PREDICTIVE MODELS FOR CATALYST DEVELOPMENT

A THESIS SUBMITTED TO
THE FACULTY OF ARCHITECTURE AND ENGINEERING
OF
EPOKA UNIVERSITY

BY

REBEKA KONDI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

JULY, 2022

Approval sheet of the Thesis

This is to certify that we have read this thesis entitled **“Please input thesis title here !!!!”** and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

____

Assoc. Prof. Dr. Name Surname
Head of Department
Date: Month, dd, year

Examining Committee Members:

Assoc. Prof. Dr. Name Surname	(Computer Engineering)	________________
Dr. Name Surname	(Computer Engineering)	________________
Dr. Name Surname	(Computer Engineering)	________________

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name Surname: Rebeka Kondi

Signature: ______________

ABSTRACT

PREDICTIVE MODELS FOR CATALYST DEVELOPMENT

Kondi, Rebeka
M.Sc., Department of Computer Engineering
Supervisor: Dr. Maaruf Ali

With these hard times that we are living after covid, inflation but also problems like fertilizer shortage and supply of chain issues, has made everyone turn their attention to better, more affordable, faster, and organic solution almost in every field of science and not only.

The inspiration for this project was found on the BioSPRINT project, where the target reaction is the simultaneous dehydration of multiple C5 and C6 sugars to produce 5-HMF and FUR. The objective was to find machine learning (ML) models that would speed up the discovery of catalysts using high-throughput (HTP) screening techniques. Maximum activity for the conversion of complex sugar combinations is sought, with the best selectivity for the major products of interest.

The three additional models used are generalised boosted regression modelling, extreme gradient boosting and boosted generalised additive models for location, scale, and shape.

The results show that XGBoost has the best performance overall. All the models performed poorly in the case of Selectivity. Another approach for this response is to apply a transformation on the response variable. The performance of these models can be potentially improved by adding new “catalytic-informed” features, that will be engineered based on the expert knowledge about the problem.

Keywords: *Machine Learning, Catalysis, Predictive Modelling, Variable Selection, Solvent, Gradient Booting.*

ABSTRAKT

MODELE PARASHIKUESE PER ZHVILLIMIN E KATALIZATOREVE

Kondi, Rebeka

Master Shkencor, Departamenti i Inxhinierisë Kompjuterike

Udhëheqësi: Dr. Maaruf Ali

Me këto kohë të vështira që po jetojmë pas Covid-it, inflacioni por edhe problemet si mungesa e fertilizatorëve dhe zinxhiri i furnizimit, ka bërë që të gjithë ta kthejnë vëmendjen drejt zgjidhjeve më të mira, më të përballueshme, më të shpejta dhe organike pothuajse në çdo fushë të shkencave dhe jo vetëm. .

Frymëzimi për këtë projekt erdhi nga projekti BioSPRINT, ku rezultati i synuar është dehidratimi i njëkohshëm i sheqernave C5 dhe C6 për të prodhuar 5-HMF dhe FUR. Objektivi ishte gjetja e modeleve të machine learning (ML) që do të përshpejtonin zbulimin e katalizatorëve duke përdorur teknikat e shqyrtimit me performancë të lartë (HTP).

Tre modelet shtesë të përdorura janë: generalised boosted regression modelling, extreme gradient boosting dhe boosted generalised additive models për vendndodhjen, shkallën dhe formën.

Rezultatet tregojnë se XGBoost ka performancën më të mirë në përgjithësi. Të gjitha modelet performuan dobët në rastin e Selektivitetit. Një qasje tjetër për këtë zgjidhje është aplikimi i një transformimi në variablin te varur. Performanca e këtyre modeleve mund të përmirësohet potencialisht duke shtuar veçori të reja në lidhje me katalizatorët, të cilat duhet te bazohen në njohuritë më eksperte rreth problemit.

Fjalët kyçe: *Machine Learning, katalizator, modele parashikuese, zgjedhja e variablave, tretës, Gradient Booting.*

I dedicate my thesis to my husband Dr. Soteris Solomou, that is close to me in every step of my life, my biggest blessing, and my biggest supporter.

ACKNOWLEDGEMENTS

I would like to express my special thanks to my supervisor Proof. Dr. Maarud Ali for his continues guidance, encouragement, motivation, and support during all the stages of my thesis. I sincerely appreciate the time and effort he has spent to improve my experience during my graduate years.

And again, to my valuable Dr. Soteris Solomou

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Introduction

The 20s have always been some hard years in the history of mankind, and if we have learned something from history is that it repeats itself. It looks like this century too we are passing some hard years. With the pandemic that looks like we passed it, with this sudden lifting of measurements and the supply chain problem, the fertilization problem, the shortage on personnel that have started on some of the biggest companies worldwide and the forecasting of the recession in the horizon in the upcoming 6-9 months, it looks like it has shifted the human's perspective, focus, and aim on finding more fast, affordable, and ecofriendly solutions in any field.

For better understanding of the thesis, it needs to be started first with some definition of concepts like:

Biomass energy, or energy derived from living things, has been utilized by humans since the first "cave men" used wood fires for cooking and warmth. Biomass is organic, which means it is made up of material derived from living beings like plants and animals. Plants, wood, and garbage are the most frequent biomass sources utilized for energy. Biomass feedstocks are what they're termed. Biomass energy can be a non-renewable source of energy.

Biorefining is the process of obtaining various value-added products from the valorization of biomass, while Process Intensification (PI) is a technique for generating significant gains in manufacturing and processing by lowering the equipment size to output ratio, reducing waste, and lowering energy usage.

Catalysis is the use of catalysts to alter chemical processes. For business reasons, the response is hastened or amplified. Because catalysts reduce the minimum energy required to initiate a chemical reaction, catalytic reactions are quicker than uncatalyzed reactions (activation energies).

This thesis details machine learning models that can be used in catalysis and aim to find relevant descriptors to connect experimental data with desired Figure of Merit (FOM).

Data analysts use Figures of Merit to describe a method. Precision, accuracy, sensitivity, linear dynamic range, detection limit, and selectivity are the six figures of merit that are utilized.

What was studied in this thesis is the selectivity which refers to a catalyst's tendency to favor favorable reactions over unwanted ones at a faster pace.

It has found inspiration on the BioSPRINT project, where the target reaction is the simultaneous dehydration of multiple C5 and C6 sugars to produce 5-HMF and FUR. The thesis builds on the work of, [1] by adding three additional models that were implemented on the project’s experimental data set. The three additional models used are generalised boosted regression modelling, extreme gradient boosting and boosted generalised additive models for location, scale, and shape.

This thesis is divided in 4 chapters. The organization is done as follows:

In Chapter 1, the introduction and model review. Chapter 2 includes the materials and methods, preprocessing, evaluation metrics and modeling framework. Chapter 3 consists of the results for each model. In Chapter 4, are the discussion and future work.

1.2 Models Review

1.2.1 Gradient Boosting

A high-level description is that it is a type of machine learning technique that utilizes previous models by combining them with the new ones to achieve the smallest prediction error. They are combined via their target outcomes which are compared against the other predictions. The impact on the overall prediction error is then assessed.

For example, a target outcome gets a high value when a small alteration in the prediction results to the error having a large drop. In the same sense, if a small change in the prediction does not impact the overall error, then the next target outcome is zero. The term gradient boosting comes from the gradient of the error against the prediction

1.2.2 Extreme Gradient Boosting

The idea behind this method is to generate decision trees in a sequential manner. The weights play a crucial role here. All the independent variables are given weights, which are then put into the decision tree. After that, the forecasts are obtained. The weight of variables that the tree predicted incorrectly is raised, and the variables are put into the second decision tree. These various classifiers/predictors are then combined to create a more powerful and accurate model. It may be used to solve issues including regression, classification, ranking, and user-defined prediction.

1.2.3 Generalized Additive models for location, scale, and shape

GAMSLSSs are a popular semiparametric modeling approach that regresses not only the expected mean but also every distribution parameter (e.g., location, scale, and shape) to a set of covariates, in contrast to conventional generalized additive models. They were introduced by [2] as a class of statistical models for regression problem univariate response.

Given a set of variables, GAMSLSS models have the benefit of not requiring the conditional distribution of the response variable to belong to the exponential family. Instead, you can choose from a large range of discrete, continuous, and mixed discrete-continuous distributions.

One further essential feature of GAMLSSs is the fact that every parameter of the conditional response distribution has its own predictor as well as link function. The

GAMLSS method makes it possible to model the regression of each distribution parameter on the covariates, in contrast to the normal GAM method, which often can only be used to model the conditional mean of the response variable (considering other distributional characteristics as fixed). Location, scale, skewness, and kurtosis are common distribution parameters, although degrees of freedom (of a -distribution) and zero inflation probability can also be modeled. The whole conditional distribution of a multiparameter model is thus tied to a set of predictor variables of interest in the GAMLSS technique. [3]

CHAPTER 2

MATERIALS AND METHODS

2.1 Data Description and preparation

The following section describes the data used in this project and presents the pre-processing prior to using it in the modelling part.

2.1.1 Description

In catalyst development, the goal is to link catalyst descriptors to FOMs. FOM is a quantitative index describing catalyst’s usefulness e.g., selectivity and conversion. Dataset of catalyst libraries consists of features that explain physicochemical properties of the materials such as electronic structure properties, physical properties, atomic properties etc.

The following figure shows the flow from the catalysts to the reaction conditions and finally to the figures of merit.

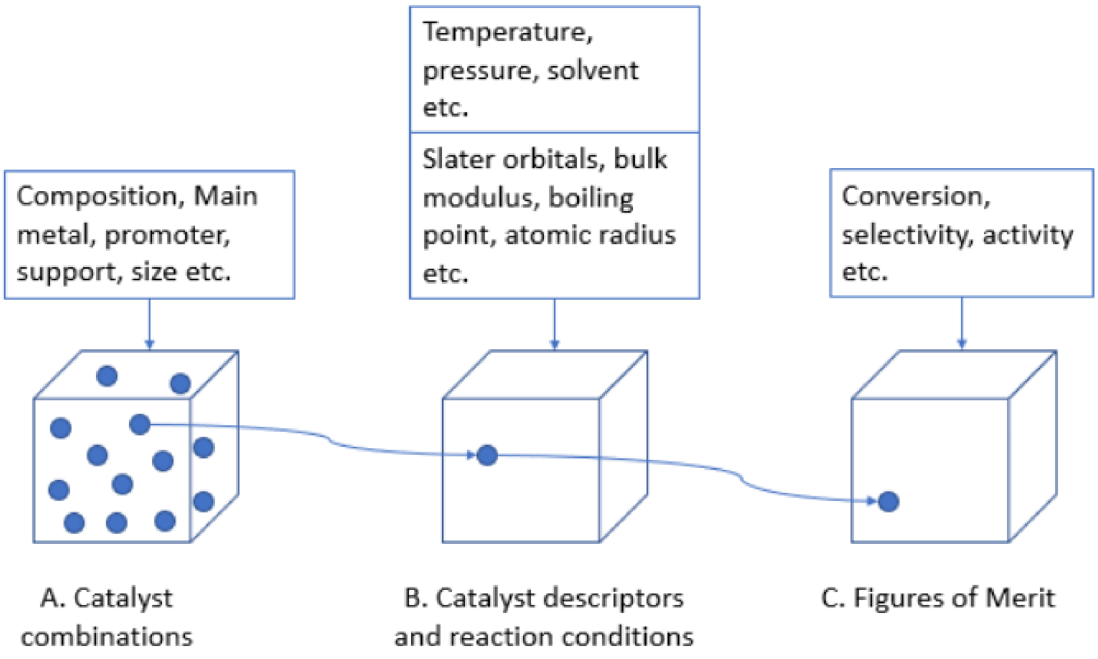

Temperature, pressure, solvent etc.
Slater orbitals, bulk modulus, boiling point, atomic radius etc.
Composition, Main metal, promoter, support, size etc.
Conversion, selectivity, activity etc.
A. Catalyst combinations
B. Catalyst descriptors and reaction conditions
C. Figures of Merit

Figure 1: Catalyst Flow

There are no libraries that map the properties of catalysts with the FOM for simultaneous dehydration reaction of C5 and C6 sugars to 5-HMF and furfural. The dataset provided for the experiment was obtained from [4] paper (based on hydrogenation reaction of 5-ethoxymethylfurfural). In addition, a set of properties for the main metals and promoters were provided. The data set consists of the following:

Conversion and selectivity percentages for two different solvents (diethyl carbonate and 1,4-dioxane), and three different temperatures (80, 100 and 120 °C).

Eight different main metals (Au, Cu, Ir, Ni, Pd, Pt, Rh, Ru) and six promoters (Bi, Cr, Fe, Na, Sn, W) that were used as catalysts. For each main metal and promoter, we have a set of properties that are used as input features to the machine learning models.

Slater-type orbitals values were obtained from [4]. In order to obtain the data some additional settings needed to be applied. Those settings are related to the chemical composition and structure of the elements involved in the experiment. Those are:

- Aluminium Oxide support in all observations.
- Main metal’s loading of 1 wt% and promoter 10 mol% related to main metal.
- Feedstock composition kept constant in the studied dataset.

Since the chemical properties and structure of the elements is outside the scope of this project, no further explanation or analysis will be presented.

As mentioned above, the aim is to use machine learning algorithms to predict the conversion and selectivity for the two solvents, using the properties of main metals and promoters as input features. The same naming convention for the response variables is also used:

- Conversion: conversion with diethyl carbonate solvent
- Selectivity: selectivity with diethyl carbonate solvent
- Conversion1: conversion with 1,4-dioxane solvent
- Selectivity1: selectivity with 1,4-dioxane solvent

2.2 Pre-processing

This step was important to understand the properties of the dataset and to format the data to be used in the machine learning algorithms. It consisted of visual inspection of the data, handling the missing values, applying any necessary transformations that would improve the fitting algorithm and remove any data that do not provide any useful information.

The first step was the inspection of the response variables. All four responses were percentages and took values in the interval [0,100]. As pointed out in [1], and from the plots in the diagonal of the figure 1 below, where the histograms with the densities superimposed for each response variable were plotted, it was seen that they do not follow normal distribution. Especially in the case of conversion and selectivity of the 1,4-dioxane solvent, where they were bimodal with most of the density at 0 and 100. Since the assumption of normality did not hold, it was needed to investigate more flexible regression frameworks, that could model response variables with more flexible distributions. One such framework is the Generalised Additive Models for Location Scale and Shape (GAMLSS), where they used a distributional regression approach, where all the parameters of the conditional distribution of the response variable were modelled using explanatory variables. GAMLSS could model the response variable using distributions that can be outside the exponential family.

For each pair of main metal and promoter at different temperatures, the properties of those elements were used to predict the conversion and the selectivity of the catalysis. Given this nature of the problem, the input features had many repeated values that made it hard to model the responses using linear relationships or spline functions to capture non-linear relationships. Hence, using tree-based methods looked a more reasonable approach. Also, due to the high dimensionality of the dataset (100+ descriptors and 4 response variables), gradient booting techniques could be used. An advantage of boosting was that the fitting of the model and the feature selection are performed simultaneously.

The dataset contained NAs for some values in the input features and for couple of values in the responses. Specifically, the values for conversion and selectivity of diethyl carbonate solvent for the pair Ir/W were missing. The corresponding rows were omitted for the given solvent. In addition, the following features were removed due to the NA values in some of the observations:

- m_Vickers_Hardness_MPa
- m_Mass_Mg_Susc_m3_kg
- m_Molar_Mg_Susc_m3_mol
- m_Volume_Mg_Susc
- m_Van_der_Waals_Radius_pm
- p_Vickers_Hardness_MPa
- p_Poisson_Ratio
- p_Mass_Mg_Susc_m3_kg
- p_Molar_Mg_Susc_m3_mol
- p_Volume_Mg_Susc
- p_Van_der_Waals_Radius_pm.

A better approach, rather than removing these columns, was to impute the missing values. However, expert knowledge in the properties of the elements was needed to check if the imputed values can make a statistically significant impact. Note that the "p" and "m" in front of the names of the features above corresponds to promoter and main metal, respectively.

As a last step on the pre-processing of the data is the standardization applied to the input feature with a personalized function. The numeric features were scaled using the transformation $\frac{x-\tilde{x}}{\sqrt{var(\boldsymbol{x})}}$, where $\tilde{\boldsymbol{x}}$ is the mean of the feature values and $var(\boldsymbol{x})$ the variance. Also, the response variables were scaled to be between 0 and 1, by dividing them by 100.

The final data set for dioxane solvent consists of 101 input features and 144 observations. For diethyl we have 101 input features and 141 observations.

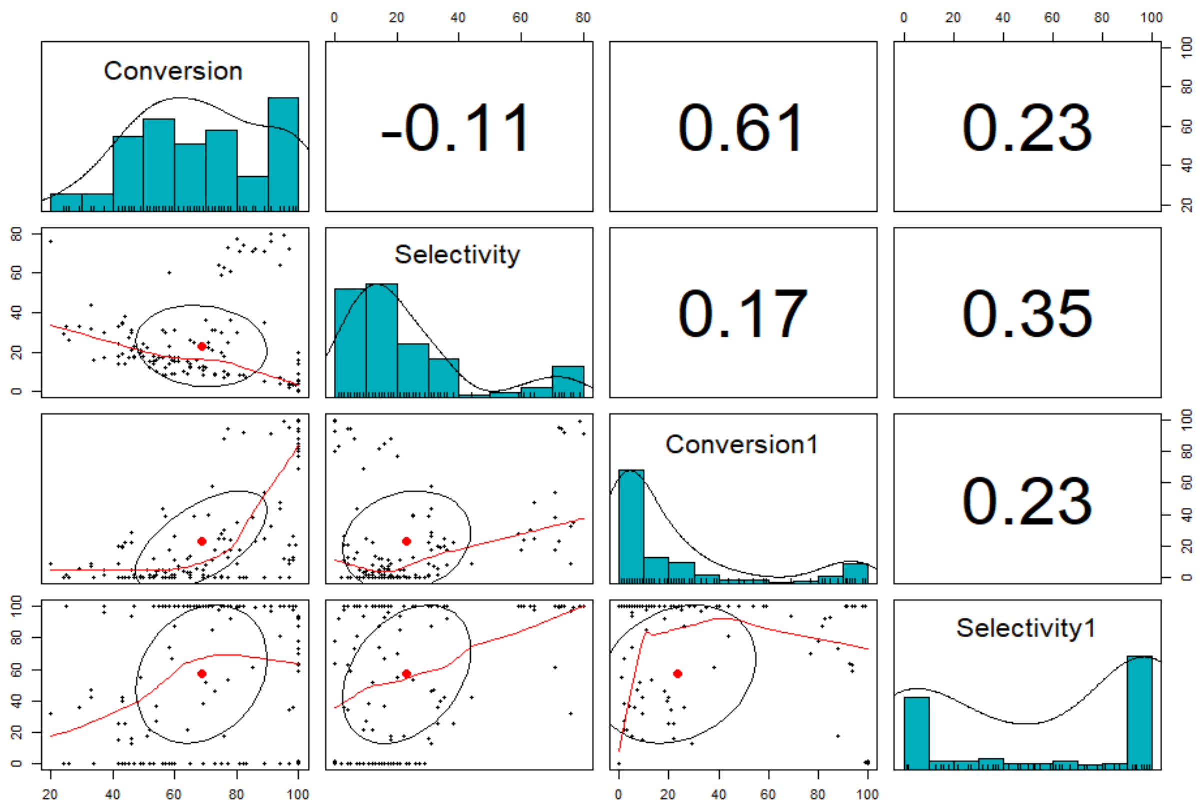


Figure 2: The figure shows the histogram of each response variable in the main diagonal, with the empirical density superimposed to the plot. In the upper diagonal, we the correlation between each pair of the responses and in the lower diagonal, the scatter plot.

2.3 Evaluation Metrics

Before starting to implement the models, it is needed to choose the evaluation method. The method used was the following:

The data was split into training and test set. The idea was to use the training set to select and train the model and then do the final evaluation on the test set. Thus, the test set can be considered as completely unknown (or out-of-sample). The data was split to be 80% for training and 20% for testing.

First, it was used use 5-fold cross-validation on the training set to calculate the expected prediction error of the model. Having chosen the best model using the expected prediction error, the test set was used to calculate the prediction error on new data. It was expected for these two values to be similar, otherwise it would have been an indication of overfitting. Overfitting means when the model performs good on the training set but does not do so on the unseen data (training set). So, the model has learned the data set very good and when is tested on the test set it does not predict good.

Note that the "RMSE Training" value provided on the tables for each model below corresponds to the cross-validation error, except for the gamboostLSS models. In this case it corresponds to the training set error.

The metric that was used to compare the models is the RMSE given by:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y}_i)^2},$$

where y_i and $\widehat{y}_i$ are the actual and predicted values respectively.

2.4 Modelling framework

The three modelling ideas that are implemented were based on component-wise gradient boosting; Generalized Boosted Regression Modelling (GBM), eXtreme Gradient Boosting (XGBoost) and Boosted Generalised Additive Models for location, scale, and shape (gamboostLSS). Gradient boosting is a machine learning method for optimizing prediction accuracy and for obtaining statistical model estimates via gradient descent techniques. A key feature of this method is that it carries out variable selection during the fitting process.

Given the nature of the problem and the dataset available, the base-learners used to regression trees were constrained. All the models were implemented in R using the following packages and their dependencies:

gamboostLSS1, gbm2, xgbootst3

[1] https://cran.r-project.org/web/packages/gamboostLSS/index.html
[2] https://cran.r-project.org/web/packages/gbm/index.html
[3] https://cran.r-project.org/web/packages/xgboost/index.html

CHAPTER 3

RESULTS

3.1 Generalized Boosted Regression Modelling (GBM)

Using R, generalized boosted regression models for predicting the Conversion and Selectivity for each solvent were fitted. The basic tuning parameters of the model were the number of iterations (M) and the size of the constituent trees (J). The model also had several hyperparameters that also needed tuning. In general, gradient boosting is a greedy algorithm and regularization techniques are needed to avoid overfitting. The following parameters corresponds to the regularization techniques used:

Early stopping – to select the optimal number of trees to be added.

Subsampling - the fraction of the training set observations randomly selected to propose the next tree in the expansion.

Shrinkage parameter (η) – controls the learning rate of the boosting procedure.

A detailed description of these parameters can be found in [5].

For choosing the optimal values for each of these parameters, the following approach was followed:

Initialized with many iterations (M) and a small value for the shrinkage parameter (η). The idea is by keeping the learning rate small will require more rounds for to converge. For the size of the constituent trees, a maximum depth $J = 2$ was used, which implies that it was allowed up to 2-way interactions. Finally, cross validation was used to find the approximation of the optimal stopping round or optimal number of trees to be added.

After specifying the optimal number of trees, the rest of the hyperparameters were tuned. A hyperparameter search grid was created and searched for the set of parameters that result in the lowest cross-validation error.

Following the approach described above the following results were obtained:

Solvent		RMSE Training	RMSE Test
Diethyl carbonate	Conversion	8.249346	8.209366
	Selectivity	7.344877	6.603622
1,4-dioxane	Conversion	9.228265	6.930663
	Selectivity	33.36084	30.72591

Table 1: Results of the Generalized Boosted regression Modelling (GBM)

To get some insights from the model, the variables that have the most influence need to be understood. For this the relevant influence as described in Hastie et al. 2009 (chapter 10, pages 367 - 370) were used. Figures in the Appendix show the drivers with the highest relative influence for each of the response.

The important features are derived so to continue the model needs to decipher how the dependent variable changes based on those features. The partial dependence plots for the drivers with the highest influence and their interactions can be found in appendices. Note that the plots are provided using the standardized features. However, the relationship will be the same on the original scale.

3.2 eXtreme Gradient Boosting (XGBoost)

XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible, and portable. It implements machine learning algorithms under the Gradient Boosting framework. Like GBM, XGBoost also follows a gradient boosting framework and fits regression trees. However, XGBoost has some advantages over GBM. XGBoost uses regularization penalties in the objective function to control the complexity of the model, which helps to avoid overfitting. Moreover, it is more efficiently implemented for speed and scalability.

Some additional data preparation was needed for XGBoost, since it takes only numeric features. We used One-Hot encoding for factors. XGBoost has similar hyperparameters as GBM, and a similar approach for hyperparameters tuning as used in the GBM. A detailed description of all the parameters can be found in XGBoost's official website[4]. The results are given in the table below.

Solvent		RMSE Training	RMSE Test
Diethyl carbonate	Conversion	7.73798	8.370523
	Selectivity	6.63216	6.267892
1,4-dioxane	Conversion	9.54526	6.310388
	Selectivity	31.87828	30.09495

Table 2: Results of eXtreme Gradient Boosting (XGBoost)

The feature importance for each case and the partial dependence plots are provided in the Appendix.

[4] https://xgboost.readthedocs.io/en/latest/parameter.html

3.3 Boosted GAMLSS models

GAMLSS use a distributional regression approach, where up to four parameters of the conditional distribution of the response variable are modelled using independent variables. GAMLSS can model the response using distributions from a wide range of available distributions that can be found in Stasinopoulos D. M. et. al. 2017b[5].

The first step in using the GAMLSS models, is to choose an appropriate distribution. Based on an exploratory analysis using distributions from the GAMLSS package and the available functionality provided to fit the distributions to the data, the chosen marginal distributions were constrained to be flexible enough to capture the marginal distribution for each response variable. Note that the distribution given from the GAMLSS's built-in function for some of the responses might not be the best. The reason is, to fit a distribution with four parameters, using a flexible regression model for each parameter, it is recommended to have ~1000 observations. Since the training set contains only ~100 observations, a distribution that is flexible and can be fitted with the given data was chosen.

The approach that was used for fitting the GAMLSS models is gradient boosting [3]. The base learns used are stumps (regression trees with a single split). The noncyclic method to fit the model was also used. This means that at every iteration only one base learner is selected to one of the distribution parameters.

The hyperparameters of the model is the early stopping of the algorithm and the shrinkage parameter. For tuning these parameters, cross-validation was used, which was based on the following approach: We choose a small value for the shrinkage parameter and a large number for the boosting iterations. Then the number of iterations was chosen based on which gave the smallest cross-validation error. The distributions chosen for each of the responses are the following:

Conversion: Generalized Beta type 1 (GB1)

Selectivity: Logit Normal distribution (LOGITNO)

Conversion1: Beta Distribution (BE)

Selectivity1: Logit Normal distribution (LOGITNO)

[5] https://www.gamlss.com/wp-content/uploads/2018/01/DistributionsForModellingLocationScaleandShape.pdf

The histogram of the responses with the marginal distribution chosen are given in the appendices.

Following the approach described above the following results are obtained:

Solvent		RMSE Training	RMSE Test
Diethyl carbonate	Conversion	11.8355	12.86876
	Selectivity	12.94835	16.16082
1,4-dioxane	Conversion	9.310062	14.57453
	Selectivity	36.07054	39.57918

Table 3: Results of Boosted GAMLSS models

CHAPTER 4

DISCUSSION

4.1 Summary

XGBoost has the best performance overall. All the models performed poorly in the case of Selectivity of the 1,4-dioxane solvent. Another approach for this response is to apply a transformation on the response variable. The performance of these models can be potentially improved by adding new “catalytic-informed” features, that will be engineered based on the expert knowledge about the problem.

4.2 Future Work

What there is left to do, is to try other models on the database. To search for better results and to do the comparison between the models with the database where the standardization, cross validation, and feature selection is done or not. Like a grid search to see which of the models would yield the best response.

The modelling done in this thesis was solely centered around gradient descent models. There are other methodologies that can be used to better fit the available data.

With the current dataset being small, the next step is to leverage ensemble learning. This technique consists of training multiple machine learning models and combining their outputs together. Those different models are used as a base to create one optimal predictive model.

In addition, an AI framework can be utilized. Neural networks can be adjusted to perform well with small datasets.

Bibliography

[1] P. Uusitalo, "Development of Predictive Models for Catalyst Development," *Faculty of Technology, University of Oulu,* 2021.

[2] D. M. S. Rigby R. A., "Generalized additive models for location, scale and shape (with discussion).," *Applied Statistic 54,* pp. 507-554, 2005.

[3] A. M. M. S. B Hofner, "gamboostLSS: An R Package for Model Buildig and Variable Selection in the GAMLSS Framework," *Journal of Statistical Software 74(1),* pp. 1-31, 2016.

[4] M. H. G. R. Ras, "Selective Hydrogenation of 5-Ethoxymethylfurfural over Alumina-Supported Heterogeneous Catalysts," *Advance Synthesis & Catalysis,* pp. 3175-3185, 2009.

[5] H. &. T. Friedman, "Regularization Paths for Generalized Linear Modelsvia Coordinate Descent," *Journal of Statistical Software, 33(1),* pp. 1-22, 2010.

[6] Friedman, "Greedy function approximation: a gradient boosting machine," *The Annals of Statistics,* vol. 29, no. 5, pp. 1189-1232, 2001.

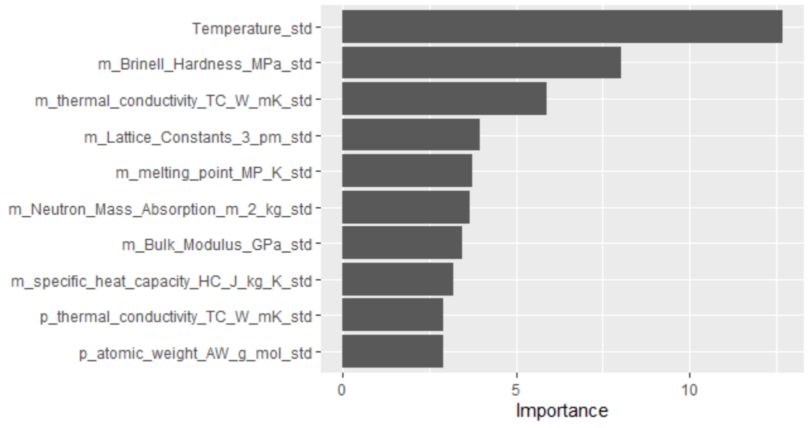
[7] T. T. R. F. Hastie, The Elements of Statisctical Learning, New York, NY, USA: Springer New York Inc., 2001.

Appendix: Features Importance and Partial Dependence Plots

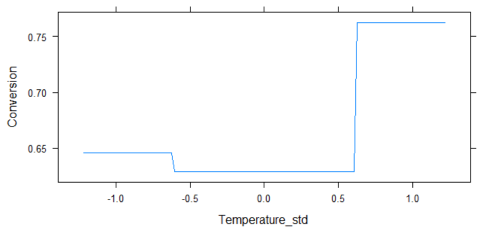
1.1 GBM

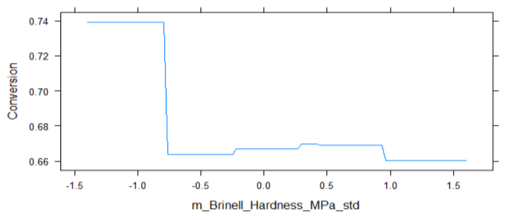
1.1.1 Conversion 1,4-diethyl solvent

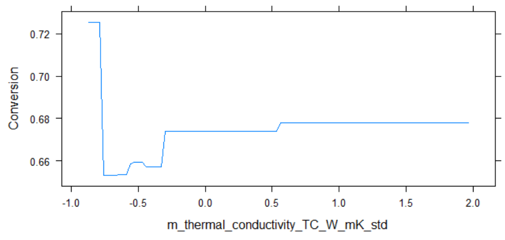
- Feature Importance:

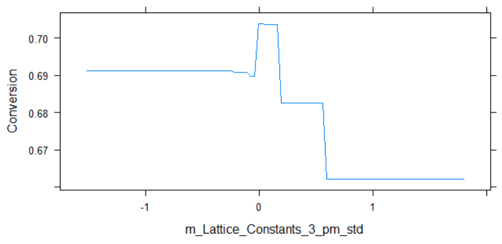


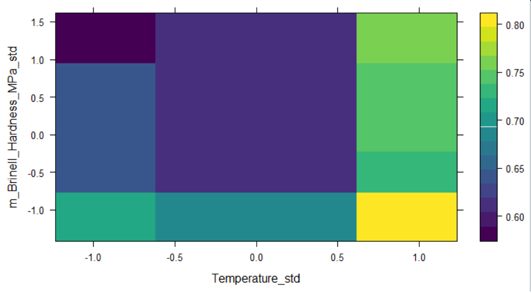
- Partial Dependence Plots:

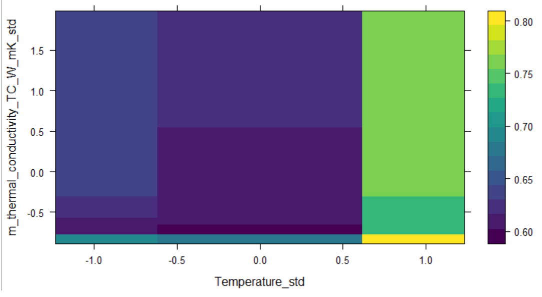






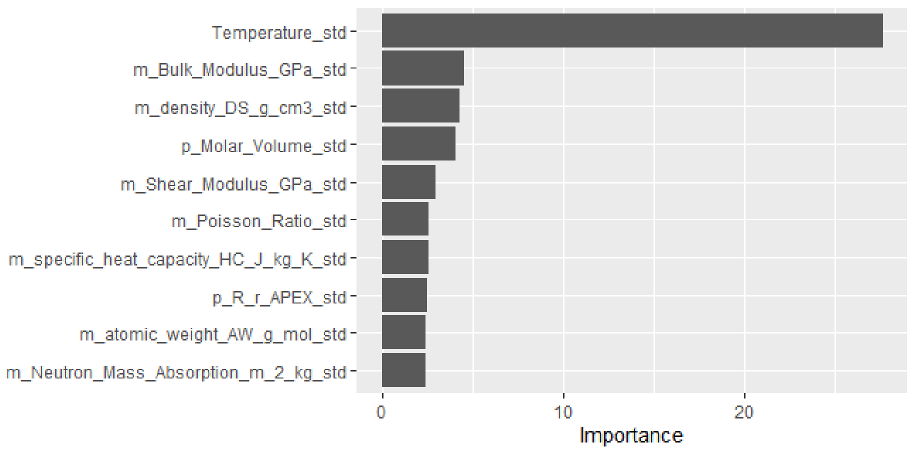




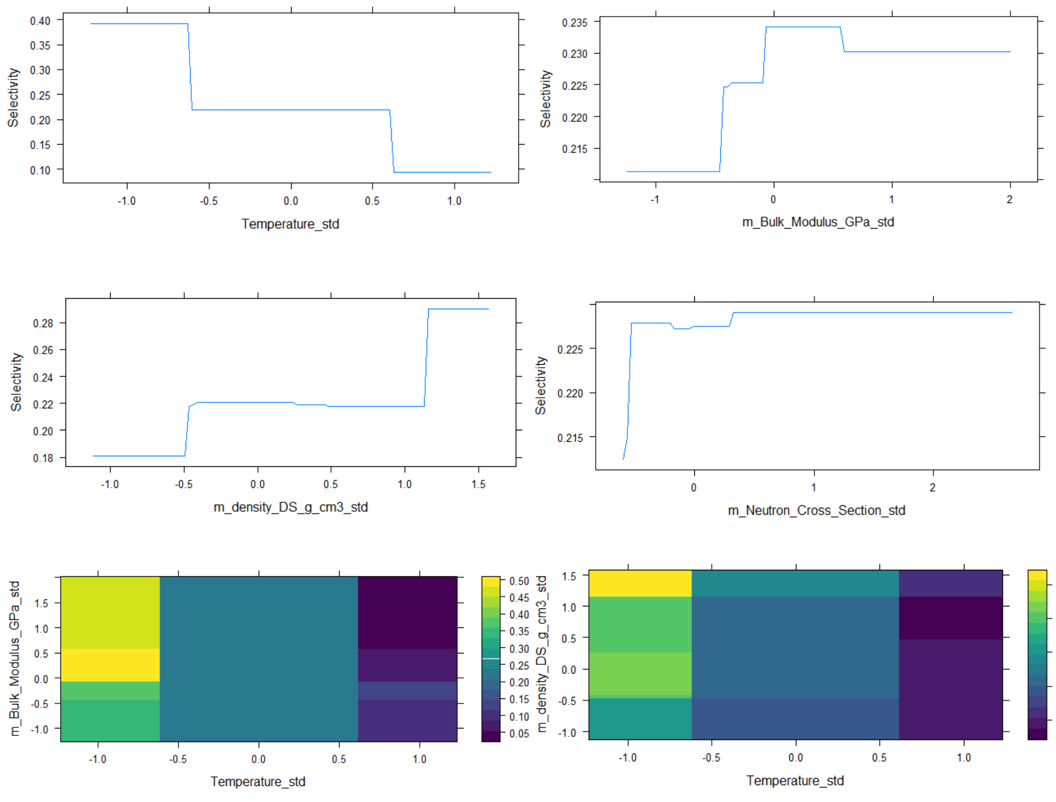


1.1.2 **Selectivity 1,4-diethyl solvent**

- Feature Importance:

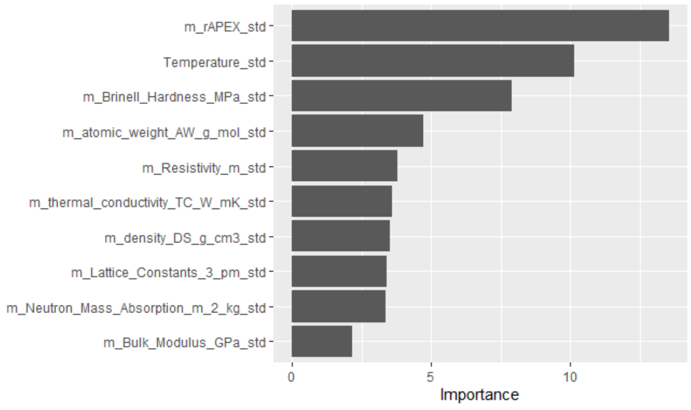


- Partial Dependence Plots:

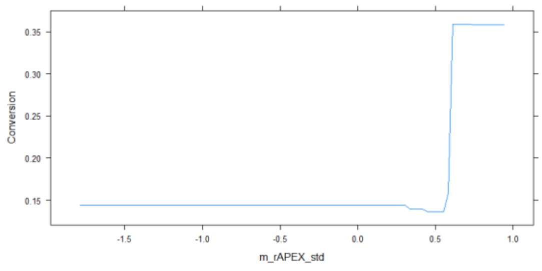


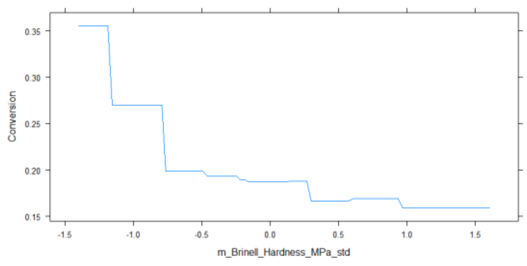
1.1.3 <u>Conversion dioxane solvent</u>

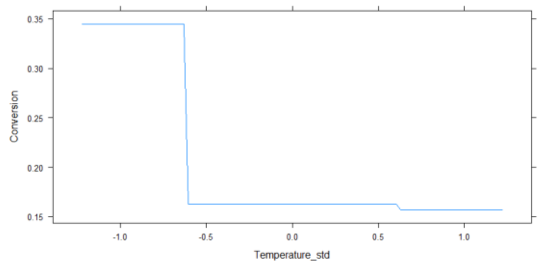
- Feature Importance

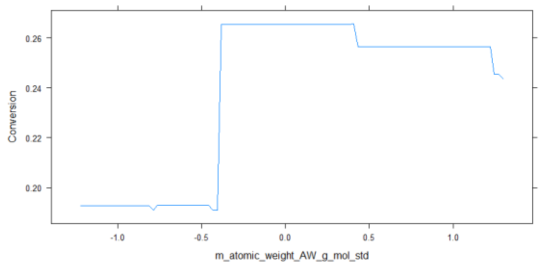


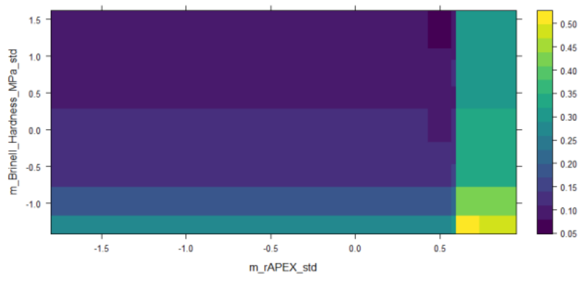
- Partial Dependence Plots

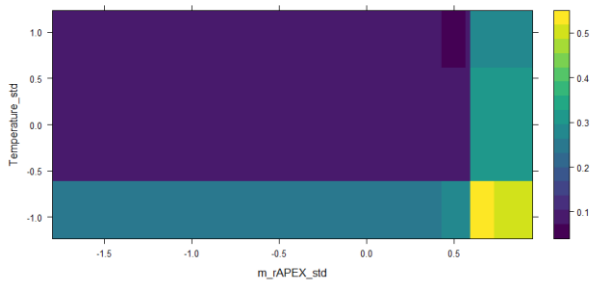






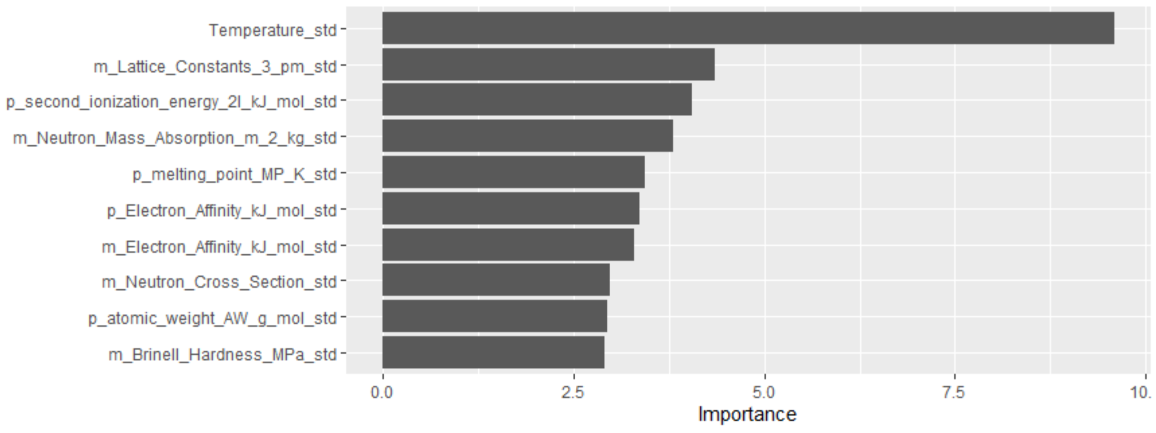




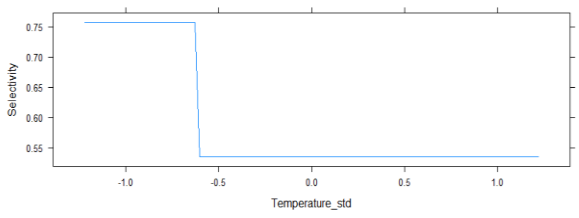


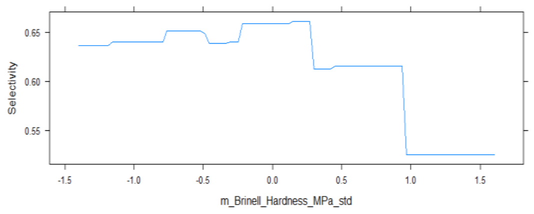
1.1.4 **Selectivity dioxane solvent**

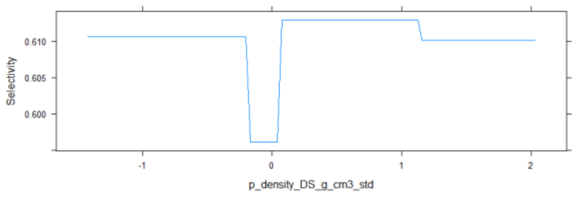
- Feature Importance

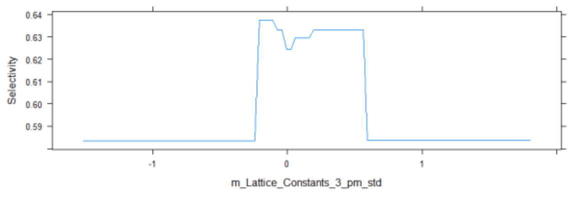


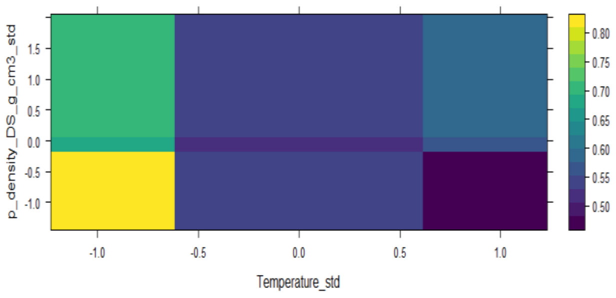
- Partial Dependence Plots



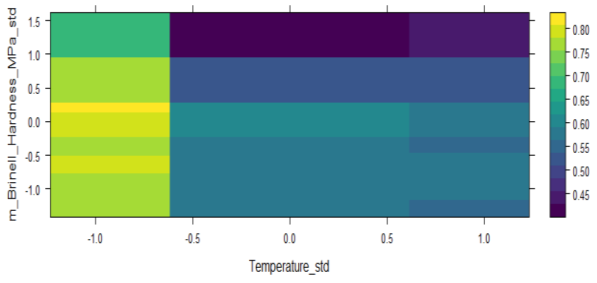








p_density_DS_g_cm3_std
1.5
1.0
0.5
0.0
-0.5
-1.0
-1.0
-0.5
0.0
0.5
1.0
Temperature_std
0.80
0.75
0.70
0.65
0.60
0.55
0.50

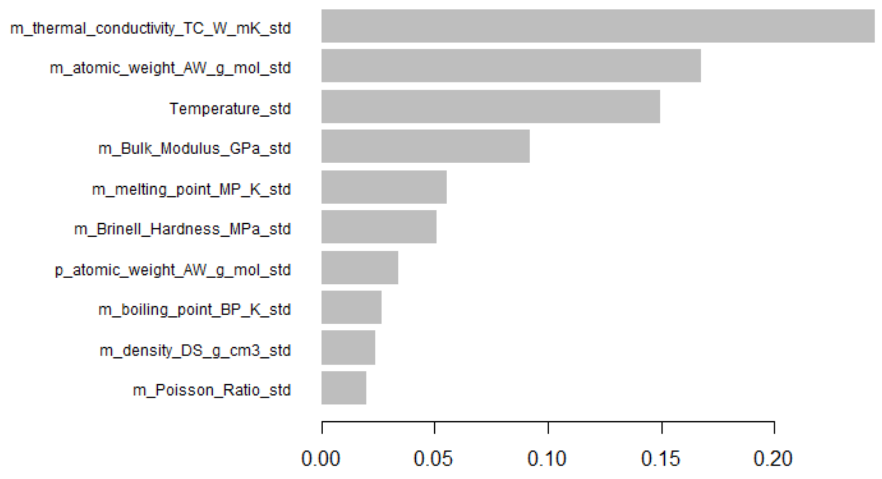


m_Brinell_Hardness_MPa_std
1.5
1.0
0.5
0.0
-0.5
-1.0
-1.0
-0.5
0.0
0.5
1.0
Temperature_std
0.80
0.75
0.70
0.65
0.60
0.55
0.50
0.45

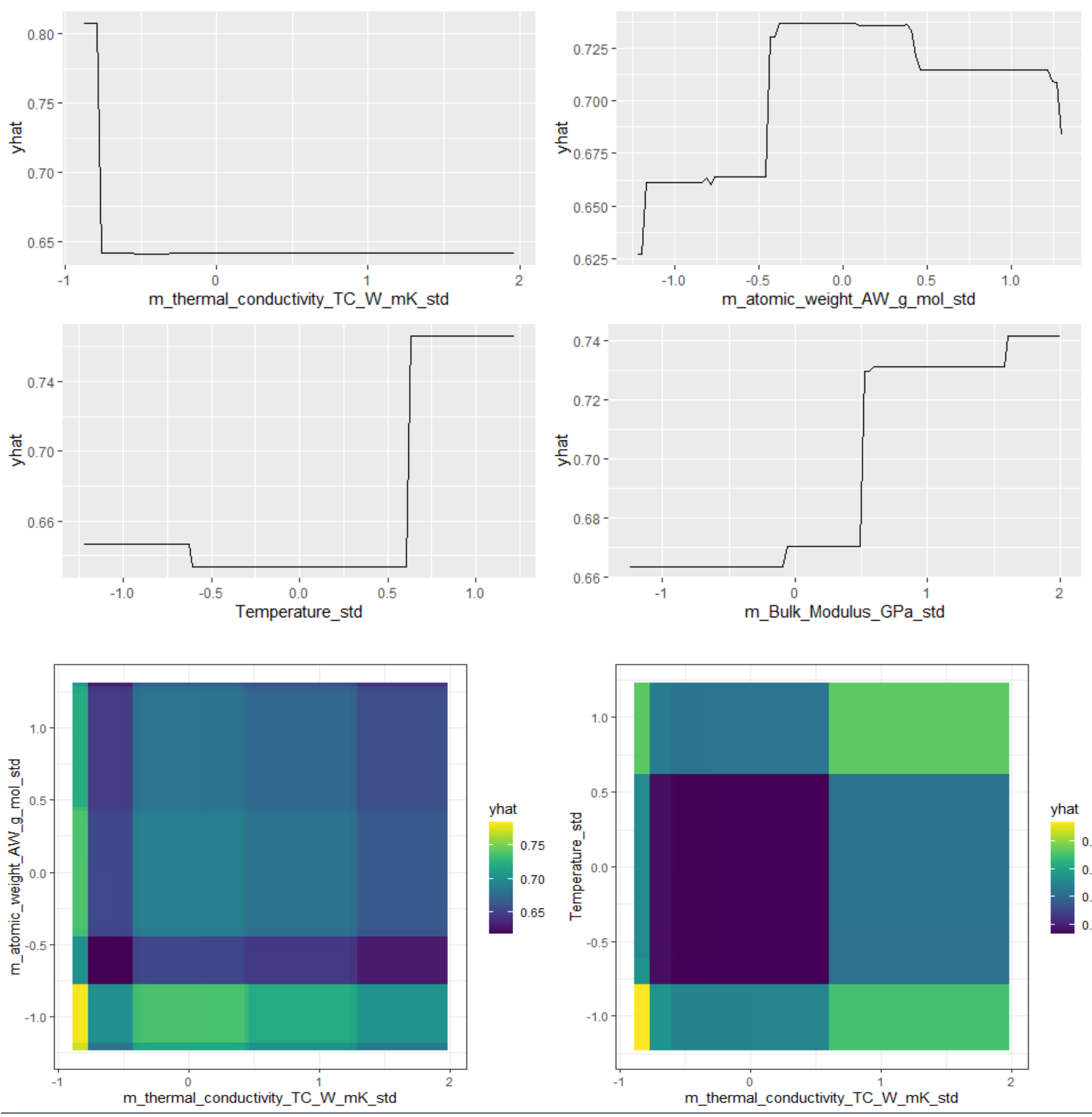
1.2 XGBoost

1.2.1 Conversion 1,4-diethyl solvent

- Feature Importance

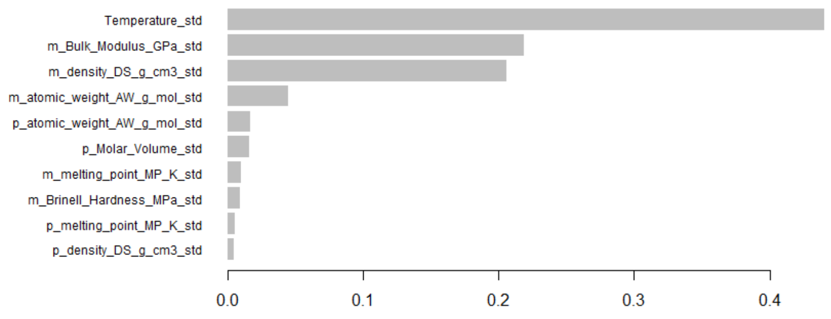


- Partial Dependence plots

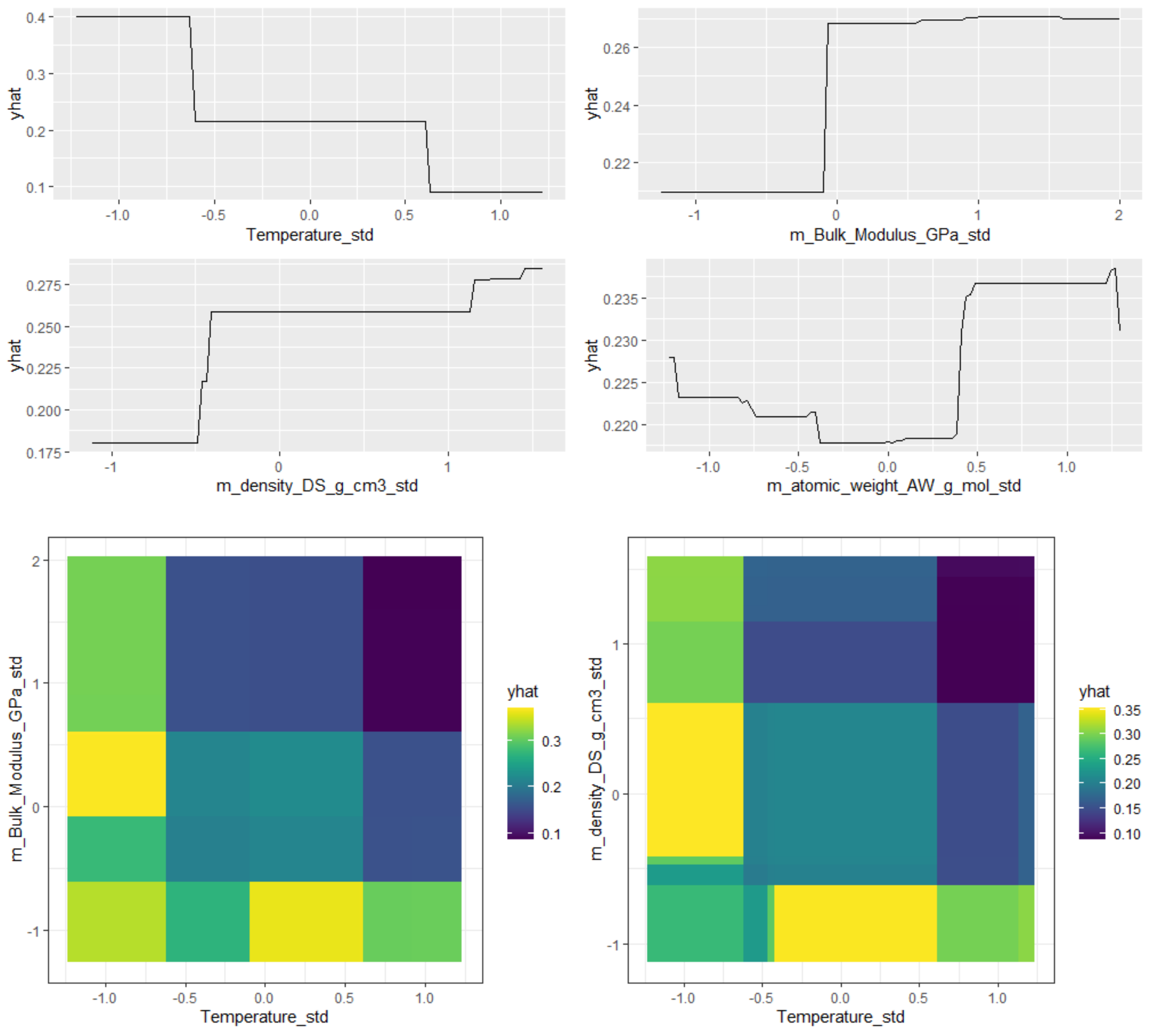


1.2.2 Selectivity 1,4-diethyl solvent

- Feature Importance

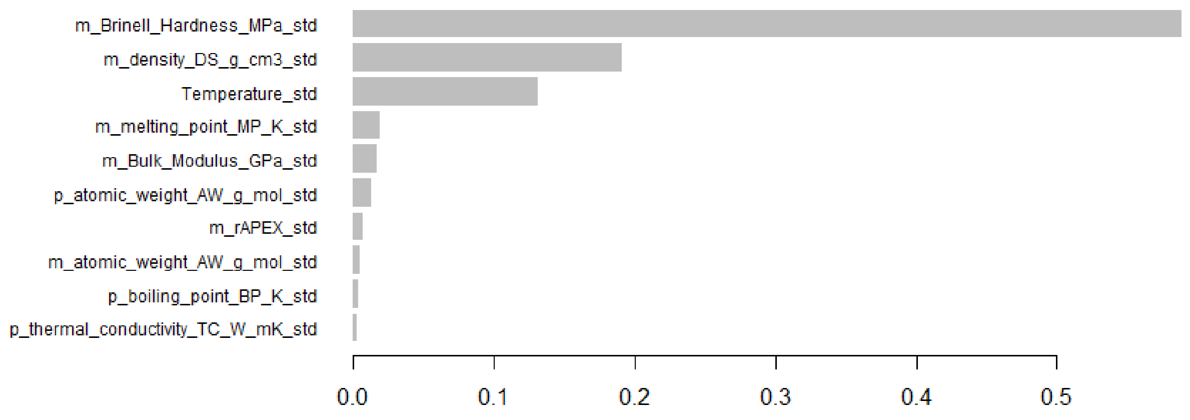


- Partial Dependence plots

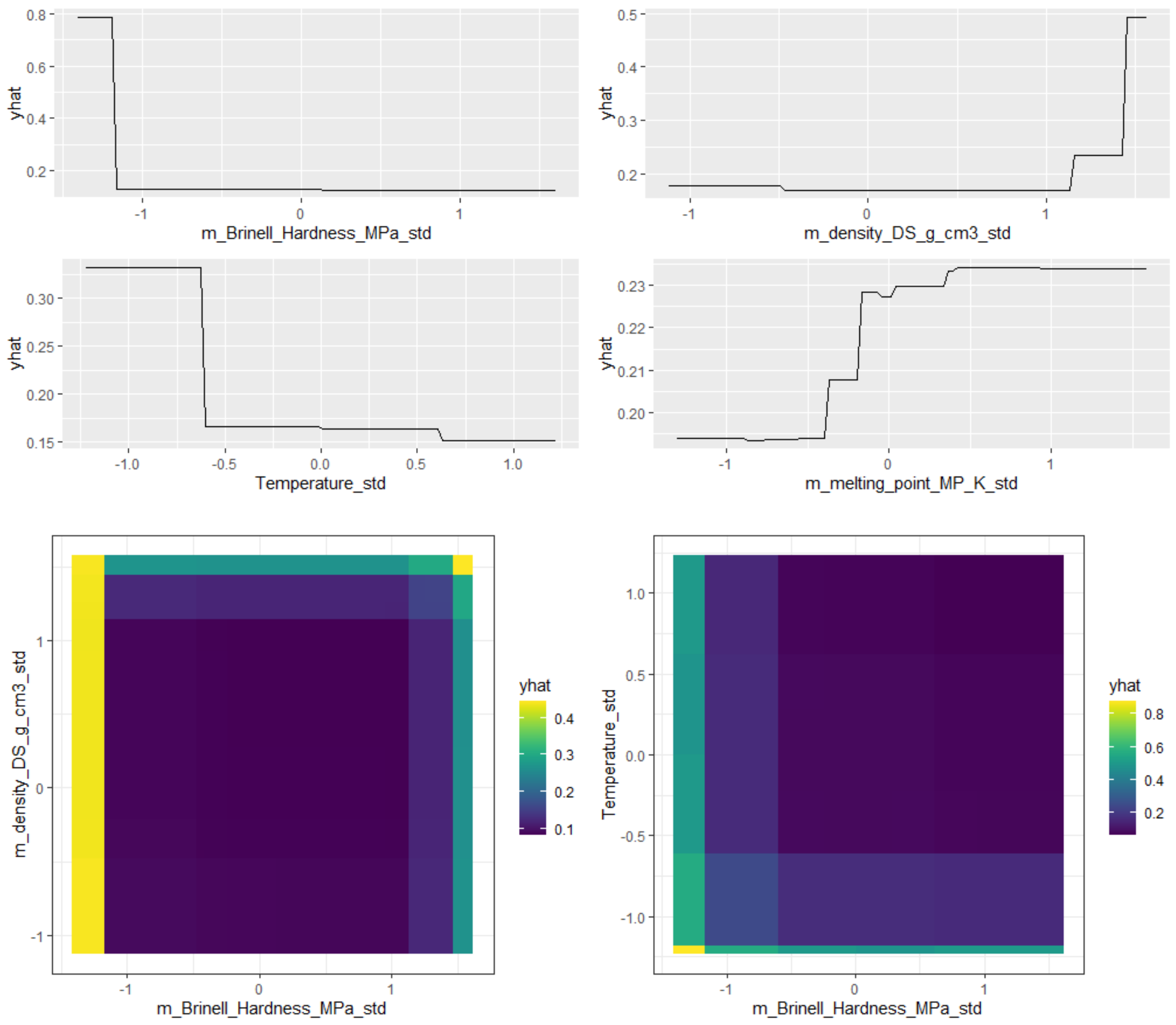


1.2.3 **Conversion dioxane solvent**

- Feature Importance

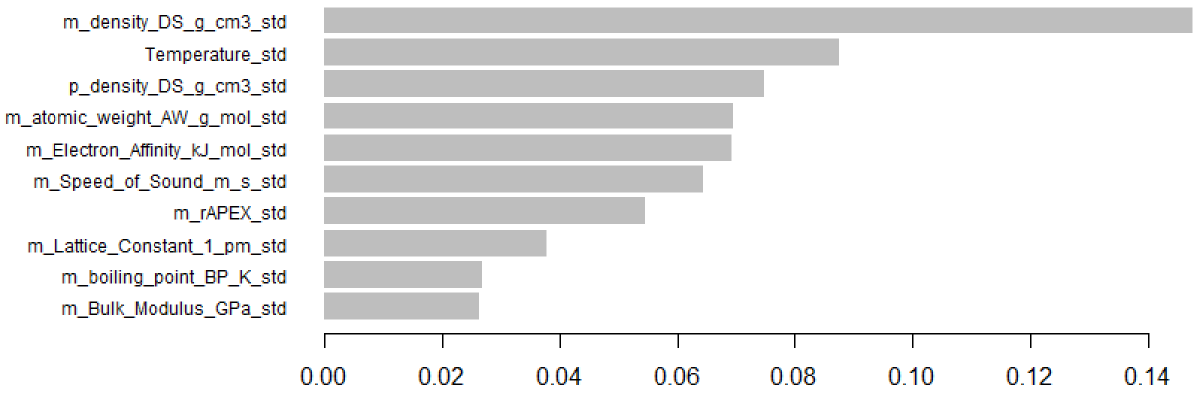


- Partial Dependence plots

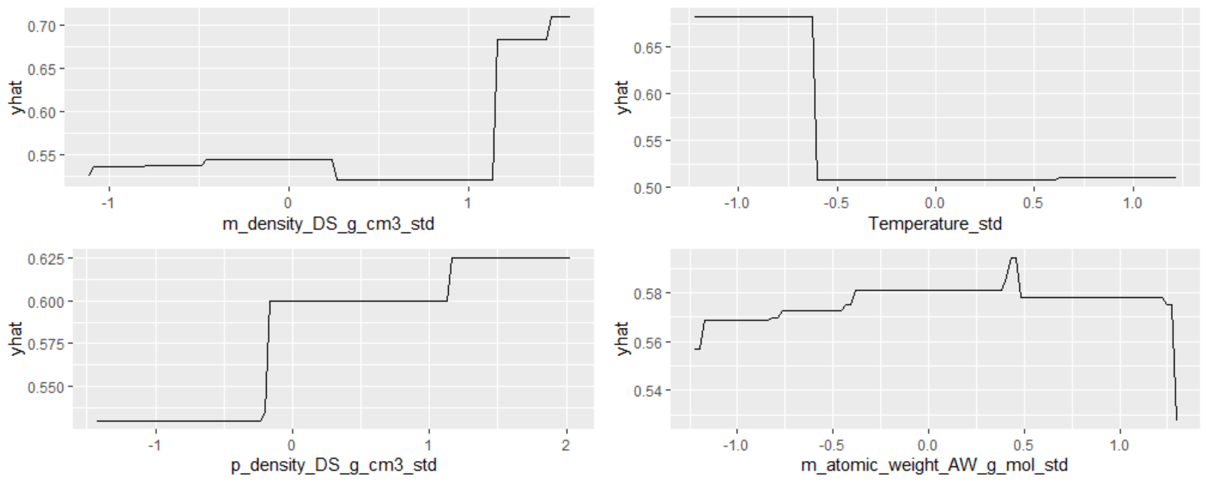


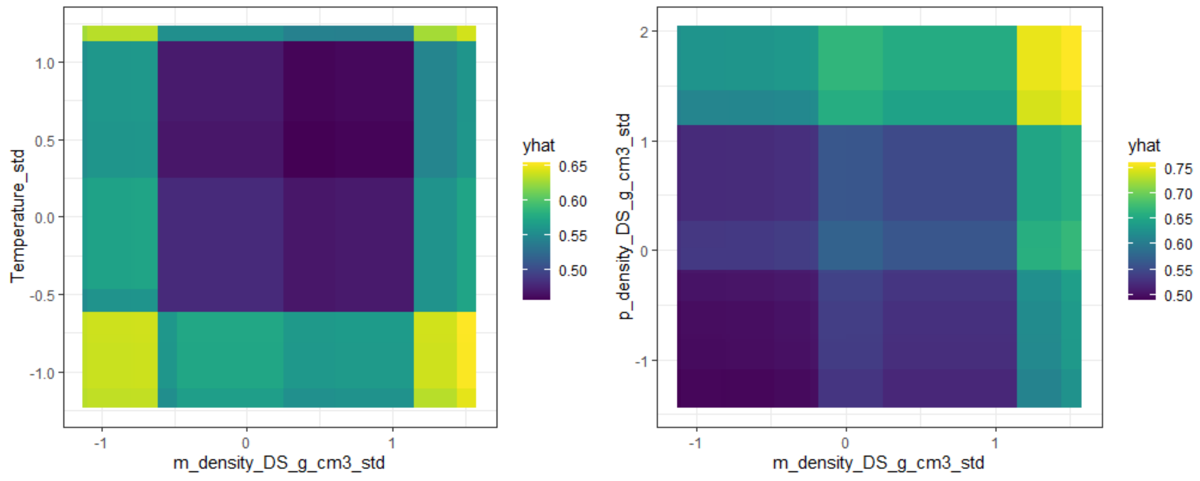
1.2.4 **Selectivity dioxane solvent**

- Feature Importance



- Partial Dependence plots

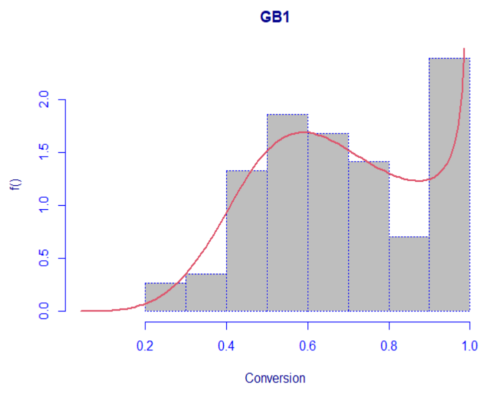


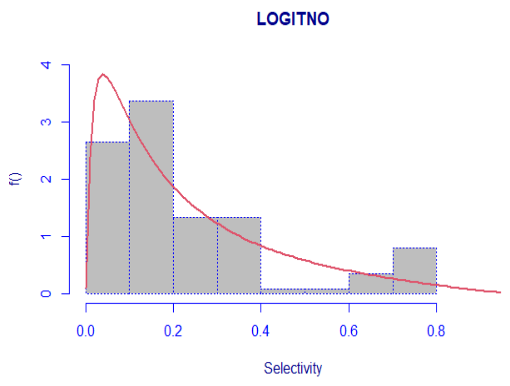


1.3 **gamboostLSS**

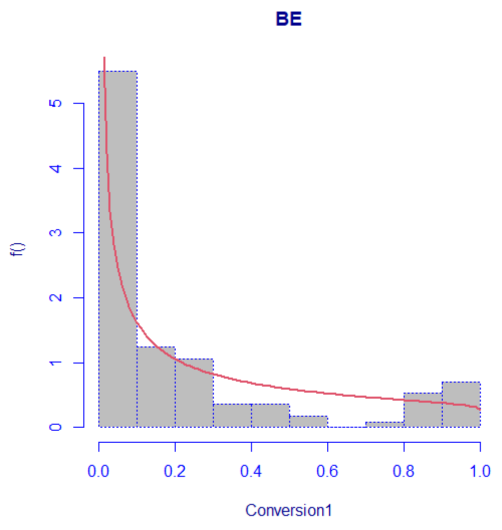
The figure below shows the histogram of each response variable with the chosen marginal distribution superimposed (red curve).

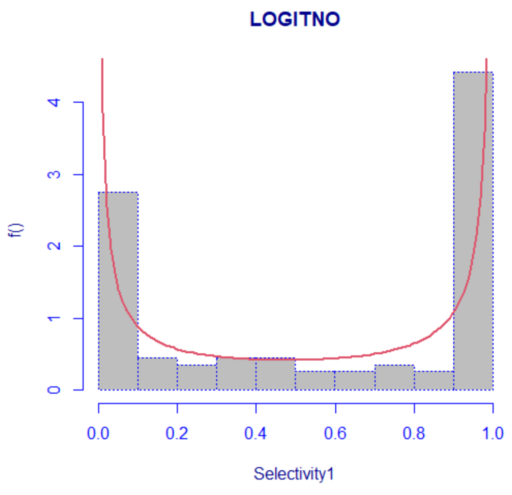
- Diethyl solvent





- 1,4-dioxane solvent





Appendix: Code for GBM in R

```
## Script name: ml_models.R
## Purpose of script: Predictive Models for Catalysis - part of thesis
## Author: Rebeka Kondi
## Date Created: 17/04/2022
## Email: bekakondi@gmail.com
## Notes: To run the script you need to add the data "Databank.xlsx" and
##       script "data_preparation.R" in the working directory.
# --------- GBM ----------------------------------------
#training <- solvent_data[,.SD,.SDcols = c(response_name,num_features,fac_features)]
training <- data.table::copy(solvent_data)
# rename the response to target to avoid code replication
data.table::setnames(x = training, old = response_name, new = "target")

# Split the data to train and validation set
set.seed(1) # set seed for reproducibility
test_ind <- sample(
   x = 1:nrow(training),
   replace = FALSE,
   size = nrow(training) * 0.2
)
test_set  <- training[test_ind,
                  .SD,
                  .SDcols = c("target", num_features,
                           setdiff(fac_features, c("Promoter_fac", "Mainmetal_fac")))]
train_set <- training[!test_ind,
                  .SD,
                  .SDcols = c("target", num_features,
                           setdiff(fac_features, c("Promoter_fac", "Mainmetal_fac")))]

train_set_with_metals <- training[!test_ind]
test_set_with_metals <- training[test_ind]

# Distribution of the response in the training and the validation set
nrow(test_set)
nrow(train_set)
par(mfrow = c(1, 2))
hist(
   train_set$target,
   main = paste0("Training Set"),
   xlab = response_name,
   freq = FALSE
)
hist(
   test_set$target,
   main = paste0("Test Set"),
   xlab = response_name,
   freq = FALSE
)

# Fit a generalized boosted regression models
# We use cv to determine the optimum number of tress for this learning rate

gb_model <- gbm::gbm(
   formula = target ~ . ,
   data = train_set,
   distribution = "gaussian",
   #"laplace"
   n.trees = 10000,
   shrinkage = 0.05,
   cv.folds = 5,
   bag.fraction = 0.5,
   #train.fraction = 0.8, # to use this we need to shuffle the data
   interaction.depth = 2,
   verbose = FALSE
)
best <-
   which.min(gb_model$cv.error) # optimal stopping time based on cv

# RMSE for training set with the optimal number of iterations
sqrt(gb_model$cv.error[best]) # cv error
sqrt(gb_model$train.error[best]) # training error
# Predictions and RMSE of test set
y_hat <- predict(gb_model, newdata = test_set, n.trees = best)
ModelMetrics::rmse(y_hat, test_set$target)

gbm::gbm.perf(gb_model, method = "cv")

# Tune hyperparameters
hyper_grid <- expand.grid(
   shrinkage = 0.05,
   n.trees = best,
   interaction.depth = c(1, 2, 3, 5),
   n.minobsinnode = c(5, 10, 15),
   bag.fraction = c(0.5, 0.65, .8, 1),
   optimal_trees = 0,
   # a place to dump results
   min_RMSE = 0                    # a place to dump results
)
# total number of combinations
nrow(hyper_grid)
# grid search
i <- 1 # for debugging
for (i in 1:nrow(hyper_grid)) {
   set.seed(123) # reproducibility
   gbm.tune <- gbm::gbm(
     formula = target ~ .,
     distribution = "gaussian",
     data = train_set,
     n.trees = hyper_grid$n.trees[i],
     interaction.depth = hyper_grid$interaction.depth[i],
     shrinkage = hyper_grid$shrinkage[i],
     n.minobsinnode = hyper_grid$n.minobsinnode[i],
     bag.fraction = hyper_grid$bag.fraction[i],
         #train.fraction = .75,
         cv.folds = 5,
         n.cores = NULL,
         # will use all cores by default
         verbose = FALSE
     )
     # add min training error and trees to grid
     hyper_grid$n_trees <- hyper_grid$n.trees[i]
     hyper_grid$min_RMSE[i] <-
         sqrt((gbm.tune$cv.error[hyper_grid$n.trees[i]]))
}

hyper_grid <-
     hyper_grid[order(hyper_grid$min_RMSE), ] # ordered results
gbm.fit.final <- gbm::gbm(
     formula = target ~ .,
     distribution = "gaussian",
     data = train_set,
     n.trees = hyper_grid$n.trees[1],
     interaction.depth = hyper_grid$interaction.depth[1],
     shrinkage = hyper_grid$shrinkage[1],
     n.minobsinnode = hyper_grid$n.minobsinnode[1],
     bag.fraction = hyper_grid$bag.fraction[1],
     #train.fraction = .75,
     cv.folds = 5,
     # will use all cores by default
     verbose = FALSE
)

# Results
train_rmse <-  sqrt(gbm.fit.final$cv.error[hyper_grid$n.trees[1]])
y_hat <-
     predict(gbm.fit.final, test_set, n.trees = hyper_grid$n.trees[1])
test_rmse <- ModelMetrics::rmse(y_hat, test_set$target)
res <- data.table::data.table(
     Model = "GBM",
     Solvent = stringr::str_to_title(solvent),
     ResponseName = response_name,
     train_rmse = train_rmse * 100,
     test_rmse = test_rmse * 100
)

# Get/Plot the relative importance and PDP
vip::vip(gb_model)
rel_inf <- gbm::relative.influence(gb_model, n.trees = best)
rel_inf <- rel_inf[order(rel_inf, decreasing = TRUE)]
names(head(rel_inf))
par(mfrow = c(2, 2))
plot(gb_model , i = names(rel_inf)[1], ylab = response_name)
plot(gb_model , i = names(rel_inf)[2], ylab = response_name)
plot(gb_model , i = names(rel_inf)[3], ylab = response_name)
plot(gb_model , i = names(rel_inf)[4], ylab = response_name)
plot(gb_model ,
     i = c(names(rel_inf)[1], names(rel_inf)[2]),
     level.plot = T)
plot(gb_model ,
     i = c(names(rel_inf)[1], names(rel_inf)[3]),
     level.plot = T)


# Plots Predicted Values VS Actual values
# CV in the Training Set
dt2plot <- data.table::copy(train_set_with_metals)
dt2plot$x <- dt2plot$target * 100
dt2plot$y <- gbm.fit.final$cv.fitted * 100
dt2plot[, text := paste0(
    "Main Metal: ",
    Mainmetal_fac,
    " \n",
    "Promoter: ",
    Promoter_fac,
    " \n",
    "Temperature: ",
    Temperature
)]
# plot_ly
fig1 <- dt2plot %>%
    plot_ly() %>% add_trace(
        type = 'scatter',
        mode = 'markers',
        x = ~ x,
        y = ~ y,
        marker = list(
            size = 15,
            color = 'rgba(255, 182, 193, .9)',
            line = list(color = 'rgba(152, 0, 0, .8)',
                        width = 2)
        ),
        text = ~ text,
        hovertemplate = paste(
            "<b>%{text}</b><br><br>",
            "%{yaxis.title.text}: %{y:.2f}%<br>",
            "%{xaxis.title.text}: %{x:.2f}%<br>"
        )
    ) %>% layout(
        title = list(
            text = paste0(
                "GBM Out-of-Sample (CV) Predictions \n",
                "Solvent: ",
                stringr::str_to_title(solvent),
                " & Repsonse: ",
                response_name
            ),
            font = list(size = 15)),
        xaxis = list(title = "Actual Values", titlefont = list(size = 15)),
        yaxis = list(title = "Predicted Values", titlefont = list(size = 15)),
        showlegend = FALSE
    ) %>%
    add_trace(
        x = c(0, 100),
        y = c(0, 100) ,
        type = "scatter",
        mode = "lines",
        name = 'abline',
        line = list(color = "grey", dash = 'dash')
    )

dt2plot <- data.table::copy(test_set_with_metals)
dt2plot$x <- dt2plot$target * 100
dt2plot$y <-
    predict(gbm.fit.final, test_set, n.trees = hyper_grid$n.trees[1]) * 100
dt2plot[, text := paste0(
    "Main Metal: ",
    Mainmetal_fac,
    " \n",
    "Promoter: ",
    Promoter_fac,
    " \n",
    "Temperature: ",
    Temperature
)]
# plot_ly
fig2 <- dt2plot %>%
    plot_ly() %>% add_trace(
        type = 'scatter',
        mode = 'markers',
        x = ~ x,
        y = ~ y,
        marker = list(
            size = 15,
            color = 'rgba(255, 182, 193, .9)',
            line = list(color = 'rgba(152, 0, 0, .8)',
                        width = 2)
        ),
        text = ~ text,
        hovertemplate = paste(
            "<b>%{text}</b><br><br>",
            "%{yaxis.title.text}: %{y:.2f}%<br>",
            "%{xaxis.title.text}: %{x:.2f}%<br>"
        )
    ) %>% layout(
        title = list(
            text = paste0(
                "GBM Out-of-Sample (Test Set) Predictions \n",
                "Solvent: ",
                stringr::str_to_title(solvent),
                " & Repsonse: ",
                response_name
            ),
            font = list(size = 15)
        ),
        xaxis = list(title = "Actual Values", titlefont = list(size = 15)),
        yaxis = list(title = "Predicted Values", titlefont = list(size = 15)),
        showlegend = FALSE
    ) %>%
    add_trace(
        x = c(0, 100),
        y = c(0, 100) ,
        type = "scatter",
        mode = "lines",
        name = 'abline',
        line = list(color = "grey", dash = 'dash')
    )

fig_combine <- subplot(fig1,
                       fig2,
                       titleX = T,
                       titleY = T,
                       shareY = TRUE) %>%
    layout(
        title = paste0(
            "<b>GBM</b> Out-of-Sample Predictions ",
            "(Solvent: <b>",
            stringr::str_to_title(solvent),
            "</b> & Repsonse: <b>",
            response_name,
            "</b>)",
            "<br>",
            "<sup>",
            "Left: Cross-Validation predictions on the training set, Right: Predictions on the test Set",
            "</sup>"
        ),
        margin = list(t = 100, b = 100)
    )

save(res,
     fig_combine,
     file =  paste0(getwd(), '/Plots/gbm_', solvent, '_', response_name, '.Rdata'))
```