ta121/chapter/appendix-c-who-performance-status-classification.

is making a new feature by subtracting two features of *DATEMET* (the date when the patient had metastatic disease at time of diagnosis) and *DATEDIAG* (the date of diagnosis) which yields either zero or a positive value. This feature is an overall survival days and tells us how long after diagnosis patients got metastasis. This feature now is as a discrete type and is perfectly feasible to work with in making machine learning models.

Tables 5.4 and 5.5 demonstrate the features being used in the block 1 and block 2 respectively.

Feature name	**Feature type**	**Transformation needed**	**Transformed feature levels**
DATEBRTH	date	yes	1
DATEMET-DATEDIAG	numerical	no	-
SEX	nominal	yes	1
PRIMTUM	nominal	yes	7
PRTUMRES	nominal	yes	1
OPT-NONE	nominal	yes	1
OPT-RADTHRPY	nominal	yes	1
OPT-STRPTCYT	nominal	yes	1
OPT-SANDOSTN	nominal	yes	1
OPT-INTRFERN	nominal	yes	1
OPT-OTHRPRTH	nominal	yes	1
SURGMET	nominal	yes	1
SMOKHAB	nominal	yes	5
PROTHRCA	nominal	yes	3
MORPH	nominal	yes	4
KI67	numerical	no	-
CGA1	numerical	yes	4
SYNAPTOF	nominal	yes	4
OCTREO	nominal	yes	5
SOM-LIVER	nominal	yes	1
SOM-LYMPHNDS	nominal	yes	1
SOM-LUNG	nominal	yes	1
SOM-BONE	nominal	yes	1
SOM-OTHRORGM	nominal	yes	1
SOM-BRAIN	nominal	yes	1
PERFSTAT	nominal	yes	4
BMI	numerical	no	-
HORMSYMP	nominal	yes	1
CARSYNDR	nominal	yes	1
RESPONSE1	nominal	yes	4
DATETRM1-DATEDIAG	numerical	no	-

Table 5.4: *Features in block 1*

Feature name	Feature type	Transformation needed	Transformed feature levels
HIAA	nominal	yes	4
CGA2	nominal	yes	4
HMGLBN	nominal	yes	3
LACTDHDR	nominal	yes	3
PLATELTS	nominal	yes	3
WHITEBLD	nominal	yes	3
CRETININ	nominal	yes	1
ALKPHSPH	nominal	yes	4
TUMMARK1	nominal	yes	1

Table 5.5: *Features in block 2*

5.1.3 Handling missing data

Three features of *KI67* , *BMI* and *CGA2* included one missing item each. In order to impute these missing values, we use three techniques of univariate (4.2.3), multivariate (4.2.3) and k-nearest neighbours (4.2.3) to estimate their values. Table 5.6 shows the imputation of missing items for these features.

Feature name / level	Univariate method	Multivariate method	K-nearest neighbours method
KI67	65	63	58
BMI	24	25	24.75
CGA2-> 2UNL	0	0	0
CGA2->Normal	0	0	0
CGA2-Normal	0	0	0
CGA2-Not Done	0	1	1

Table 5.6: *Imputation of missing values*

Eventually for two features of *KI67* and *BMI* taking average among estimations, we come up with the values of 62 and 24.58 as the imputations of the missing values. For the feature *CAG2*, taking the mode of levels which had been estimated as 1, we choose to consider the missing item as *CGA2-Not Done*.

5.1.4 Identifying outliers using PCA and the *Hotelling's* T^2 statistic

As has been discussed in chapter 4.4.1, by the help of principal components we can see which observations corresponding to which features in the dataset might be outliers, due to being relatively far from the centre of the subspace spanned by the principal components.[29] However for our problem, this method could not help us featuring these samples. As an instance take the figure 5.1 into consideration.

As it is been demonstrated, 60% of variance of the features in the first block is explained by around 20 components. Therefore finding the possible outliers by 2D

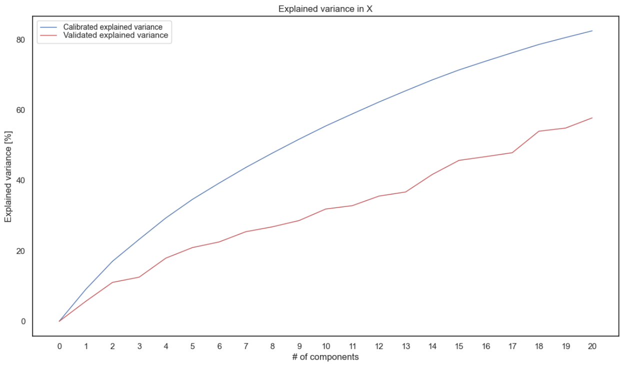

***Figure 5.1:** Explained variance in block 1*

scores plots requires checking all the possible permutations of the 20 components. Although this approach is feasible, the final results are not trustworthy since every permutation of two components might have different weight based on the variance that their combination is explaining.

Moreover to this, considering figure 5.2 the first two components in the first block are explaining around 17% of the variance of the features. However spotting the outliers based on the plot is not an easy task. Since none of the observations seems to be extremely far from the centre, we can not confidently announce any sample to be an outlier.

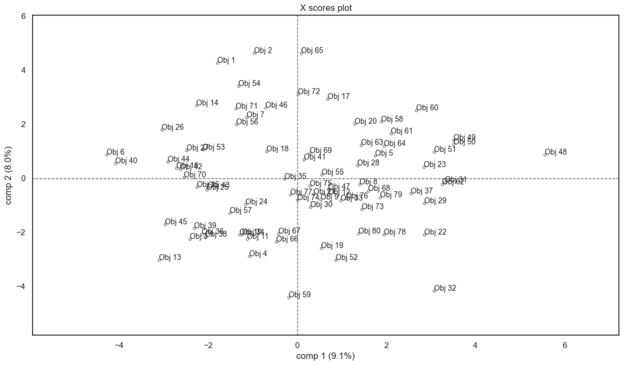

***Figure 5.2:** Score plot for the two first components in block 1*

That is why we ought to use *Hotelling's* T^2 statistic for each sample to find the outliers. (More detailed explanation is given in chapter 4.4.2). Using this method, we came to the conclusion that the scores are sensitive to the number of components. As an instance if we choose 20 components in our PCA for the first block, the largest sum of squared of scores is corresponded to the sample number 1. However if we choose 25 components, the corresponding sample number is 30. Moreover to that, different blocks of features yields different outliers. Thus, although PCA or *Hotelling's* T^2 statistic can sometimes help finding outliers, in our problem these two methods could not help us. However in the upcoming sections we will see how other methods actually found some samples which were outliers in our dataset and PCA or *Hotelling's* T^2 statistic did not spot them.

5.2 Feature selection

Using Repeated Elastic Net Technique (RENT) [16] we come up with three different approaches in order to choose the features in the final multi block method. These approaches have been attained by having RENT feature selection on different test sets in terms of features or samples being included in the dataset. In other words we compare the performances of the models obtained by the different test sets to see which dataset would yield better results. In the subsequent sections we will explain all the approaches in detail, however before that we will discuss how we used the repeated stratified k-fold cross validation 4.5 to ensure all the samples get the chance to be in the test split at least once.

5.2.1 RENT and repeated stratified K-Fold cross-validation

Referring to section 4.5, we had RENT [16] for feature selection [16] on different splits of training data to see which features have are most frequent of being selected across different splits of the dataset.

For our problem, we consider $k = 4$ splits with 2 repeats. Figure 5.3 shows these two repeats of stratified 4-fold cross validation on RENT model. As it is been shown, in every repeat, we train our model using RENT [16] on 3 folds of data and then test it on the remaining fold. Since we have 2 repeats and 4 folds, we eventually obtain 8 different models which can be compared in terms of their performance on the test set.

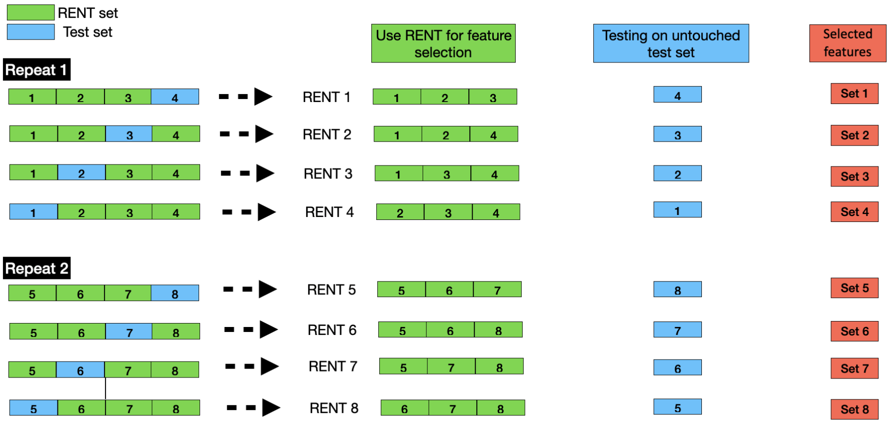
RENT set
Test set
Use RENT for feature selection
Testing on untouched test set
Selected features
Repeat 1
1 2 3 4 RENT 1 1 2 3 4 Set 1
1 2 3 4 RENT 2 1 2 4 3 Set 2
1 2 3 4 RENT 3 1 3 4 2 Set 3
1 2 3 4 RENT 4 2 3 4 1 Set 4
Repeat 2
5 6 7 8 RENT 5 5 6 7 8 Set 5
5 6 7 8 RENT 6 5 6 8 7 Set 6
5 6 7 8 RENT 7 5 7 8 6 Set 7
5 6 7 8 RENT 8 6 7 8 5 Set 8

Figure 5.3: RENT on two repeat of stratified 4-fold cross validation.

It should also be noted that the number of generalised linear models (K different models in chapter 4.3.1) in RENT have been considered as 700 and the model which evaluates the performance of selected features on unseen test data set is logistic regression. [16]

In the next couple of subsections, we will talk about three approaches we already talked about for implementing the RENT to select features.

5.2.2 Selected features based on the first approach

The first approach we consider is using the dataset with no missing values. In other words the features we use either do not encompass any missing items or have so few number of missing values which are also imputable. This full dataset has two blocks with 80 samples and each block consists of 57 and 27 features respectively.

Considering continuous target as the number of days that patients live after their cancer diagnosis, we obtain the following results on 8 different splits of data. It should be also mentioned that based on 80 samples, every training split consists of 60 and test splits have 20 samples.

The parameters used in RENT training are $\tau_1 = 0.9$, $\tau_2 = 0.9$, $\tau_3 = 0.975$. (Referring to section 4.3.1 for more explanation about the parameters). These numbers are the default values of RENT, however we also have the possibility of changing the parameters in order to improve our training leading to a better feature selection.

Reviewing tables 5.7 and 5.8, their second columns show which features have been selected for the corresponding split and the next columns demonstrate the performance of logistic regression model on the test fold based on the features being selected.

Splits	**Selected feature(s)**	R^2	**RMSEP (error)**
Split 1	AGE, OCTREO-Negative	-0.32	1157.8
Split 2	CGA1-Negative, SOM-LUNG	0.04	901.36
Split 3	CGA1-Negative, OCTREO-Negative	-0.27	1113.4
Split 4	SURGMET	0.27	387.8
Split 5	AGE, SEX, PRIMTUM-Colon, SURGMET, KI67, SYNAPTOF-Negative	-2.9	866.6
Split 6	Age, SEX, OCTREO-Negative	-0.12	1302.1
Split 7	PROTHRCA-No	0.05	684.4
Split 8	PRTUMRES, SURGMET, MORPH-Other, CGA1-Negative	-0.5	1174.8

Table 5.7: *Selected features and model performance based on the first approach for the first block*

Consequently on average, Repeated Elastic Net Technique selects features which lead to a model with performance of -0.47 and -0.4 for the first and second block respectively. Also, based on the metric defined in section 2.3.1, on average the RMSEP (Root Mean Squared Error of Prediction) of the models corresponding to the first and second blocks are 948.5 and 946.4. Negative performance and

Splits	Selected feature(s)	R^2	RMSEP (error)
Split 1	CGA2-Normal, PLATELTS$\geq 400x10^9/L$	-0.08	1045.9
Split 2	HIAA-Normal, CGA2-Normal, LACTDHDR $\geq$ 2UNL, LACTDHDR-Not Done	-0.07	951.8
Split 3	HIAA$\geq$Normal $\leq$ 2UNL, HIAA-Not Done, CGA2$\geq$ 2UNL, CGA2-Normal, CGA2-Not Done, WHITEBLD-Normal, ALKPHSPH$\geq$3 UNL, ALKPHSPH$\geq$Normal $\leq$ 3 UNL, ALKPHSPH-Normal, ALKPHSPH-Not Done, TUMMARK1	-0.39	1162.7
Split 4	CGA2-Normal	-0.97	638.5
Split 5	CGA2-Normal	-0.24	484.7
Split 6	CGA2-Normal, LACTDHDR-Not Done	-0.001	1229.4
Split 7	CGA2$\geq$2UNL, CGA2-Normal, CGA2-Not Done, LACTDHDR-Not Done, ALKPHSPH-Not Done ALKPHSPH$\geq$Normal $\leq$ 3 UNL, ALKPHSPH-Normal	-1.4	1094.1
Split 8	CGA2-Normal, LACTDHDR$\geq$ 2UNL, LACTDHDR-Not Done	-0.04	964.5

Table 5.8: *Selected features and model performance based on the first approach for the second block*

relatively high error demonstrates that this dataset and the models fail to explain the variance of the target which leads us to think about the next approaches.

5.2.3 Selected features based on the second approach

As has been already discussed, poor performance of the models using main dataset without missing values made us thinking about other feasible approaches to model the problem. In other words, we firstly prioritise having a dataset with full samples even this might lead to disregarding some features. However, after obtaining the first results we consulted with the experts and they suggested to include two more features which encompass 9 missing items in the first block. So, although this approach causes having a dataset with lower number of samples, we tried it and surprisingly obtained better results, explained as follows.

The first feature which had been included in the dataset, namely *RESPONSE1*, is the variable based on CT-scans and shows how the patients responded to a specific treatment. The other feature is made by subtracting two date variables of *DATETRM1* and *DATEDIAG* namely *TIMETOTRM1*. *DATETRM1* is the date of first treatment after diagnosis and *DATEDIAG* is the date of diagnosis which makes the new feature as the number of days between the first treatment and diagnosis.

The new dataset consists of two blocks with 71 samples, 62 and 27 features in every block respectively. It should also be noted that, although we apparently added two features to the first block, due to nominal type of *RESPONSE1* with 4 levels (referring to table 5.4), we practically include 5 more features considering *TIME*-

TOTRM1, as well.

Training RENT on this dataset, we came up with the results being summarised in tables 5.9 and 5.10 for the first and second block. It should also be noted that having a fixed seed value [9] helps us having the same test and training splits so we can compare the results with the previous ones. In other words, having the same splits with lower number of samples, we are able to compare the performance of the model and see if the new dataset with two new features can actually help explaining the target variance better or not. As a reminder, `random.seed` is a function which is used to reproduce the output for several times. [9]

Same as before the parameters used in RENT training are $\tau_1 = 0.9$, $\tau_2 = 0.9$, $\tau_3 = 0.975$, which are the default values of this package. Also, the number of ensemble models $k = 700$ does not change.

Splits	Selected feature(s)	R^2	RMSEP (error)
Split 1	PRIMTUM-Colon, SURGMET, RESPONS1-Complete Response (CR)	0.26	712.6
Split 2	PRTUMRES	-0.31	979.1
Split 3	SURGMET, CGA1-Negative, RESPONS1-Complete Response (CR)	-0.22	807.5
Split 4	SURGMET, SMOKHAB-Unknown, CGA1-Negative, SOM-LIVER,RESPONS1-Complete Response (CR), TIMETOTRM1	0.44	872.8
Split 5	RESPONS1-Complete Response (CR)	0.46	405.3
Split 6	PRIMTUM-Colon, SURGMET, CGA1-Negative, CGA1-Strongly Positive,SOM-LUNG, TIMETOTRM1	0.56	728.3
Split 7	PRIMTUM-Colon, SURGMET, RESPONS1-Complete Response (CR)	0.38	422.4
Split 8	RESPONS1-Complete Response (CR), RESPONS1-Progressive Disease (PD)	0.07	1095.9

Table 5.9: *Selected features and model performance based on the second approach for the first block*

Considering the average performance among different splits based on the data with 71 samples with respect to information in tables 5.9 and 5.10, we have R^2's of 0.20 and -0.36 for the first and second block respectively. Although the performance of the model for the second block did not dramatically increase, we see a very significant improvement on the performance of the models in the first block. Thus considering two new features with the cost of removing 9 samples actually helped having a better model with respect to its performance.

One of the practical features of RENT is its ability to give us information about every sample in terms of their contribution in the model to explain the variance of the target.[16] In other words, using `get-summary-objects()` function returns the average absolute error for each object. This attribute helped us moving to the third approach which leads to even better model performance.

Splits	Selected feature(s)	R^2	RMSEP (error)
Split 1	CGA2-Normal, LACTDHDR-Not Done	-0.33	954.1
Split 2	HIAA-Normal, CGA2-Normal	-0.15	918.9
Split 3	CGA2-Normal, LACTDHDR-Not Done	-0.23	811.8
Split 4	CGA2-Normal, LACTDHDR≥Normal ≤2UNL, ALKPHSPH≥Normal ≤ 3 UNL	-0.03	1191.6
Split 5	CGA2-Normal	-0.94	768.8
Split 6	CGA2-Normal, LACTDHDR-Not Done	-0.01	1101.2
Split 7	CGA2-Normal, HMGLBN≤11 g/dL, LACTDHDR-≥Normal ≤ 2UNL	-0.9	741.8
Split 8	CGA2-Normal, LACTDHDR≥ 2UNL, LACTDHDR-Not Done, WHITEBLD≥ $10x109/L$	-0.3	1297.6

Table 5.10: *Selected features and model performance based on the second approach for the second block*

5.2.4 Selected features based on the third approach

This approach has three main steps:

- Removing the samples that cause too much noise and error in the model
- Transformation of the target vector since the transformed response yields better results
- Removing some levels of the some of features, suggested by the experts

Determining the samples causing too much error

Using `get-summary-objects()` function [16], we can see which observations caused the highest average absolute error. Take table 5.11 into consideration. The first column is the sample number. The second column is the average number of times when the sample has been selected to be in the test split. For example, among $K = 700$ generalised liner models in the RENT, observation number 21 had been selected 184 times on average among 8 splits. The last column tells us the average absolute error of the corresponding sample. Note that the table is sorted based on the last column, and contains the first 40 samples. Information for the rest of the samples can be found in section 7.

After obtaining the results summarised in table 5.11, the very first thing which comes to mind is which of the objects might be candidate to be disregarded in the data. Considering the average and standard deviation of absolute error in the last column for all of the observations, the first three samples of 21, 42, 43 are those with highest absolute errors. In other words, using 3σ rule [31], standard deviation of the mentioned samples are greater than $\bar{X} + 3 \times \sigma$ (X is the absolute error here), so they are samples with highest absolute error in our dataset.

Obs #	# test	ABS error
21	184	5145.9
42	184	3987.0
43	195	3619.1
40	189	2733.1
47	202	2341.9
29	180	2221.7
62	188	1783.3
44	180	1637.0
41	184	1510.1
15	184	1490.0
1	184	1450.0
56	178	1420.7
64	193	1420.3
3	186	1376.7
20	184	1339.4
46	190	1298.1
53	177	1254.9
36	188	1191.1
8	187	1106.1
35	195	1098.8
21	184	5145.9
65	187	1086.9
4	198	1066.9
7	181	1038.2
26	178	1026.2
57	178	952.0
28	178	919.8
48	181	914.2
18	189	913.2
50	188	879.1
12	181	871.2
33	182	870.0
49	182	868.3
69	169	866.4
17	176	804.9
37	185	790.0
39	193	786.7
38	183	784.9
58	186	783.7
59	181	782.7

Table 5.11: *Summary object of 40 samples*

Referring to chapter 4.4.3, RENT helped us identifying observations which deviate from the other samples. The case which PCA and *Hotelling's* T^2 statistic failed to identify. Therefore, a solution for improving the previous model is to disregard these three samples and have a new dataset with 68 samples. As a reminder, the third approach is a complement approach of the second one therefore we run RENT on the dataset including not only previous variables but also two features of *RESPONSE1*, *TIMETOTRM1* with sample size of 68 observations, three items of 21, 42, 43 are being removed from it.

In the next step we will consider transformation on the target to obtain the best possible result with respect to performance of the models.

Transformation of the target vector

As has been already mentioned in chapter 2.4.2, sometimes uneven distribution of target can lead to a model with too much error. In our study, the target vector, defined as the number of days between the diagnosis and last observation of the patient, has a distribution demonstrated in figure 5.4. As it can be seen in the plot, the distribution is right skewed. As a reminder, skewness is defined as deviation from symmetrical distribution of a random variable. [2] On average the model performance on the first block is around 0.39 and the second block is -0.11 when using untransformed target with the dataset which the noisy samples have been

[2] `https://www.investopedia.com/terms/s/skewness.asp`

removed from it. Although the performance has experienced boosting when comparing to the second approach, working with transformed target even yields better results which is explained in the coming subsections.

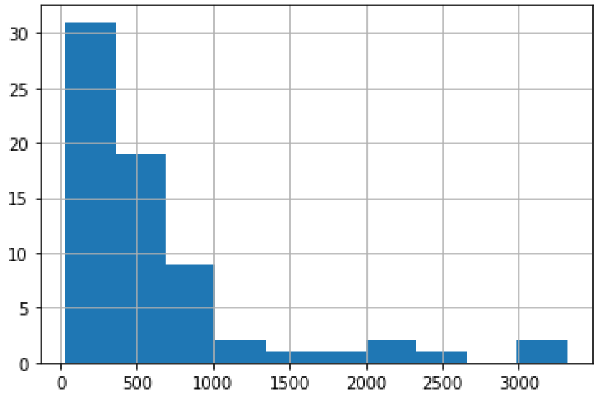

Figure 5.4: *The target distribution*

Having Box-Cox transformation (referring to chapter 2.4.2 for detailed explanation) we come up with a new vector of transformed target. The distribution of this vector is demonstrated in the figure 5.5 which is closer to a symmetric distribution. Now the new transformed target is ready to be used in the next steps of feature selection and main multi block modelling implementation.

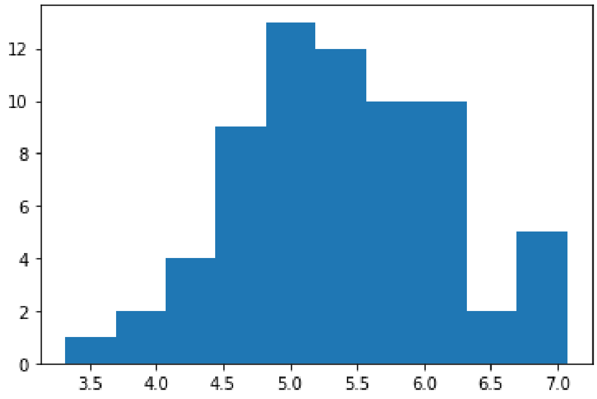

Figure 5.5: *Distribution of transformed target*

Removing some features according to experts' suggestions

After obtaining the results, experts believed that some of the levels of some features can be excluded from the data. Therefore by ignoring those we eventually come up with the results, summarised in table 5.12 and 5.13 for the first and second blocks respectively. On average the model performance on the first block is around

0.34 and the second block is 0.21. For two of the splits of the first block, the model performance based on R^2 (detailed explanation about this metric is given in chapter 2.3.1) even reaches to 0.68 which is a very good improvement.

Splits	Selected feature(s)	R^2	RMSEP (error)
Split 1	SURGMET, CGA1-Negative, RESPONS1-Complete Response (CR), RESPONS1-Progressive Disease (PD), TIMETOTRM1	0.34	0.55
Split 2	PERFSTAT-WHO 0, RESPONS1-Complete Response (CR), RESPONS1-Progressive Disease (PD), TIMETOTRM1, CGA1-Negative	0.30	0.67
Split 3	RESPONS1-Complete Response (CR), RESPONS1-Progressive Disease (PD), TIMETOTRM1	0.22	0.58
Split 4	SMOKHAB-Smoker, RESPONS1-Complete Response (CR), SURGMET, RESPONS1-Complete Response (CR)	0.41	0.68
Split 5	CGA1-Negative, RESPONS1-Complete Response (CR), RESPONS1-Progressive Disease (PD), TIMETOTRM1	0.67	0.42
Split 6	SURGMET, PERFSTAT-WHO 0, RESPONS1-Complete Response (CR), RESPONS1-Progressive Disease (PD)	0.03	0.86
Split 7	RESPONS1-Complete Response (CR), RESPONS1-Progressive Disease (PD), TIMETOTRM1	0.48	0.53
Split 8	SMOKHAB-Smoker, PERFSTAT-WHO 0, RESPONS1-Complete Response (CR), RESPONS1-Progressive Disease (PD), TIMETOTRM1	0.27	0.62

Table 5.12: *Selected features and model performance based on the third approach for the first block*

Validation study of the models yielded from RENT in the third approach

Referring to section 4.3.2, figures 5.14 and 5.15 demonstrate the validation study of the models yielded by RENT for the first and second block, respectively. Noted that for every 8 splits we can have a model and the model can be validated.

As the figures show, for most of the cases, RENT feature selection (red line is the prediction score of RENT models) is doing a better job than models of random selection of features (blue curves) and permuted test labels (green curves). (so-called VS1 and VS2 respectively). Thus, we rely on the results of the RENT feature selection and will use all the features selected at least one time. Table 5.16 shows which features eventually would be used in our final multi block model.

Splits	Selected feature(s)	R^2	RMSEP (error)
Split 1	CGA2-Normal, HMGLBN≤11 g/dL LACTDHDR≥ 2UNL	-0.04	0.69
Split 2	CGA2-Normal, LACTDHDR≥ 2UNL, ALKPHSPH-Normal	0.39	0.63
Split 3	CGA2-Normal, HMGLBN≤11 g/dL LACTDHDR≥ 2UNL	0.37	0.52
Split 4	CGA2-Normal, LACTDHDR≥ 2UNL, ALKPHSPH-Normal	0.37	0.71
Split 5	CGA2-Normal, LACTDHDR≥ 2UNL, ALKPHSPH-Normal	0.18	0.66
Split 6	CGA2≤Normal ≤ 2UNL, CGA2-Normal, LACTDHDR≥ 2UNL	0.02	0.86
Split 7	CGA2-Normal, LACTDHDR≥ 2UNL, PLATELTS≥400x10x9/L	0.20	0.66
Split 8	CGA2-Normal, HMGLBN≤11 g/dL, ALKPHSPH-Normal LACTDHDR≥ 2UNL, ALKPHSPH≥3 UNL	0.18	0.66

Table 5.13: *Selected features and model performance based on the third approach for the second block*

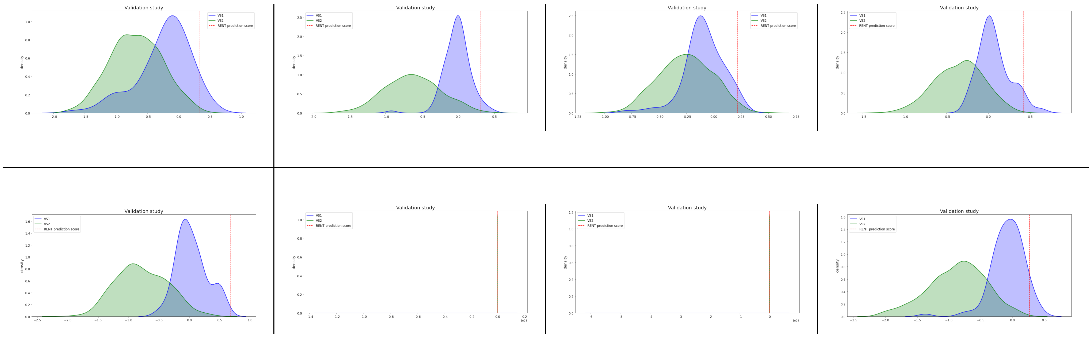

Table 5.14: *Validation study of RENT models for the first block*

Validation study of RENT models for the first block

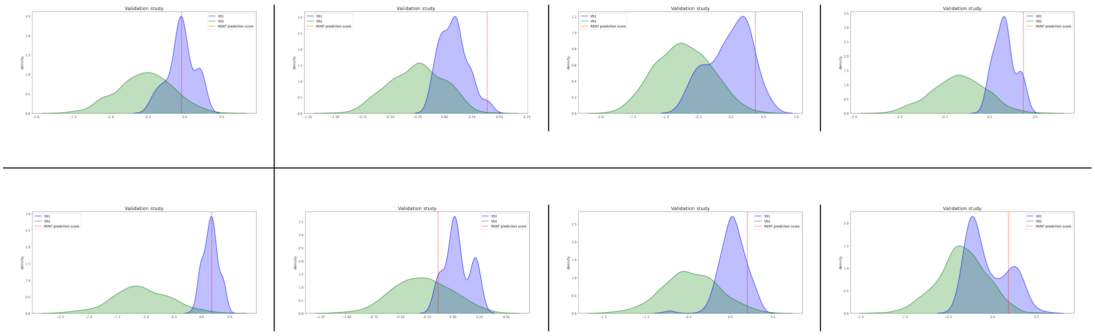

Table 5.15: *Validation study of RENT models for the second block*

Validation study of RENT models for the second block

Selected features in the first block
SURGMET
SMOKHAB-Smoker
CGA1-Negative
PERFSTAT-WHO 0
RESPONS1-Complete Response (CR)
RESPONS1-Progressive Disease (PD)
TIMETOTRM1

Selected features in the second block
Normal$\leq$CGA2 $\leq$ 2UNL
CGA2-Normal
HMGLBN$\leq$11 g/dL
LACTDHDR$\geq$ 2UNL
PLATELTS$\geq$400x10x9/L
ALKPHSPH$\geq$3 UNL
ALKPHSPH-Normal

Table 5.16: *Final selected features*

5.3 Multi block analysis

After preprocessing the data, having features ready to be modelled on the target, it is the time to use response oriented sequential to finally model the target to the selected features in every block. It should be reminded that the transformed Y is also used in this section. In case of needing any prediction from the model we can reversely do the computations on the predicted values in order to change them to the original values.

5.3.1 Features and the target

The model we are willing to obtain is visualised in figure 5.6. As it is showing, we have 1 target vector modelling on 2 blocks of features, including 7 variables in each.

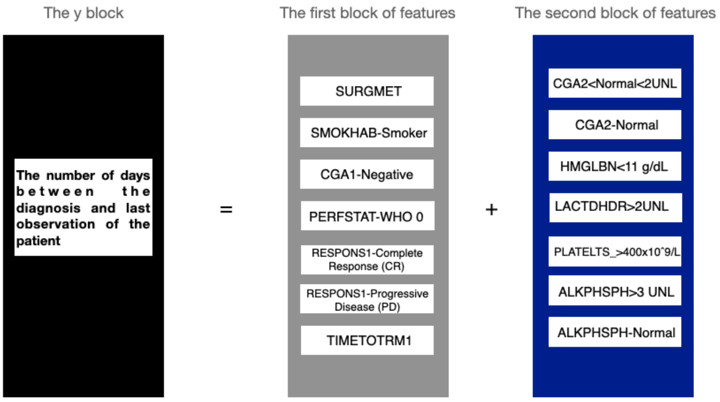
The y block
The first block of features
The second block of features
The number of days between the diagnosis and last observation of the patient
=
SURGMET
SMOKHAB-Smoker
CGA1-Negative
PERFSTAT-WHO 0
RESPONS1-Complete Response (CR)
RESPONS1-Progressive Disease (PD)
TIMETOTRM1
+
CGA2<Normal<2UNL
CGA2-Normal
HMGLBN<11 g/dL
LACTDHDR>2UNL
PLATELTS_>400x10ˆ9/L
ALKPHSPH>3 UNL
ALKPHSPH-Normal

Figure 5.6: *Our multi block problem demonstration*

Using Response Oriented Sequential Alternation (ROSA) multi block method, we finally come up with the results explained in the subsequent sections.

Using 2 components, around 74% of target variance is being explained in cross-validation sets. Therefore the model has dimension of $68 \times 1 \times 2$ representing number of samples, responses and components respectively. The first component explains 67% and the second one explains around 7% of the target variance. It should also be noted that, most of the variance in the first and second component is from the first and second block, respectively. (Figure 5.7 shows this case). Based on the figure 5.7 we can also conclude that most of the information of the target being explained by the blocks is hidden in the first one. Regarding model error, RMSEP is around 0.45 and basically the first component summarised all the information from the first block and likewise for the second component. As a note, this RMSEP is with respect to the Box-Cox transformed target, so it cannot be interpreted in the same scale of untransformed target as number of days.

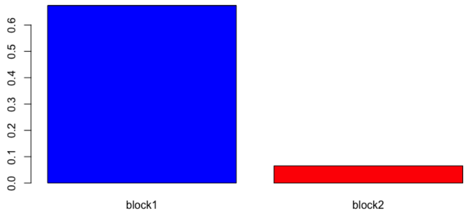

Figure 5.7: *Block-wise explained variance*

Figure 5.8 shows the direction of the features impacting the target. Interesting fact about these results is that two features of *CAG2* and *ALKPHSPH* with their normal level have positive relationship with the target. This means that if the patients have normal amount of chromogranin and alkaline phosphatase (more detailed explanation about the features in chapter 3), they can live longer. Moreover to this, considering the *RESPONSE1* variable as the indicator of treatment response, those who still had the progressive disease after treatment live shorter (negative relationship with the target) and those who responded completely after treatment (*RESPONSE1-Complete-Response*) have a strong positive relationship with target and means could live longer. Feature *SURGMET* also has a strong positive relationship and it shows if the patient could have a surgery this can help them to live longer. *PERFSTAT* was the variable indicating the performance status of the patients. Apparently based on our results not only this variable is among significant ones but also based on the information given in tables 5.2 and 5.8, patients who could carry out all normal activities without restriction could live longer and WHO-0 level of the *PERFSTAT* feature has strong positive relationship with our target. Abnormal levels of *CAG2*, *HMGLBN*, *PLATELTS* and *LACTDHDR* lead to shorter living days (due to negative relationship with the target).

Two features of *SMOKHAB-Smoker* and *TIMETOTRM1* have positive significant relationship with the target. The first one obviously says that patients who smoke regularly have the chance of living longer. The second one declares the patients

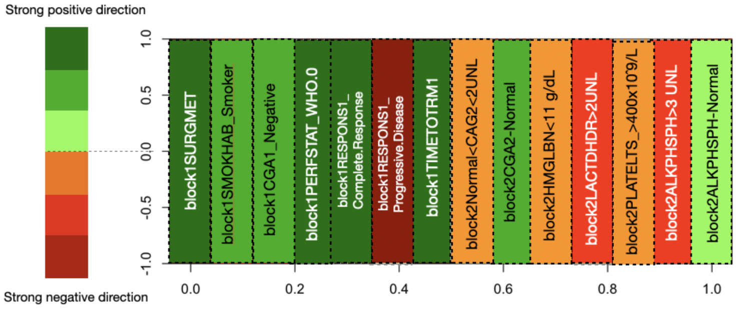

***Figure 5.8:** Coefficients' direction in ROSA model*

who received the first treatment long after the diagnosis also can live more days. These two results seem a bit weird at first but considering statistical paradoxes in data science we can explain the cause. [3] Existence of a mediation variable (the third variable which interferes in the middle [4]) might be the cause. More specifically, the guess is that for example smoking patients have a higher chance to be hospitalised sooner. Those who are hospitalised and are receiving treatment earlier may live longer days . Figure 5.9 shows it better. (It should be note that this assumption is just a guess and has the potential to be studied). The second feature which has an unexpected relationship in terms of its direction with the target is *TIMETOTRM1*. The first thought is that the patients who received earlier treatments after the diagnosis should live longer. In other words the lower number of days between diagnosis and first treatment, the higher days they live. However our model says something different and shows their relationship as a positive one. (Higher number of days between first treatment and diagnosis leads to higher number of living days). In explanation of this phenomenon, the assumption is that the patients who got the first treatment earlier might have a severe condition after their first diagnosis. So basically earlier receiving of the first treatment might be due to the severity of their conditions and these patients mainly do not have high chance to live longer. The solution which comes to mind is that, having a classification initially between the patients based on the severity of their diagnosis and then trying to model each class separately can help having better results. However as it had been already discussed these are assumptions and have so much potential to be studied and scrutinised.

Features' importances

Based on what we had so far, we can order the features based on their strength of impact on the target. In other words using regression coefficients which are

[3] https://www.kdnuggets.com/2021/04/top-3-statistical-paradoxes-data-science.html

[4] https://en.wikipedia.org/wiki/Mediation_(statistics)

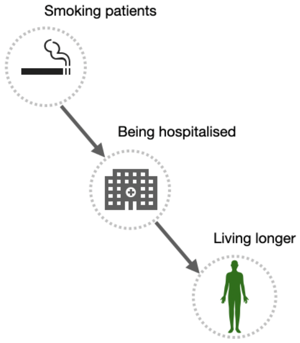

Figure 5.9: *Mediation variable*

found at the end (sorting them by their absolute value) relating matrix of features **X** to Y (vector of target) using our 2 components, we will not only find the most important features but also can report the direction of relationship features have with the target. ROSA summarised the mentioned information in a table like table 5.17. Not surprisingly, most of the very significant features based on the regression coefficients come from the first block. The first 6 features in the table are from the first block. We actually expect that, since most of the target variance has been explained by the components extracted from the first block (figure 5.7). Moreover to this, it seems the variable *RESPONSE1* is a real important feature, by which two levels of it had been selected by RENT and ROSA also identified these two levels as the first two significant features. As a reminder this feature shows how the patients respond to the treatment. If the treatment leads to ongoing disease, patients will live shorter life however if the treatment has a complete response on the patients, it will significantly help them living longer.

Features	Regression coefficients	Direction
RESPONS1-Progressive.Disease	$0,27$	Negative
RESPONS1-Complete.Response	$0,23$	Positive
PERFSTAT-WHO.0	0.20	Positive
TIMETOTRM1	0.19	Positive
CGA1-Negative	0.14	Positive
LACTDHDR>2UNL	0.11	Negative
SMOKHAB-Smoker	0.09	Positive
CGA2-Normal	0.09	Positive
HMGLBN<11 g/dL	0.07	Negative
ALKPHSPH-Normal	0.05	Positive
Normal<CAG2<2UNL	0.05	Negative
PLATELTS>400x10x9/L	0.02	Negative

Table 5.17: *Regression coefficients*

5.3.2 Loadings and scores in multi block analysis

One of the interesting aspects of multi block analysis and specially ROSA is that using loadings and scores plot we can study the patients and the features deeper down. Take the figure 5.10 into consideration. Loadings (more detailed explanation in section 2.1) could help us having a categorisation among our significant features. Based on figure 5.10 around 16% of the variance of the dataset is being explained by two components. These two components also tell us that we can put five features of *ALKPHSPH>3 UNL*, *HMGLBN<11 g/dL*, *LACTDHDR>2UNL*, *PLATELTS>* $400x10x9/L$ and *RESPONS1- Progressive Disease* in the one category. Two features of *RESPONS1-Complete.Response* and *TIMETOTRM1* in another category and the rest in the third category. In simple words, this plot is doing a clustering of features based on positive and negative correlation with the response.

The above result is actually in good harmony with what we have already obtained in figure 5.8. All the features with either red or orange colour (having negative relationship with the target) have been categorised in the first class. Two features of *RESPONS1-Complete.Response* and *TIMETOTRM1* have strong positive relationship with the target and here they are also in the same category. The rest of the features are also in positive relationship with our response and are being classified together. Another interesting fact about this plot is that two features of *ALKPHSPH* and *CGA2* with their normal levels have loadings overlapping each other.

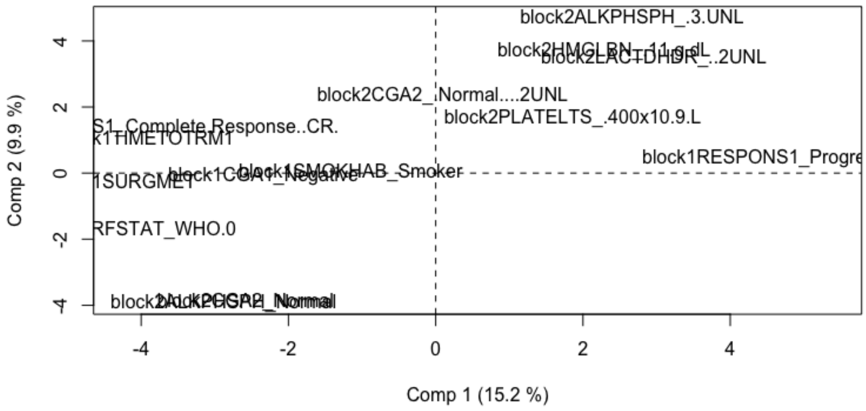

Figure 5.10: *Loadings plot*

Another useful plot is the loadings and scores plot. Figure 5.11 shows it. Based on this plot we can see samples with number of 66, 46, 33 have normal level of *ALKPHSPH* and *CGA2*. Sample number 45 represents the patient with performance status of 0. (Referring to table 5.2 for information about the status).

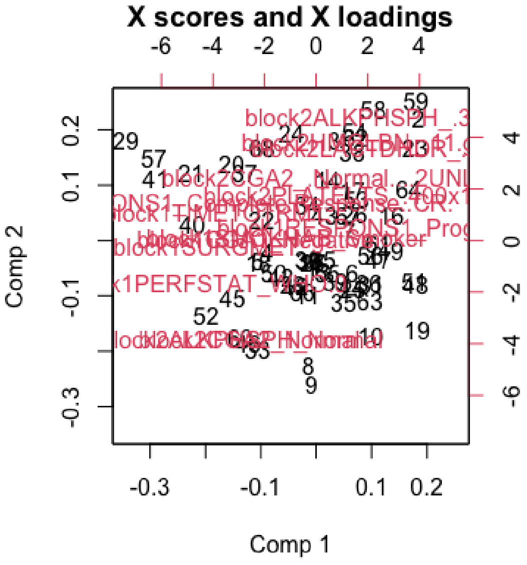

Figure 5.11: *Loadings plot*

Chapter 6

Discussion

In this chapter we will suggest some more potentials of this research in order to obtain better and more accurate results.

6.1 Dataset

6.1.1 Features and samples

The dataset used in this study included around 90 features. However due to high number of missing items, we had to exclude many variables. What can be suggested at the first place is to complete the dataset, filling the missing values; so that it is feasible to include more features. The new features might either be identified as important ones or change the situation for other features, in a way to be selected as significant ones or the other way around.

In addition to the missings' problem, we can also see if the first block can be categorised into two blocks. In the last meeting with the experts, they suggest that the first block has this potential to be break into more blocks. However due to time shortage, we did not make this change. Even-though it can lead to an improvement in terms of block creation and eventually model performance.

6.1.2 The target

What we experienced in this study was that changing the target from binary to continuous made a significant contribution in model performance. Therefore having several targets summarised in a block of response instead of one dimensional target vector , might also help having better model. Besides, nonetheless Box-Cox transformation improved our model, there are plenty of other transformation techniques which might help yielding even better results.

6.2 Methods

6.2.1 Detection of outliers

This study used three methods of PCA, *Hotelling's* T^2 statistic and RENT feature selection, to find the outliers. Although RENT helped identifying them, we can make use of other available methods such as fuzzy logic-based outlier detection [32] or cluster analysis-based outlier detection [33] for a better detection of them.

6.2.2 Feature selection

In this research we mainly used Repeated Elastic Net Technique (RENT) to select important features. As been discussed in section 4.3.1, this method has three parameters of τ_1, τ_2 and τ_3. Due to time shortage we used default values of $\tau_1 = 0.9$, $\tau_2 = 0.9$, $\tau_3 = 0.975$. However these parameters can be tuned and the best combination of them can be applied to obtain a better model. Noted that *Sickit-learn* [9] in Python has a pre-defined function of `GridSearchCV()` to search over specified parameter values for any estimator.

In addition to tuning the parameters of RENT, there are also other techniques of feature selection that can be applied. Having results of other methods we can also evaluate performance of RENT feature selection to see if this is the optimal technique for our problem to select the most important features.

6.2.3 Multi block analysis

ROSA [8] was the multi block method used in this study. However as is mentioned in chapter 2.2, there are many other multi block techniques such as SO-PLS [7] or MB-PLS [6]. It can also examine if other multi block techniques yield the same result as ROSA did or not. Different methods can also be compared with each other to see which one leads to a better model in terms of its performance.

Chapter 7

Conclusions

This research proved the importance of data preprocessing. Using the real world data, we conclude that more than 70% of work load is about preprocessing. Having regular contact with the experts of the field was also a very important part of preprocessing step since they always had some ideas about the way our dataset was built.

Moreover to that, we observed that in a wide dataset where the number of features exceeds the number of samples, feature selection is a vital step that must be taken. If the feature selection part was excluded, we would not be able to obtain a model with a relatively acceptable performance.

The real world data most likely do not include many samples. The importance of cross-validation was also observed in this research. Cross-validation helped us making the most use of the available samples.

Target is a very important variable, therefore it is necessary to define a variable which not only measures our target but also does not have imbalanced classes. Symmetric distribution of response is also vital. Therefore we should always be aware of having some proper transformations to have a relatively even distribution of the target. This issue was also something that the current research proved.

Last but not least is the multi block importance. Expressly, this research proved that problems including blocks, should be treated as multi block ones. If we did not use multi block method to analyse the data in this study, we could not achieve desirable results. As is been mentioned in chapter 2.2, in multi block datasets the features are not defined as single variables and instead, the data includes blocks of multiple relevant features. Multi block methods such as ROSA, takes this issue into consideration and make the model. Therefore the importance of using multi block methods has been proved by this research, as well.

Bibliography

[1] O. Celik, “A research on machine learning methods and its applications,” 09 2018.

[2] T. Ayodele, *Machine Learning Overview*, 02 2010.

[3] M. Iqbal and Z. Yan, “Supervised machine learning approaches: A survey,” *International Journal of Soft Computing*, vol. 5, pp. 946–952, 04 2015.

[4] D. Pirouz, “An overview of partial least squares,” *SSRN Electronic Journal*, 10 2006.

[5] K. Dunn, *Process Improvement Using Data*, 2021.

[6] K. H. L. Age K. Smilde, Tormod Næs, *Multi block Data Fusion in Statistics and Machine Learning - Applications in the Natural and Life Sciences*. Wiley, 2021.

[7] O. Tomic, J. Niimi, T. Naes, D. W. Jeffery, S. E. Bastian, and P. K. Boss, “Application of sequential and orthogonalised-partial least squares (so-pls) regression to predict sensory properties of cabernet sauvignon wines from grape chemical composition,” *ELSEVIER, Food Chemistry*, vol. 8, pp. 195–202, 2018.

[8] K. H. Liland, T. Naes, and U. G. Indahl, “Rosa - a fast extension of partial least squares regression for multiblock data analysis,” *WILEY, Journal of Chemometrics*, vol. 12, pp. 1–12, 2016.

[9] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[10] P. S. L. Breiman, “Submodel selection and evaluation in regression: The x-random case,” 1992.

[11] R. R. R. Bharat Rao, G. Fung, “On the dangers of cross-validation. an experimental evaluation,” 2008.

[12] J. F. T. Hastie, R. Tibshirani, *The Elements of Statistical Learning*. Springer, 2009.

[13] T. H. R. T. G. James, D. Witten, *An Introduction to Statistical Learning*. Springer, 2013.

[14] V. M. Sebastian Raschka, *Python Machine Learning*, 2017.

[15] J. Osborne, “Improving your data transformations: Applying box-cox transformations as a best practice,” *Pract Assess Res Eval*, vol. 15, pp. 1–9, 01 2010.

[16] K. H. L. U. G. I. C. M. F. O. T. Anna Jenul, Stefan Schrunner, “Rent - repeated elastic net technique for feature selection,” vol. 16, 2020.

[17] A. C. Alice Zheng, *Feature Engineering for Machine Learning*. O’REILLY, 2018.

[18] C. P. Kedar Potdar, Taher Pardawala, “A comparative study of categorical variable encoding techniques for neural network classifiers,” *nternational Journal of Computer Applications*, vol. 175, no. 4, pp. 7–10, 2017.

[19] D. B. R. Roderick J A Little, *Statistical Analysis with Missing Data.* John Wiley, 1986.

[20] K. G.-O. Stef van Buuren, “Multivariate imputation by chained equations in r,” *Journal of Statistical Software*, vol. 45, pp. 1–67, 2011.

[21] M. C. M. Gustavo E. A. P. A. Batista, “A study of k-nearest neighbour as an imputation method,” 2002.

[22] J. Li, K. Cheng, S. Wang, F. Morstatter, R. Trevino, J. Tang, and H. Liu, “Feature selection: A data perspective,” *ACM Computing Surveys*, vol. 50, 01 2016.

[23] M. Müller, “Generalized linear models,” 02 2004.

[24] B. N. Shima Kashef, Hossein Nezamabadi-pour, “Multilabel feature selection: A comprehensive review and guiding experiments,” 2018.

[25] D. Chicco and G. Jurman, “The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation,” *BMC Genomics*, vol. 21, 01 2020.

[26] V. Barnett and T. Lewis, *Outliers in Statistical Data.* John Wiley, 1994.

[27] C. Pascoal, M. R. Oliveira, A. Pacheco, and R. Valadas, *Detection of Outliers Using Robust Principal Component Analysis: A Simulation Study*, 12 2010, vol. 77, pp. 499–507.

[28] K. Jajuga and M. Walesiak, *Standardisation of Data Set under Different Measurement Scales*, 01 2000, pp. 105–112.

[29] K. Alaluusua, “Outlier detection using robust pca methods,” Ph.D. dissertation, 08 2018.

[30] P. F. Kurt Varmuza, *Introduction to Multivariate Statistical Analysis in Chemometrics*, 2009.

[31] R. Lehmann, “3sigma-rule for outlier detection from the viewpoint of geodetic adjustment,” *Journal of Surveying Engineering*, vol. 139, pp. 157–165, 11 2013.

[32] S. Cateni, V. Colla, and M. Vannucci, “A fuzzy logic-based method for outliers detection,” 01 2007, pp. 605–610.

[33] Sheng-yizJiang and Q.-b. An, “Clustering-based outlier detection method,” vol. 2, 11 2008, pp. 429 – 433.

Appendix

Appendix A: Summary object of the rest of the samples

Obs #	# test	ABS error
23	183	781.8
27	186	780.7
25	182	777.0
5	187	773.8
16	177	769.4
22	188	765.6
60	183	764.0
34	188	762.7
55	178	754.5
66	190	747.4
31	176	745.9
54	179	743.9
13	174	715.4
70	199	714.4
68	186	712.9
11	180	708.6

Obs #	# test	ABS error
14	175	701.8
9	189	675.9
52	184	665.1
45	178	645.7
0	177	639.8
30	190	631.9
67	179	588.5
51	194	582.2
19	176	578.4
61	182	564.7
63	180	522.1
2	186	476.5
24	182	452.9
32	183	434.7
10	180	427.6

Norges miljø- og biovitenskapelige universitet
Noregs miljø- og biovitskapelege universitet
Norwegian University of Life Sciences

Postboks 5003
NO-1432 Ås
Norway