Universiteit Leiden

Master Computer Science

Automated Regression Pipeline for Time-Series problems with Real-World applications.

Name:	Pedro D. Santamaria Hernandez
Student ID:	s2379325
Date:	31/08/2021
Specialisation:	Advanced Data Analytics
1st supervisor:	Prof.Dr.T.H.W.Baeck
2nd supervisor:	Dr.Anna V. Kononova

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Content

List of Figures

List of Tables

Abstract

In this Master's Thesis research project, an automated machine learning pipeline for time series regression problems is proposed and applied to a number of real-world data sets in areas such as earthquake strength regression (based on seismic data), age regression (based on neurological data), and mental condition regression (based on neurological data). The automated pipeline consists of a set of modules comprising data pre-processing and feature extraction (using $tsfresh$ python package), feature selection (using $RFECV$) and modeling regression algorithm (using $Random\ Forest\ Regressor$) with hyperparameter optimization (using $Optuna$ software framework). The problem addressed in the study consists in developing an effective automated time-series regression pipeline that could serve as an accurate model to predict an output variable, with a specific focus on its potential application in the medical domain, for predicting mental conditions from EEG measurements. We validate the effectiveness of the resulting tuned model with two different datasets (seismic signals and EEG data) and achieve a coefficient of determination R^2 0.56 with seismic signals in comparison with the previous results obtained by other researchers. Moreover, using the EEG dataset from Leiden University Medical Center, the approach results in a model with an R^2 of 0.432. Not being able to provide a high enough precision for these predictions despite the experience gained from previous experiments of the pipeline, probably due to the small size of the dataset. Also, results indicate the values in the extremes of the range are much less correlated than the rest. We conclude that the large datasets used in some of the experiments include sufficient variable resolution to predict with better performance and less noise, avoiding the heteroscedasticity and multicollinearity in contrast to small datasets in time-series models, and that the automated machine learning approach for time series regression can be a first step towards mental condition prediction based on EEG data.

Keywords: *Random Forest Regression,Machine learning, Supervised Learning*

Acknowledgements

I would like to thank my supervisors, Thomas Bäck and Anna V. Kononova whose dedication and hard work side by side along with me, their love for detail, and the goal of doing a well-done Thesis, made this work possible. Special mention to Milan Köch and Marios Kefalas whose patiently transmitted invaluable knowledge from the regression field, and dedicated time in helping me to understand important background information to reach the most practical results possible. Special mention to Matthijs van Leeuwen and Alexandra Blank, who contributed with valuable insights along the way, and guided me academically during this project. I would also like to thank LIACS for providing a creative and collaborative environment which makes it an excellent place to study. Big thanks to the Leiden University (LU) for promoting this great MSc in Computer Science. Finally my deepest gratitude goes to my family and my girlfriend for their constant unconditional support, which always encourages me to keep working hard and never give up.

1 Introduction

Artificial intelligence (AI) will be increasingly used in society, business, industrial applications, and the medical domain due to the constant increase of data density and complexity.[13] AI algorithms and machine learning are already being used by healthcare providers, medical research institutions, as well as life sciences firms. Many of AI uses in medicine field involve diagnostic and therapy recommendations as well as patient engagement and adherence.[13] The vast amount of AI category applications in the field of medicine involve diagnosis and treatment recommendations; patient engagement and adherence; and administrative activities.[13] However, involving the issue of automation and work in healthcare, there is an ongoing discussion about ethical issues related to the application of AI, as it is expected that it can replace humans in a wide range of activities in the medical domain within a very short period.[13]
As a statistical approach, machine learning (ML) to fit models according to the data and to learn by training them with data. Because of this, ML has gotten perhaps the most normally utilized type of AI in the medical domain. In the field of healthcare, image analysis is the most widely recognized application for ML. Although this study is about precision medicine, for the optimal fit, a patient's characteristics and the treatment setting are taken into account to determine whether treatment procedures are likely to be successful. It is termed supervised learning that most machine learning and precision medicine applications require a training dataset that includes a known outcome variable (e.g. the beginning of illness).[13]
The continuous interest in resolving the medical problems that combines time series prediction utilizing AI (e.g. EEG) has posed a major challenge for medical researchers, data analysts, and biologists who are keen to develop the best prediction methods. Their main focus is put on how the efficient algorithms and technologies may be introduced for managing and analyzing large time-series datasets with multiple variables to characterize a patient's health. The following research question will be studied in this Thesis: Can we develop efficient time series regression approach by using feature-based techniques for applications in the medical domain?
To make predictions, we employ regression equations. After fitting a model, regression equations are a significant component of the statistical output. According to the coefficients in the equation, each independent variable and the dependent variable have a specific relationship.
Thus, Supervised Learning algorithms include regression and classification techniques. Both methods are used in Machine Learning to make predictions and deal with labeled datasets. It is the form of application of each of the different methods that characterized them within the automatic learning of the algorithms.
Thus, the main difference between regression and classification algorithm is that regression is used to predict continuous values such as price or age, while classification algorithm is used to predict/sort discrete values, such as true or false, disease or not.
Regarding time-series datasets, ML can be applied to solve the problems where a numeric or categorical value must be predicted, and the rows of data are ordered by time. There are two different types of time series datasets, univariate and multivariate. When the time series datasets only have one variable is called univariate datasets, for instance, sales or demography datasets. On the other hand, the multivariate datasets are composed of various variables. For instance, meteorology or monitoring datasets.
As result, it became necessary to implement machine learning (ML) methods, as a result of a huge variety of required knowledge needed for developing medical research projects. Due to this variety of knowledge combined with the lack of biodata scientists experts in the medical industry often calls for simple but good performing modeling techniques which can also be ap-

plied by clinicians with some knowledge of data science. In other words, the biomedical sector requires simple, comprehensible, and reliable ML techniques which solve real-world problems with good performances in an automated manner. Such automated methods belong to the group of so-called Automated Machine Learning (AutoML), in particular, an automated ML pipeline consists of feature extraction, feature selection, modeling algorithm, and optimization. Hence, Random Forest Regressor (RFRs) is the chosen model algorithm because it offered great robustness as well as benefits in cost and time[41]. Thus, RFRs providing variable importance measures that properly rank the selected features according to their predictive power.
Most research efforts, [29] have traditionally been done to extract suitable features. Unlike deep learning procedures and GPU calculations [45], where relevant features can be discovered in the training process. It is important to understand the discovered features because we can learn something new about the area in which our model works or be sure that the model extracts features without high computation cost [43].
This work discusses and applies the leading edge algorithms of variable selection methods [21] such as recursive feature elimination cross-validated (RFECV) or recurrent relative variable important measure (r2VIM). The thesis' main contribution is an automated regression pipeline for time series which includes an efficient feature extraction and feature selection method to be used in the prediction of various variables related to the medical domain and real-world domain applying regression with Hyperparameter optimization. To synthesize, the relevance of values for each feature is calculated on its observed minimal value of score importance from several runs of RF; to understand the accuracy of the prediction it is important to evaluate the predictions. Therefore, before several runs of RFR, only efficient features selected by r2VIM or RFECV and previously extracted by tsfresh are selected as important for modeling the prediction. Finally, the predictions' objective with real EEG labeled data sets, is to develop a predictive model with the highest accuracy using random forest regression.

1.1 State of Research

The following research will be the focus of this thesis: (a) building an automated regression pipeline for time-series in the medical domain, (b) an assisted feature selection optimization method, (c) comparing the performance of different feature selection algorithms in three different data-sets, and (d) presenting the results for predicting the selected variable and an efficient selection feature method.

1.2 Outline

The thesis structure is organized as follows: Section 2 will introduce some related work on time series pipelines, regression techniques, and the evolution of variable selection methods for random forests. This section will also introduce what has taken place in other academic papers by acknowledging the relationship between the previous work and current topics. Also, it will give some definitions, explanations, and examples of the automated regression pipelines used in this thesis. Section 3, will present the pipeline with the different modules of feature extraction, feature selection, and Random forest regression (RFR) alongside the datasets used. In Section 4, the discussion takes place about the problem statements and some hypotheses within this research. Also, includes four kinds of different experiments which can provide experiment results. To conclude, future work will be discussed in Section 7.

2 State of the art

In this chapter, we will present some work that has been done on the topic of machine learning regression pipelines (specifically time series pipelines) and predicting algorithms. We will also present some work that has been done concerning the Leiden University Medical Center (LUMC) dataset which contains real EEG recordings of 125 patients. Thereafter, we will mention the aspects regarding the regression, feature extraction and feature selection.

2.1 General Work

We begin with a short literature review placed to highlight previous research undertaken in the field of predictions using classification and regression. In ML, a machine learning algorithm is applied to automatically construct a model from data. This research focuses on regression in Machine Learning (ML) through Random Forest. Nevertheless, ML-based predictions can be modeled in one of two approaches: Classification, which can predict the possibility of several features on n-number of features calculated from the selected features; Regression, which can predict the remaining features until the variable appears. The independent feature (variable) - used to predict the value of the dependent feature (variable) - in both modeling approaches is categorical. Dependent features for Classification are categorized and discrete, while dependent features for regression are numerical and continuous. ML method object of study in this work is RF and is demonstrated from a regression approach. In many cases, a preprocessing step may be applied to the data, before using an ML algorithm. This preprocessing can help the machine learning algorithm to learn faster or better. When dealing with imperfect data several techniques may be used to deal with the processing of features that involve missing or inadequate data, or data that is in a format incompatible with the machine learning estimator being used. For instance, in this work is a given dataset with the Electroencephalography (EEG) signals of patients, and a set of annotated labels, one can automatically construct a model predicting the possibility of having a disease in the patients based on the annotated labels applying regression techniques. A previous predictive model based on classification, as comparison of the present study, is the Automated Machine Learning for EEG-Based Classification of Parkinson's Disease Patients.[27] The use of Boruta algorithm (feature selection step in machine learning pipelines) is an elegant wrapper method built around the Random Forest model. The use of a Boruta model determines feature importance by comparing the relevance of the real features to that of the random probes, it provides accurate and stable results of feature selection to perform the classification on multi-variate time series input. Previously there are several works in this field such as the work of Hansen et al[18], which focused on the classification of Peptides using random forests and genetic algorithms to conduct feature selection.

Similarly, the Automated machine learning classification pipeline for Parkinson's disease [27], which focuses on Boruta, as the feature selection algorithm used with the LUMC dataset with the EEG recordings of 125 patients. It creates a random forest model based on the real features and so-called shadow features provided by arbitrarily shuffling values for each actual feature. All applicable features are contained in the algorithm (compare with the minimum-optimal set). The Boruta algorithm is a wrapper algorithm over the random-forest classification algorithm, not just a stand-alone algorithm.

In the regression, pipeline performed this step moves forward in the feature selection algorithms (r2vim, RFECV) for the implementation of regression.

2.2 Regression analysis

Regression is a statistical tool used to evaluate the relationship between variables and to predict scores on the variable given and scores on other variables. It is a method for fitting a curve across a set of points (measurements) by applying some criterion to determine how well a curve fits to the points. Based on our chosen regression method and criterion, a regression function produces a curve that best fits the points using our chosen regression method. There are several types of regressions that may be performed depending on what we anticipate the relationship to be: if we expect it to be linear, for instance, we conduct some kind of linear regression, if we expect it to be quadratic, for instance, we do some kind of quadratic regression, and so forth.[44] We employ several additional regression-related terms that the reader may be unfamiliar with, we explore the specific regression method used in this Thesis reference following, and we clarify two generic concepts here using Figure 1. Looking at this diagram, we can observe a line with the equation $y = ax + b$. We can see that b is the junction of the line with the $y - axis$, which is known as the intercept. In the image, we can also observe that as the $x - coordinate$ of the line increases by one, the line rises in the $y - direction$ by a [17]. As a result of traversing a distance of one in the $x - axis$, the slope of the line is defined as the difference in $y - axis$ that the line travels through.

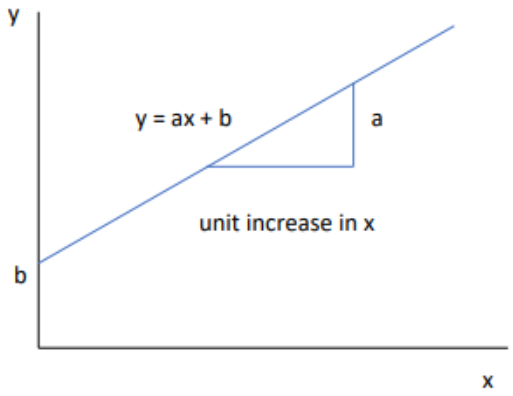

Figure 1: Graph with an equation of the line included, to introduce the terms 'slope' and 'intercept'.

Regression analysis is capable of handling a wide range of situations [17].For instance, utilize regression analysis to perform different tasks:

- Model curvilinear and linear relationships.
- Include categorical and continuous variables.
- Model multiple independent variables.
- Determine if the influence of one independent variable depends on the value of another independent variable by evaluating interaction terms.

In a regression analysis, each independent variable changes are correlated with those of the dependent variable. As a result, every variable in the model is statistically controlled using regression [17]. A residual is another important regression concept to understand how well

the model fits the data, assessing the differences between the observed values and the fitted values.These differences indicate the model's error.There is no perfect model. As a result, the observed values and the fitted values will never be identical.

$$\text{Residual} = \text{Observed value - Fitted value}$$

The equation 1 below shows how to calculate the residuals:

$$e_i = y_i - \overline{y_i} \tag{1}$$

Related to regression analysis, the following terminologies are included[17]:

- Multicollinearity: It occurs when the independent variables are highly correlated with each other.
- Heteroscedasticity:It occurs when a dependent variable's variability does not match across values of an independent variable.
- Outliers: It is an extreme value in the dataset (a very high or a very low).
- Overfitting: It occurs when the algorithm performs well on the training set, but not on the test set. The problem of high variance is another name for it.
- Underfitting: It occurs when the algorithm performs so poorly that even a training set is unable to be properly fitted. The problem of high bias is another name for it.

2.3 Decision Tree - Regression

In this work, the research on decision trees has also been explored in detail. Decision tree builds regression models in the form of a tree structure. Algorithms for the decision tree consist of trees categorizing data with feature values. Each node in a decision tree shows a feature to be classified and the branches show values to be taken into account by such a node. Thus, any internal node tests a feature, the output of the test is on the branch, and the class label is on the leaf node as a result. The largest decision node is referred to as the root node, and it is the parent of all nodes. As the root node, the feature that can filter the data the most efficiently is chosen. This technique is repeated for each subset of the training data before all data has been separated into specific class batches.[38]
Every decision tree has high variance, although when we mix all of the decision trees in parallel, the resultant variance is minimal since each decision tree is fully trained on that particular sample data, and therefore the output doesn't depend on one decision tree but combining multiple decision trees instead of a single one. In the regression problem, aggregation is the part where the final output is the mean of all outputs.[44]
In addressing building decision trees the core algorithm is called ID3 is based on a top-down, greedy search across the space of potential branches, with no backtracking.[40]
In this Thesis we used Random Forest Regressor is an ensemble technique able to execute more than one decision tree using a technique called Bootstrap and Aggregation, broadly called bagging. Combining multiple models into a single, highly reliable model is the goal of ensemble techniques. Boosting, bagging, and stacking are the most prevalent ensemble techniques. As a result of reducing bias and variance, ensemble techniques are suitable for regression and classification problems.[52]

2.4 Feature Engineering

The term feature engineering describes the process of using domain expertise to extract features (properties,attributes,characteristics) from raw data and is a very important aspect of ML. The main goal of Feature engineering is to get the best possible results from a predictive model.To analyze or predict something, it will be done using the features (numeric representation of raw data) that are properties shared by independent units.[54] The dependencies are:

- The raw data selected and prepared.
- The predictive models using.
- The framing of the problem.
- The performance measures.

Predictive models rely on features to influence results. The efforts of feature engineering mainly focus:

- Creating a suitable input dataset, compatible with the ML algorithm requirements.
- Improving the performance of ML models.

The importance of feature engineering is described because the features in the data will have a direct impact on the predictive models that are deployed and the results that can be achieved. Thus, several properties influence the results, therefore it needs great features that describe the inherent structures of the data. [36] Great features mean:

- The flexibility of the good ones, employ less complex models that are faster to run, easier to understand, and easier to manage.
- Simpler models, there is a representation of all accessible data that might be used to better characterize that underlying problem.
- Better results, as a result of choosing the right models and optimized parameters.

Fundamental methods used in the feature engineering process are discussed below [54]:

- Imputation: It is necessary to remove rows or columns or impute values using the column medians when there are missing data.
- Extracting date:In most cases, date columns give significant information regarding the model's target, but because dates might be presented in a variety of forms, standard extraction is essential.
- Scaling:Normalization or standardization is required since most numerical features of the dataset do not have a defined range.
- Handling Outliers: Outliers can be handled in two ways. Using standard deviation and percentiles to determine whether to drop or cap, these will discover them.
- Binning:It is used to make the model more resilient and to prevent overfitting.

- Logarithm transformation:It is a mathematical transformation to handle skewed data,normalize the magnitude differences and reduces the effect of the outliers.
- One-Hot encoding: When categorical data is converted to a numerical format, the values in each column are split across multiple flag columns.
- Feature split:It depends on the characteristics of the column, how the split function is applied.
- Grouping Operations:By categorical column grouping or by numerical column grouping, the aggregation functions of the features must be determined.

Using the above fundamental methods of feature engineering the prediction accuracy is greatly increased, proceeding with others methods of data preparation such as feature selection,train/test splitting, and sampling used below.

3 Methods

This section provides a detailed overview of the used steps to build the Automated Regression Pipeline for Time-Series developing the different methods employed below.

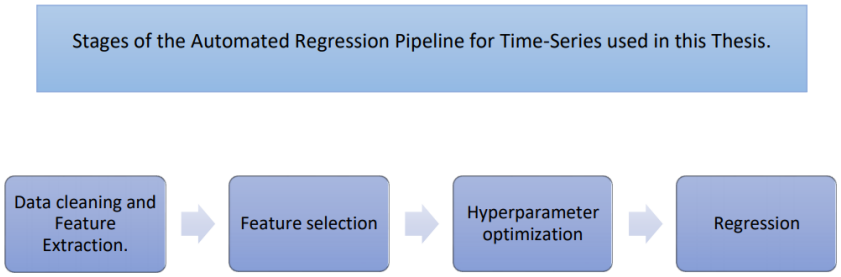

Figure 2: Pipeline Configuration

3.1 Data preparation and feature extraction

Several methods can be used to deal with missing or inadequate data.The machine learning estimator is used to handle inaccurate data or data which is incompatible with. Sometimes, the problem of missing values within features or targets must be tackled when dealing with datasets of any size from the real world.
The magnitude of this problem could range from a large number of missing values (sparse data) to only a few missing values across a number of features. Thus, different ML algorithms and specialized implementations have a variety of awareness to missing data, such as Naïve Bayes, deal smoothly with missing values as it is linear and its features are handled separately, still others can not allow for missing values, especially nonlinear methods, such as random forests[7]. In this case, we presented the method used for dealing with missing data in this Thesis. This consists of either remove some or all of the observations that include missing data (complete-case analysis) or alternately by imputing the missing values by using a mathematical or stochastic process [47]. This approach is possible only with large datasets or very few missing data columns, as it removes observations and thus limits the effectiveness of a supervised algorithm to train successfully [31]. A shortcoming of the approach is that observations with missing data may not be uniformly distributed across all target classes, and instead may be more highly correlated with certain outcomes; thus, removing observations may skew the overall results and thus the predictive strength of the model.
Instead, a second alternative consists of interpolating the feature values based on an algorithm that can range from easy use of the mean of non-missing data to so sophisticated that other machine learning strategies like logarithmic regression are used to calculate the missing value. However, imputation of too many values can lead to weaknesses in the model as well[9]. For simplicity,in this Thesis we imputed missing values, using the mean of other data from the same feature, despite this approach having the potential of inducing bias [20].
To address the feature selection problem with the identification of all strongly and weakly relevant attributes. It is hard to solve for time series regression and classification, for which each label or regression target is connected with several time series and metadata at the same time

[11]. We used $tsfresh$, which is an efficient and scalable feature extraction algorithm for time series based on a Python package. The mentioned algorithm joined feature importance filter with the feature extraction methods.Thus, the features extracted will be used for describing or clustering time series depending on the extracted characteristics with a scalable feature selection based on non-parametric hypothesis tests and the Benjamini-Yekutieli procedure [11]. Furthermore, tsfresh is compatible with the pandas and scikit-learn libraries and can be used to construct models that perform classification/regression tasks on time series [12].
Loosely speaking, the use of tsfresh is imposed because there are a variety of features that are significant depending on the domain and the various datasets used in this Thesis. To create an automatic and generic approach to generate the features for the time series datasets, it is used to extract a large number of features and then select the most important features for the overall regression task.
The tsfresh combines the components from the hypothesis tests with the feature significance testing[12], given by the following three steps:

- Perform a set of all tested null hypotheses
- For each generated feature vector
- Perform the Benjamini-Yekutieli procedure

For instance, when we evaluated the LUMC EEG dataset, for each pair of patients and epoch, tsfresh extraction procedure gives us for each feature function, 1023 features computed for each EEG time series, in total (21 electrodes per patient, resulting in 21 recorded time series) composed by 21 x 1023 = 21483 features.
Hence, the FRESH algorithm extracts all features in step 1. which are then individually investigated by the hypothesis tests in step 2. and finally in step 3. the decision about the extracted features is made.

3.2 Feature selection

The goal of the feature selection phase is to pick the set of features that are most relevant to the regression problem, having said that feature selection is also known as Variable selection or Attribute selection. Essentially, it is the selection mechanism for the most important/relevant Features of a dataset. There is a major role in developing a successful data mining method by choosing suitable features. It is best to recognize when a feature collection is significant if you have a dataset with a large number of features. This kind of data set is also called a high-dimensional dataset. For this high dimensionality, there are some complications such as the training time of the machine learning model can increase dramatically, making the model very complex, leading to overfitting.
Manual feature selection involves a full understanding of the data and the data domain. Oversight of features can lead quickly to results with poor predictive capability because the model does not know the key information in the dataset that could show an important pattern. On the other hand, the use of contrasting features may also lead to a substandard model as the model can cause overfitting and excessive noise, also generally reduce the speed at which the estimator in a supervised learning environment trains and predicts.With this in mind, proper feature selection is crucial to training classifiers or regressors that provide the highest predictive capability [21].
An overview of the algorithms used is explained following [14]:

Recursive feature elimination and cross-validated selection (RFECV), aims to find the best and minimal collection of variables, that contribute to a good cross-validated prediction model. It begins with an RF constructed on all variables.[34] A certain proportion of the least important variables is then deleted and a new RF is generated using the remaining variables. These stages are iteratively applied until a single variable is left as input. At each stage, the performance of the prediction is assessed based on the out-of-bag samples that were not used for model construction. The collection of variables that leads to the RF with the smallest error or an error within a small range of the minimum is picked. The original method as implemented in the R package varSelRF calculates variable importance only once based on the RF with all variables. RFECV is often applied to analyze high-dimensional molecular data sets generated, e.g. in transcriptomics, proteomics and metabolomics experiments [33].
Recurrent relative variable importance, the idea of r2VIM is based on the assumption that genome-scale datasets usually contain many unimportant variables, which can be used to estimate the importance values of null variables[14]. Several RFs are generated based on the same data set and parameter values differing only in the seed of the random number generating process. Each RF is used to calculate importance values, which are divided by the absolute minimal importance value observed in each run resulting in relative values that can be interpreted more easily. In case the minimal observed importance is exactly zero in a specific RF run, the minimal importance value overall runs are used as a denominator. Variables with a minimal relative importance value greater or equal to a specified factor are selected and defined as important variables.[50] To identify genetic variants that are associated with complex diseases, r2VIM has been developed. Compared with standard statistical methods, such as logistic regression, the power to detect causal variants is only slightly decreased. The method can also be applied to identify gene-gene interactions.
Vita algorithm proposed by Janitza et al.[23] is similar to r2VIM, as it uses only the existing data without any permutations to estimate the null distribution of variable importance scores. The observed non-positive variable importance scores are used to construct a distribution that is symmetric around zero. The authors showed that variable importance values of null variables that are calculated using out-of-bag samples are positively skewed and asymmetric distribution is only achieved using a special cross-validation procedure (called the hold-out approach). The overall data set is divided into two equally sized subsets, and two RFs are trained using either one of the sets. Variable importance is then estimated based on the other, independent set. The final importance values, called hold-out importance, are calculated by averaging the two estimated scores per variable. Based on the resulting empirical distribution, p-values can be calculated [23].
After we evaluated the previous feature selection methods [14] in preliminary experiments, we realized in the comparison that the feature cost minimization,less expensive and time-consuming method is recursive feature elimination and cross-validation (RFECV),this selected feature technique that fits our model and deletes the weakest feature(s) until the specified number of features is attained. Features are ordered by the model's coefficients or feature importances attributes, and RFE tries to reduce dependencies and collinearity that might exist in the model through the recurrent elimination of a small number of features in each loop.
RFE must retain a specific amount of features, however, but it is not known in advance how many are legal features. Cross-validation with RFE is used to find the optimum number of features for determining various feature subsets and selecting the best collection of features. In addition to their cross-validated test score and uncertainty, the RFECV visualizer plots the number of features in the model and visualizes the selected number of features [3].

3.3 Stratified cross-validation

In stratified K-fold cross-validation, the output variable is stratified first and the dataset is randomly split into K-folds to ensure each fold comprises about the same proportion of different strata. Despite the fact that several studies [8] observed no significant benefits from using stratified cross-validation in regression situations. Notwithstanding, [28] analyzed model selection and classification problems evaluation and showed that a stratification is usually a good approach to cross-validation folding.
For several repetitive cross-validations, when a model has been selected the stratification's problem becomes redundant, thus stratified cross-validation is wise to be used in the evaluation of the model and when a class imbalance occurs the stratification can lead to unstable performance measures. We want to point out that there is no clear consensus on the use of stratified cross-validation or any other splitting approach which takes into account the values of the output variable. Furthermore,the 10-fold and 5-fold cross-validations are the most widely used stratified K-fold CVs for evaluating ML models [32].Thus,the 5-fold cross-validation was employed in models of building the experiments of this Thesis to offer an un-biased prediction. To prevent the following effects the use of the aforementioned technique is necessary:

- Overfit Model: Overfitting happens when the noise of the data is captured by a statistic model or machine learning algorithm. It happens by default if the model or algorithm fits the data well.

- The overfitting of a model leads to good accuracy for training data, but poor performance in new data sets. Such a model is not of practical use because it cannot predict results for new cases.

- Underfit Model: Underfitting happens when the underlying trend of the data is not captured by either a statistical model or machine learning algorithm. Intuitively, it happens if the model or algorithm does not sufficiently fit the results. Often, underfitting is a consequence of a model that is too simple which means that the missing data is not handled properly, therefore the removal of unnecessary features or features that don't add anything to the predictor variable is not done properly.

3.4 Hyperparameter Optimization (HPO)

Finding the hyperparameter settings that optimize the efficiency of an algorithm concerning a specific performance measure is known as hyperparameter optimization. Many aspects contribute to the difficulty of configuring hyperparameters for machine learning algorithms [53]:

- Hyperparameters can exhibit nonlinear and non-convex behavior. Having a very small k in the k-nearest neighbor algorithm may overfit, reducing performance. On the other hand, having high k may underfit, again reducing performance. Somewhere in-between is where performance is most likely to be best. Even then, in some cases varying k by small amounts has no measurable impact on performance.

- Hyperparameters can be an integer, continuous or non-numeric. Examples of integer hyperparameters are the value of k in k-nearest neighbors and the maximum depth of a decision tree. Continuous hyperparameters include regularization for Support Vector Machines and the learning rate for neural networks. Finally, non-numeric hyperparameters exist in e.g. the k-nearest neighbor algorithm there can be multiple ways to calculate distance, such as the 'Manhattan Distance' and 'Euclidean Distance'. For decision trees, there are multiple ways to measure the quality of a split, like 'Gini impurity' and 'Entropy'.

- Learning algorithms may have conditional hyperparameters. For instance, with the use of the Logistic Regression in scikit-learn, the solver used must be specified, in order to restrict the possible options available for the penalty parameter.

- Function evaluations can be expensive. The evaluation of multiple distinct hyperparameters may be costly, depending on the size of the dataset and the ML algorithm or its setup. Some models might take days or weeks, others only take several minutes to train.

While feature selection and dealing with incomplete or inaccurate data before feeding to an estimator are critical, other factors may also influence the regressor or classifier's final performance [53]. For instance, a hyperparameter for random forest classifiers is the number of decision trees the random forest will generate, and another is the number of features per decision tree in the forest that will be considered when generating a new node and split; hyperparameter optimization is the process of tuning the parameters that define the estimator's functioning, rather than the values learned by the estimator. Unlike values that are learned by the estimator during training, hyperparameters are generally user defined and passed to the estimator upon initialization. While some hyperparameters potentially impact the estimator's scoring performance, others are provided more for the speed with which the classifier may be trained [39].
To determine the highest scoring set of hyperparameters sampled, the automated processing technique for the hyperparameter tuning function is used by running multiple iterations of the estimator with different combinations of the parameter sets and a scoring function backed by cross validation [39]. Some implementations are exhaustive, testing every possible combination of parameters against the model; however, this approach, while likely to find an optimal or near-optimal solution, nonetheless suffer from being very performance demanding situation, in which the classifier or regressor can be trained, particularly for estimators for which many hyperparameters exist. As an alternative, randomized grid search, works instead by randomly sampling from the provided parameter set a predetermined number of times and returning the best scoring parameter set found after cross-validation, as above. Note that hyperparameters

must be tuned at the same time as feature selection occurs, as certain feature combinations may perform differently with certain combinations of parameters.
A more advanced method for hyperparameter optimization is Bayesian Optimization iteratively chooses sample points based on earlier results, balancing between exploration and exploitation [53]. It constructs a surrogate model for the optimization function. It models the likelihood of various optimization functions so that the expected performance based on the hyperparameter configuration can be determined. There are many possible surrogate models, but a Gaussian process is often used [53].
These surrogate models are updated based on evidence of the true objective function. This evidence comes from performing a function evaluation (machine learning experiment). This evaluation allows the construction of a distribution of values the optimization function is likely to have. Then, an acquisition function will determine which hyperparameter settings to try next, based on the outcome about the optimization function. The acquisition function has to balance exploration and exploitation. On the one hand, it should explore areas that are likely to contain good hyperparameter settings. At the same time, it should make sure not to leave areas with possibly bigger improvements unexplored [53]. Hence, Bayesian Optimization is a Sequential Model-Based Optimization (SMBO) because it uses a model of possible objective functions to determine what good hyperparameter values may be, and sequentially updates this model based on new findings. Originally SMBO techniques were not suitable for algorithm selection due to limitations, such as only supporting numerical parameters. Although SMBO was readapted to solve the problem mentioned, and for this purpose two methods were introduced, Random Online Aggressive Racing (ROAR) and Sequential Model-based Algorithm Configuration (SMAC) that are having a very good performance optimizing hyperparameters [53]. Following we described the Hyperparameter Optimization Framework used in this Thesis because its efficient and allowing the optimization program to run faster.

Table 1: Hyperparameter search space for optimizing the random forest regressor.

Parameter	**Range**
Number of trees	{1,2,.....,100}
Max depth of each tree	{1,2,.....,100}
Min number of samples required to split a node	{2, 3, . . . , 20}
Max number of features when splitting a node	{auto, sqrt}
Min number of samples required in the leaf node	{1, 2, . . . , 10}
Use bootstrap training samples?	{True, False}

3.5 *Optuna* (Hyperparameter Optimization Framework)

Optuna [1], is an open-source next-generation optimization platform with the following innovative features:

- User customization is feasible as it is versatile,lightweight,efficient and implemented with optimized pruning and sampling algorithms.
- Using define-by-run programming, allowing the user to dynamically build the hyperparameter search space.

- A flexible and easy-to-setup architecture that can be used for a wide range of tasks, from simple trials to complex distributed system experiments.

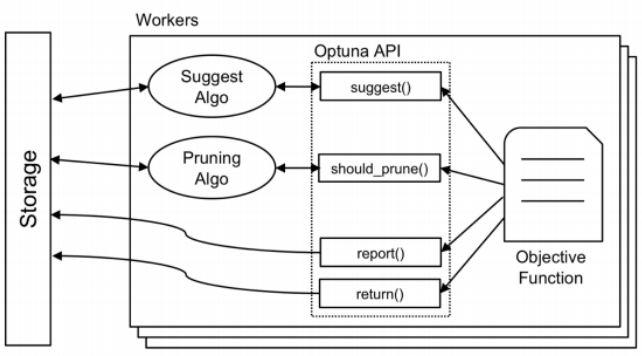

Figure 3: Overview of Optuna's system design.
In each study,each worker performs one instance of an objective function. The Objective function uses Optuna APIs to conduct the trial.When the API is used, the objective function logs in the shared storage and receives information from previous studies from the storage when is required. Each worker executes the objective function separately and shares the progress present study using the storage [1].

Therefore, breaking down to the previous points, the power of define-by-run API consists in how Optuna defines hyperparameter optimization as an objective function using a maximization or minimization process that collects a set of hyperparameters as input and returns its validation score. Optuna alludes to each process of optimization as a study and each assessment of objective function as a trial. Thus, Optuna progressively builds the objective function through the interaction with the trial object [1]. Optuna is also compatible with modular programming and able to work on complex scenarios such as optimizing the hyperparameters of stochastic gradient descent (SGD) and the topology of a multilayer perceptron. The efficiency of the search strategy that selects the set of parameters to investigate is addressed by an efficient sampling and pruning mechanism. Optuna implements relational sampling and independent sampling methods, each method respectively operates on the correlations through the parameters and the sampling of each parameter independently, hence Optuna can handle diverse algorithms including Tree-structured Parzen Estimator (TPE) [4] or Covariance Matrix Adaptation of Evolution Strategies (CMA-ES) [19].
In addition, the cost-effectiveness of the hyperparameter optimization framework is determined by the efficiency of searching strategy and the performance estimation strategy. For ensuring the "cost" part of the cost-effectiveness the pruning algorithm is used in two phases [1]:

- It systematically checks the intermediate function objective values.
- Closes a trial that does not meet the prearranged conditions.

Moreover pruning algorithm includes Asynchronous Successive Halving(ASHA) as an extension of successive halving [22]. The algorithm is fully oriented for applications in a distributional environment. The pruning algorithm is implemented with the following details:

input : target trial *trial*, current step step, minimum resource r,reduction factor η, minimum early-stopping rate s.
output: *true* if the trial should be pruned, *false* otherwise.

```
rung ←max(0,log η⌊step/r⌋)−s) ;
if step ≠ rη^(s+rung) then
    return
end
value ← get_trial_intermediate_value(trial,step)
values ← get_all_trial_intermediate_value(step)
top_k_values ← top_k(values, ⌊|values|/η⌋)
if top_k_values=∅ then
    top_k_values ← top_k (values,1)
end
return value ∉← top_k_values
```

Algorithm 1: Pruning algorithm based on Sucessive Halving

Algorithm [1] is Optuna's current pruning algorithm. The algorithm inputs include the trial liable to pruning, reduction factor, number of steps, a minimum resource to utilize before pruning, and minimal early stopping rate. The algorithm begins with the computing of the current trial, which is the number of times the trial remains the pruning.If the provisional ranking is in the top $1/\eta$, it is permitted to enter the next round of the competition. If the number of trials with a similar rung is smaller than η ,it promotes the best trials between the trials with the same rung.This solution does not allow repechage to prevent the need to record a massive number of checkpointed configurations (snapshots). Another important criteria of design is the scalable, versatile and easy-to-setup architecture design that allows to manage different types of tasks and a wide variety of applications, such as high computational experiments with many threads or lightweight computational via web interface, even it is possible the integration with docker-containers techniques [1].

3.6 Random Forest Algorithm

Namely, Random Forest (RF) is an ensemble learning algorithm based on the 'bagging' method of trees. The trees are independently built using samples from the dataset, and a majority vote is taken for the prediction. This means that it operates by creating multiple decision trees during training and outputs the class or the mean prediction for regression. Each node is split based on the best predictors from a randomly chosen subset of trees. The algorithm draws bootstrap samples from the data, grows a classification or regression tree, and then predicts new data by aggregating predictions from the developed forest of trees. The results include a measure of feature importance (i.e., the value gained by including a certain feature) and a confusion matrix displaying the accurate and inaccurate predictions. Intuitively, for each decision tree constructed during training, the classification or prediction is based on averaging the result of each decision tree. Therefore, it is suitable for both classification and regression tasks[30]. An ensemble approach is a methodology that incorporates several machine learning algorithms' predictions to create predictions that are more reliable than any other model. An Ensemble model is considered a model consisting of many models [6]. Ensemble learning is two types:

- Boosting means the use of weighted averages of a group of algorithms to allow poor learners to become more stronger. Collaboratively boosting fits, so that each running model decides in what the next model will focus on features.

- Bootstrap Aggregation (Bagging) means random sampling with replacements. Bootstrap enables one to consider the bias and volatility of the data set better. Bootstrap consists of a random sampling of a small subset of data from the dataset.

Generally, decision-trees algorithm is a generic method that may be utilized to minimize the variation of this highly variant. Bagging allows each model to operate separately before aggregating the outputs without regard for any model. As we will see below, decision trees present some problems and in consequence sensitive to the specific data on which they are trained. If the training data is altered, the resultant decision tree and, as a result, the predictions might be substantially different.
In addition, decision trees are computationally expensive for training, have a high risk of overfitting, and tend to find local optimum since after splitting they cannot retrace.[52]
A random forest is a meta-estimator, which incorporates several decision trees with some helpful changes, (i.e. it combines the result of many predictions).[55]

- The node probability can be determined by the number of samples that reach the node, divided by the total number of samples.

- When generating splits, each tree draws a random sample from the original data set, which adds a random element that prevents overfitting. The above-mentioned modifications avoid too highly correlation between the trees.

Advantages of Random Forest:

a) It is one of the most accurate learning algorithms available, and for several data sets it provides an extremely accurate classifier.

b) It can handle thousands of input variables without variable deletion.

c) It provides an unbiased internal assessment of the generalization error in the forest building.

d) It has an accurate way of estimating missing data and preserves accuracy when a significant part of the data is missing.

Disadvantages of Random Forest:

a) Random forests have been identified as overfitting some datasets with noisy classification/regression tasks.

b) Random forests prefer specific features with higher data levels,particularly categorical variables with various levels. For these type of data, the variable importance values are not trustworthy from RF values.

Regarding the regression tree used in this study,it is constructed via a process known as binary recursive partitioning. This is an iterative process splits the data into partitions or branches and divides every partition into smaller groups as the method progresses through each branch. The

splitting of regression trees is done according to the squared residuals minimization algorithm which involves the predicted sum variances for two resulting nodes should be minimized. Then each of the new branches applies this splitting rule. It continues until the maximum tree is built, which ensures that the splitting has been completed with the final observations in the training set. Because the maximum tree may become substantially big, pruning could be necessary to delete irrelevant nodes in the case of the regression trees.[46] Moreover, it is known that the Random Forest model is powerful and accurate, it usually performs great on many problems, including features with non-linear relationships. Furthermore, RF corrects the prediction of each tree and therefore tries to prevent the data overfitting that may easily occur. However, it comes with high variability in the results. We use Random Forest Regressor for each dataset, and the metric used to evaluate the algorithm is mean squared error (MSE) based on the best hyperparameter settings.[5]

3.7 Performance Evaluation

The method of determining how effectively the pipeline is fulfilling the regression analysis in order to obtain the most reliable predictions is known as performance evaluation. For a better understanding, a brief introduction to correlation is needed before embarking on performance evaluation. Correlation is a proportion of the relationship between two variables and indicates that when one variable's value varies, the other continues to shift in a specific direction. Two variables are said to correlate if an adjustment in one of them is joined by a predictable change in the other.[48]
Understanding the relationship is useful, since we can use one variable's value to predict the other.The idea of correlation is usually experienced in the scope of techniques used in modeling and business forecasting. Correlation describes the association in a way that allows the researcher to easily interpret the strength of the association. Correlation is, in essence, standardized covariance and it is defined as the covariance divided by the standard deviations of each variable:

$$\rho(X,Y) = \frac{cov(X,Y)}{\sigma_X \sigma_Y} \tag{2}$$

Dividing the standard deviations serves the purpose of rescaling the statistic so that the maximum and minimum values are always 1.0 and -1.0, respectively. Thus, a correlation of 0.6 means the same thing in terms of strength, regardless of the standard deviation of the variables. There are various forms of correlation equations, but for this thesis, the Pearson Product Moment Correlation (PPMCC) (2) will be used, which summarizes the intensity and orientation of the relationship between two variables in a single number, the correlation coefficient, also known as the Pearson coefficient or R score. As a result, a positive coefficient denotes a positive relationship, while a negative coefficient denotes a negative relationship. A zero correlation means that between the two variables there is no interaction. The proximity of the coefficient to $+1$ or -1 indicates the frequency of the variable's relationship. The idea that both correlation and regression analysis will approximate correlations between different variables is a function that they share. However, the researcher indulging in regression usually wants to find out the causal effect of one variable upon another. To explore such issues, data on the variables of interest are assembled and regression techniques are applied to estimate the effect of the causal variables upon the variable that they influence. Regression analysis includes several techniques for modeling and analyzing several variables when the focus is on

the relationship between a dependent variable (target variable) and one or more independent variables (predictors).

3.7.1 Evaluation with $\mathbf{R}^2$

The predictions of each model are evaluated with the coefficient of determination (R^2) metric, see equation (3), whereby T denotes the true values and P the predicted values. R^2 describes the proportion of variance that is explained by the model.It has no lower limit and the upper limit is 1, which expresses a perfect fit of the model.[49]

$$R^2 = 1 - \frac{\sum_{i=1}(T_i - P_i)^2}{\sum_{i=1}(T_i - \overline{T})^2} \tag{3}$$

A dataset with n values marked T_1,...,T_n, each related with a predicted value P_1,...,P_n, the residuals as T_i-P_i and $\overline{T}$ is the mean of observed data [49]. Therefore, with two square sums the validity of the data set is measured (the sum of squares and the residual sum of squares).

3.7.2 Evaluation with Spearman

The Spearman ranking correlation is widely used to evaluate the degree of association between two variables. Therefore, it is a non-parameter test that must be ordinary and the scores for one variable must be monotonically linked to the other variable, with n indicating the number of observations and d is the difference in the ranking among two ranks of each observation:

$d_i = \mathrm{rg}(P_i)\text{-}\mathrm{rg}(T_i)$.

$$\rho = 1 - \frac{6\sum d_i^2}{n(n^2-1)} \tag{4}$$

3.7.3 Evaluation with Pearson

The frequency of a relationship between two variables is measured by Pearson's correlation (Truevalue and Prediction value). Both the strength and the weight of this relationship should be assessed. While it is determined by the coefficient of determination r^2, the correlation coefficient *r* indicates the strength of a relationship. The weightiness of the relationship is noted in probability measures *p*. The value *p* expresses how unlikely is a given correlation coefficient *r* when no relationship exists in the population. Therefore, the larger correlation *r*, the stronger relationship exists,while a more significant relationship is determined by a smaller *p*.[37]

$$r_{PT} = \frac{\sum_{i=1}^{n}(P_i - \overline{P})(T_i - \overline{T})}{\sqrt{\sum_{i=1}^{n}(P_i - \overline{P})^2}\sqrt{\sum_{i=1}^{n}(T_i - \overline{T})^2}}$$

(5)

3.7.4 Evaluation with MAE

The mean absolute error is the measurement of absolute errors between a set of observations, which means, the average magnitude of errors within predicted versus observed values.

$$MAE(T, P) = \frac{1}{n}\sum_{i=1}^{n}|P_i - T_i| \tag{6}$$

3.7.5 Evaluation with MSE

The mean-squared error is a measure of the average squared error, which means, the average squared difference within the predicted values and the observed values.It measures model quality, bringing the value closer to zero results in the best models.

$$MSE(T, P) = \frac{1}{n}\sum_{i=1}^{n}(T_i - P_i)^2 \tag{7}$$

The above evaluations are a good indicator of average model performance. They are easy to compute and to understand. It can be computed with any kind of variable be it independent or dependent.
On the other hand, performance evaluation can be very time-consuming to determine the strengths and limitations of the model before comparing it with a model performer. If the model performer is overly good, it will be difficult to achieve the same performance evaluations.

4 Experiments

The three datasets (LANL Earthquake, LUMC, and EEG for Age Prediction) were carefully selected to be used in the training and testing stages for performing the experiments. There are multivariate time series datasets, specifically LANL Earthquake and EEG for Age Prediction are part of the Kaggle dataset repository, selected as validity datasets because the prediction methods are aligned with our prediction method design. The models are trained and tested with normalized features. This set-up will result in a total of 3 (datasets) * 5 (variables to predict) using the regression model performed based on the Scikit-Learn library. To validate the results of the automated regression pipeline and, its modules, metrics, and other characteristics, a set of prediction experiments were performed with different time series data sets to cross-validate the results obtained with the coefficient determination for the target prediction. These experiments prove a good validation and the reliability necessary for having good criteria of the pipeline functioning and the approach taken. Thus, the experiments are divided into three sections: (1) Performance on the LANL Earthquake dataset, (2) performance on the EEG for Age Prediction dataset, and (3) performance on the LUMC dataset case study. In our experiments, all methods are run using the k-fold cross-validation scheme, with k set to 5. This is chosen for a good balance between computational cost and solution accuracy.

4.1 Study case: Predicting the time to failure in LANL Earthquake Prediction

The prediction of earthquakes is one of the most important topics in seismology [25]. Three important aspects are key points on earthquake predictions: when, where, and magnitude. So, when the earthquake will take place, we will address it. In particular, the study analyze a snapshot of the continuous seismic signals recorded in a fault shear zone to predict the failure time (unconnected decision tree models have been created to predict the instantaneous fault shear stress or displacement).The problem, given as a regression, employs the seismic data continuously recorded by Los Alamos National Laboratory as the input and the fault time to failure as the target. Thus,the shear stress signal measured on the device is used to predict the time to failure. [42]
The data fields are:

- acoustic data - the seismic signal.
- time to failure - the time (in seconds) before the next displacement will occur.
- seg id - the test segment ids to be predicted (one prediction per segment).

Hence, applying ML to the dataset can predict the remaining time with precisely before it fails.These forecasts are based just on the physical features of the sound signal instantaneously and don't utilize its history [25]. To the best of our knowledge, ML is used for the first time in acoustic/seismic continuous data to infer failure times and we demonstrate the exact failure forecast ML applied in this experiment, based on instantaneous acoustic signal analysis at any time during a slipped cycle and revealing a previously unidentified signal. For this reason it has been used as validation dataset because of the good results of the ML analysis of earth seismic data. While there are 2624 test signals provided by Kaggle, only 341 are used for the experiments of public leader board [35]. From the original results [35], a naïve model that is

just based on the periodicity of the events (average interevent time) only gets a coefficient determination R^2 of 0.3. The time to failure of the RF tuned model in comparison is highly exact with an R^2 value of 0.89. Surprisingly, the RF tuned model accurately predicts failure not just when failure is imminent, it also demonstrates that the system is progressing continuously towards failure.

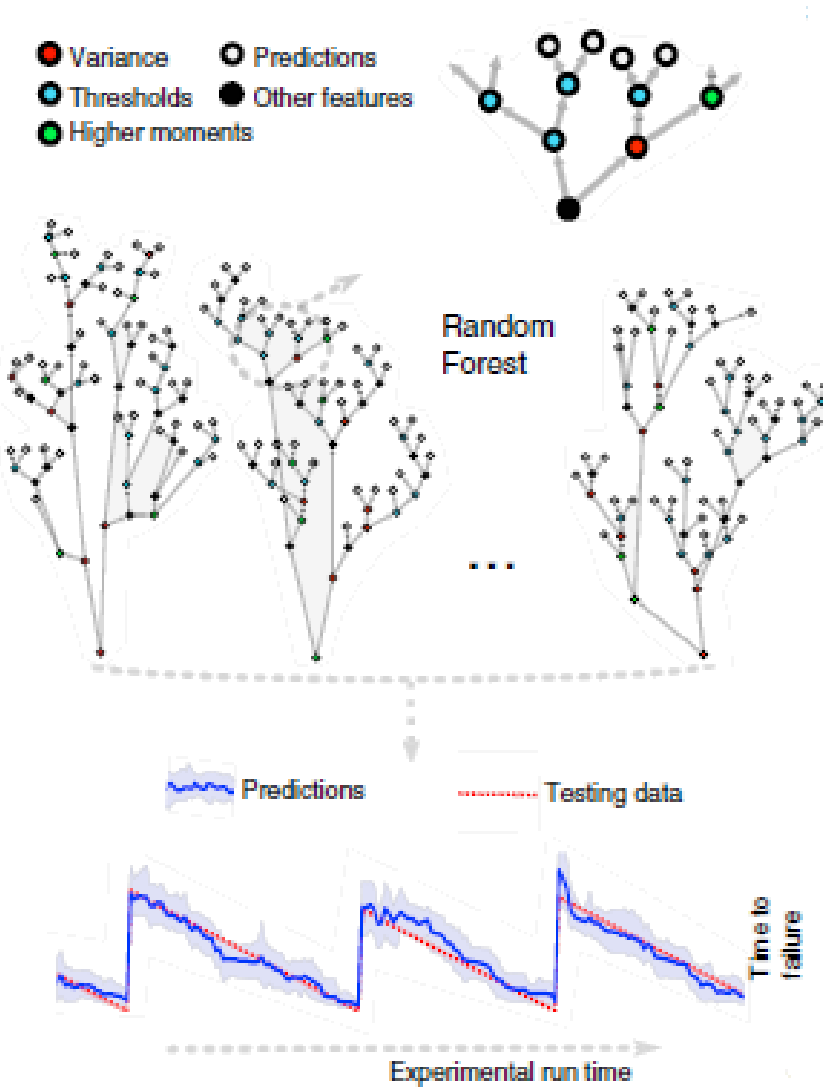

Figure 4: Explaining how a RF model tree prediction works e.g. earthquake experiment Averaging the predictions of a number of decision trees (e.g. 10000) for the next failure, the RF model predicts the time left until the next failure. On top, each tree produces its prediction after a number of judgments (colored nodes) depending on the earthquake acoustic signal characteristics. On the bottom, the RF (blue line) prediction on unseen data (testing data) with confidence intervals of 90 per cent (blue shaded region).[42]

4.2 Study case: Predicting the age from EEG

The dataset is retrieved from the publicly accessible EEG Corpus Temple University and is part of the "TUH Abnormal EEG Corpus" in particular. The initial dataset is primarily intended to differentiate between normal and pathological EEG signals. However, we would like to examine normal EEG alone as the purpose of this experiment is to predict patient's age from the person's normal EEG signal. Because the corpus consists of EEG, fMRI, and fNIRS [16]- as a whole, to make accurate predictions about individuals' brain maturity across development the EEG is the most recommended. [16]. Historically, EEG was used to diagnose epilepsy, thus the majority of the time the work was about processing and interpreting the abnormalities. Researchers began addressing the standard EEG signal for numerous applications recently, as machine learning techniques flourished[2]. The data contains 1297 patients' recordings of slightly various lengths. The original data consists of an EEG signal with the patient's information including the age information. Thus the files contain the age in the first row and the second row contains the EEG channels' names used in the recording, and the signal data follows from the third row. In the original study [2], a crucial step to achieving the best age estimate was the selection of appropriate ML algorithms to offer a better age prediction, various ML methods were tested. Realizing the following regression algorithms: Elastic Net (ENet), Vector Regression Support (SVR), Random Favor (RF), Extreme Gradient Boost Tree (XgbTree), and Polynomial Kernel Gaussian Process (gaussprPoly). This study gives us a clear view to compare with our RF results in the next section.

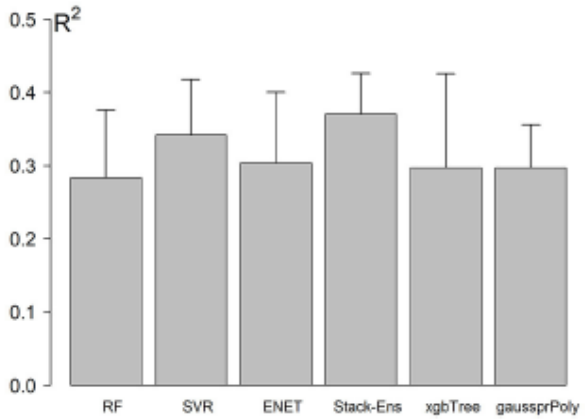

Figure 5: Models performance searching the best accuracy.
The individual performance for each ML method was calculated. Error bar represents the standard deviation of performances across the outer loop of Dataset. The results showed that stack-ensemble improved the overall performance with $R^2 = 0.37$ (0.064), MAE = 6.87(0.69) years, and RMSE = 8.46 (0.59) years. Followed by SVR that achieved the coefficient determination $R^2 = 0.34$ (0.056), MAE = 7.01(0.68) years and RMSE = 8.7(0.63) years. On the other hand, RF algorithm was the last of this comparison in the evaluation performance [2].

The age and predicted age correlation was r = 0.6. The feature importance revealed that age predictors are spread out across different feature types. When testing on 50 samples, the overall coefficient determination was $R^2 = 0.26$, which shows that there is no linear relation between the features and the target age from the small number of samples [15].

4.3 Study case: Predicting the neurological mental state from EEG

The data utilized in this thesis contain EEG records of 125 LUMC patients (Leiden University Medical Center) [27]. Three labels have been selected for prediction among those individuals with distinct clinical conditions (apathy, depression, anxiety). Each patient's EEG has been collected from five distinct epochs (EEG segments). The data were recorded with a 500 Hz (Analog to Digital) sample rate, a 16 bit AD conversion, and a 0.16 70.0 Hz band filter. Due to the sample rate and 8,192 seconds length, 4096 data points are contained in the EEG Epoch. All epochs per patient originate from the same recording (total EEG lasts roughly 15-20 minutes), but we selected 5 epochs of 8.192 seconds from the total EEG to ensure all time-series are artifact-free [27]. Every patient's epoch is based on the same record, but we have picked 5 epochs of about 8,192 seconds from the entire EEG to guarantee that all-time series are artifact-free. Unfortunately, we do not have previous results of regression to compare on these patient's datasets to validate the good performance of the pipeline performed.

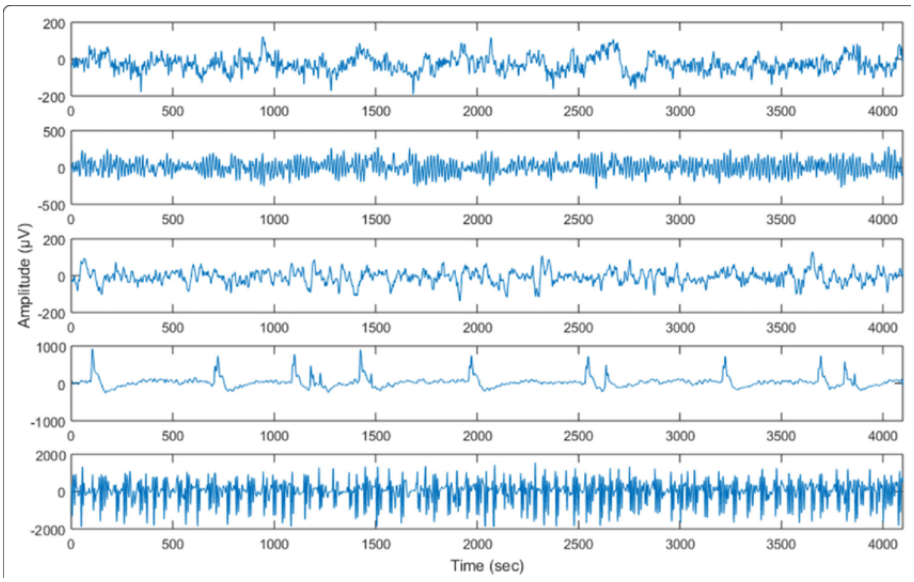

Figure 6: Typical EEG recordings e.g.five sets of the Bonn EEG database.
For showing the spectral information of the EEG signal concerning epilepsy is showed above.There are defined a number of spectral thresholds, and each frequency sub-band combination is generated [51].

Clinicians label the neurological mental state which means, the variables Anxiety, Depression and Apathy (range 0 - 30) are reflected on a scale after routine neuropsychological assessments of patients [27].

5 Results and discussion

In this section, we discuss the results, our research goals and remark on different implementation aspects. We also compare our findings with the related studies and emphasize the discrepancies in several respects. The main objective of this thesis is to validate for the time series of aforementioned data sets the good performance of the automated regression pipeline. The performance evaluation techniques utilized during these experiments have been explained in the Methods section.The experiments that have been carried out can be found in table 2, which provides a summary of studies that specifically reported the results performance from the experiments.

LANL Earthquake	Time to failure Test data R^2-score:0.559 Test data Spearman:0.584 Test data Pearson:0.748		
LUMC	Apathy Test data R^2-score:0.317 Test data Spearman:0.510 Test data Pearson:0.907	Depression Test data R^2-score:0.297 Test data Spearman:0.480 Test data Pearson:0.526	Anxiety R^2-score:0.207 Spearman:0.332 Pearson:0.364
EEG for Age Prediction	Age Test data R^2-score:0.431 Test data Spearman:1.266 Test data Pearson:1.249		

Table 2: A summary of experiments performed in this Thesis

5.1 Experiment 1 Results

Our objective in this empirical study is to make a comparative performance evaluation between the obtained and the original results, as previously described in section 4.1, the basic model only gets an R^2 of 0.3. instead a RF tuned model reached an R^2 value of 0.89.

R^2	Spearman	Pearson
0.559	0.584	0.748
0.517	0.581	0.721
0.508	0.582	0.694
0.497	0.617	0.707

Table 3: Experiments 1: Earthquake testing error metrics.

The testing error metrics results for experiment 1 are shown in table 3, which displays the metrics of section 3.7 to evaluate the performance of the regression pipeline. Each column corresponds to an error metric and each row refers to each of the four experiments models.
A relationship plot to see the relation between a given variable and target variable, acoustic

data (displayed in blue) against the time to failure (displayed in yellow) is shown by Figure bellow.

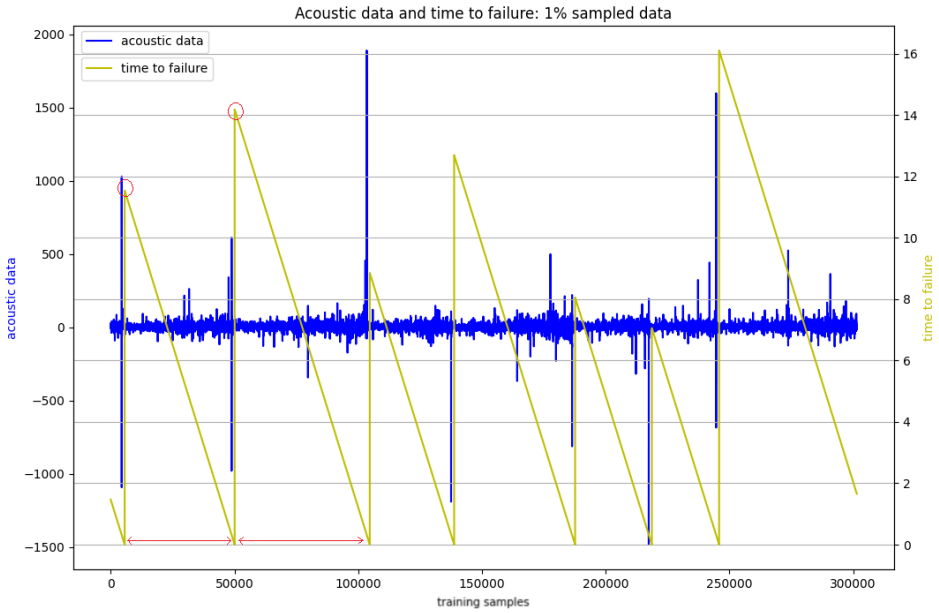

Figure 7: Representation of Acoustic data and time to failure.
Representation of 1% of the earthquake acoustic data -blue- and time to failure -yellow- (in the y-axis) over different samples i.e. time (x-axis). The probable time for the next earthquake can be determined by the value of the top spikes in the time to failure graph (e.g. in sample no. 50000 time to failure is 1500, and the time to the next earthquake in sample no. 10500 is significantly higher than in the previous occurrence, where time to failure was 900), which was obtained from the previous real-time acoustic data.

A scatter plot of the testing predictions (displayed in y axis) against the actual values (displayed in x axis) is shown in Figure 8. The reader might find useful this figure to appreciate a bit of positive correlation. Although it is possible to fit the regression when one of the variables takes discrete values, however, the simple Scatterplot produced by this dataset is not optimal.

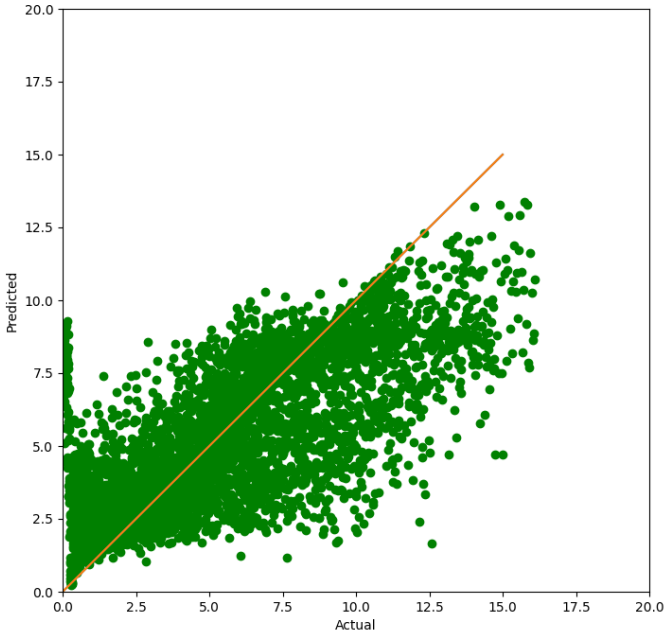

Figure 8: Scatter plot for earthquake time-to-failure predictions.
(time predicted for the next earthquake event) vs actual time events (629,145,481 samples). A positive correlation can be easily observed in the main group of values. We can also observe that predictions for time-to-failure events in the extremes (i.e. close to 0 / imminent, or close to 16 / happening very late) are much less correlated than the majority of predictions for time-to-failure values between those extremes.

On the whole, the accuracy achieved with our regression pipeline in this set of experiments is acceptable compared to previous works mentioned above, however, we observed different issues that have prevented far higher accuracy, one of them is the data has strong multicollinearity, that occurs when independent variables in the regression model are correlated, which is a problem because the variables should be independent and the correlation degree between them should be less, this can cause problems when we fitting the model and interpreting the results. Another problem is caused by the overfitting, hyperparameter tuning was performed by Optuna to reduce overfitting automatically[17]. Such as the number of leaves was decreased and the minimum data in a leaf increased. Also tried was increasing the number of data rows to 200,000. This resulted in worse overfitting and a lower R^2 score. Because this made a computationally intensive approach to the problem even more computationally difficult, this effort was quickly dropped. It is a well-known fact that Random Forest Regressor has low performance compared to LightGBM[26] or XGBoost[10], which would be a definitive method to boost the accuracy.

5.2 Experiment 2 Results

Our aim in this second study is to make again a comparative performance evaluation between the obtained and the original results, as previously mentioned the overall accuracy when testing on 50 samples,was $R^2 = 0.26$ and the best accuracy achieved was $R^2 = 0.34$ in overall.

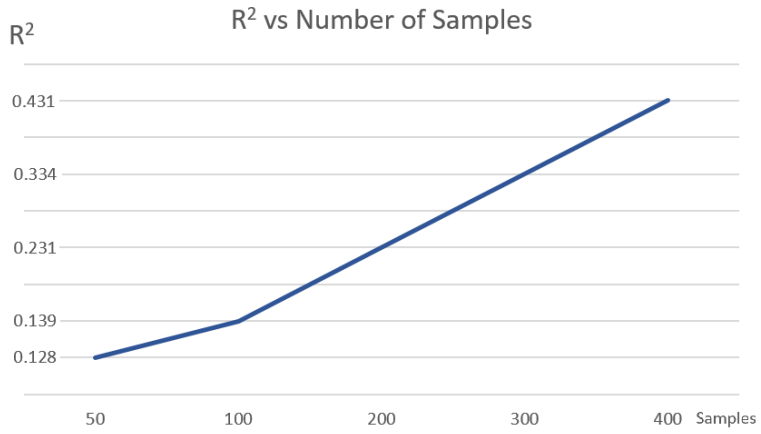

Figure 9: The effect of the number of samples on the age prediction by using the coefficient of determination.
We can observe that from 200 samples there is a linear relation between features and age.

The testing error metrics results for experiment 2 are shown in table 4, which displays the metrics of section 3.7 to evaluate the performance of the regression pipeline. Each column corresponds to an error metric and each row refers to each of the four experiments models.

R^2	Spearman	Pearson
0.139	0.423	0.402
0.231	0.678	0.669
0.334	0.881	0.968
0.431	1.266	1.249

Table 4: Experiments 2: Age Prediction testing error metrics.

The findings of the experiments suggested the aging alters the EEG signals in the brain. Thus, feature extraction is required from EEG signals to capture the relationship between the signals and the age predictors.To represent the impact of the features extraction part (tsfresh), we selected the best features (EfficientFCParameters) to improve the performance and reduce the complexity of the model. In this way, we eliminated the correlated features to select the best features, which improve the overall R^2. As a comparative using the MinimalFCParameters the results are very poor and a very low R^2. From table 4, we realize the best results were achieved $R^2 = 0.431$ and were slightly improved the results from [2], which shows the ability of our model to predict the age. Regarding the Pearson correlation coefficient of 1.249, the possible range of values for the correlation coefficient is -1.0 to 1.0. In other words, the values cannot exceed 1.0 or be less than -1.0.Therefore a correlation of -1.0 indicates a perfect negative correlation, and a correlation of 1.0 indicates a perfect positive correlation. Due to multicollinearity the

standardized regression weight can be exceed the bounds of (-1,1).Accordingly, it is a clear indication of strong multicollinearity present among the set of independent variables, and the correlation of each of the independent variables with the dependent variable. Consequently, it will be greater than one if there are 2 or more predictors that are correlated, positively or negatively, then the Beta values may exceed those bounds (-1, +1).[24]
A better modeling approach is needed. Therefore, the results show that additional samples may be possible for a potential improvement.

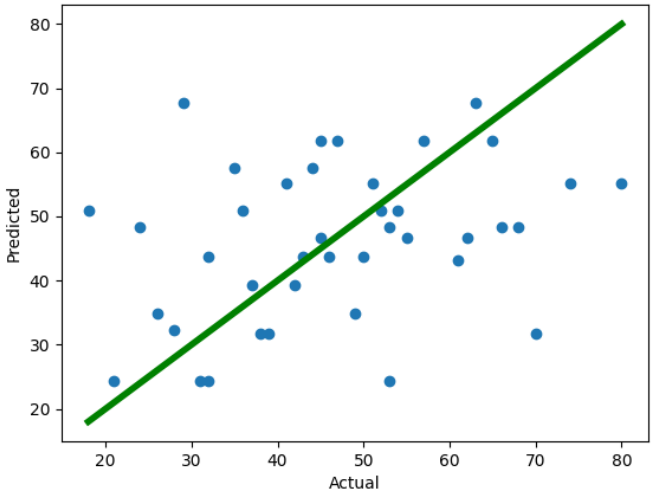

Figure 10: Scatter plot for age prediction vs actual person age (40 samples).
We can observe some degree of positive correlation. We can also observe some tendency of overestimating the lowest age ranges (below 23 years), and under estimating the higher age ranges (above 67 years)

Figure 10 shows an absolutely high positive correlation, the closer the data points come when plotted to make a straight line, the higher the correlation between the two variables, or the stronger the relationship, and we were able to offer a reasonable age forecast with an unbiased prediction of age.

5.3 Experiment 3 Results

The last experiment, as we mentioned above utilized the EEG records of 125 LUMC patients and we are not able to validate the good performance with previous metric outcomes. The testing error metrics results for experiment 3 are shown in table 5,6 and 7 which display the metrics of section 3.7 to evaluate the performance of the regression pipeline. Each column corresponds to an error metric and each row refers to each of the four experiments models.
In figure 11 we can observe some degree of positive correlation, although the smallest eigenvalue is indicating that there are strong multicollinearity problems, which means the coefficient estimates can swing wildly based on which other independent variables are in the model and the precision of the estimated coefficients are reduced, weakening the regression model.

R^2	Spearman	Pearson
0.109	0.176	0.193
0.147	0.237	0.261
0.178	0.288	0.315
0.207	0.332	0.364

Table 5: Experiments 3: Anxiety testing error metrics.

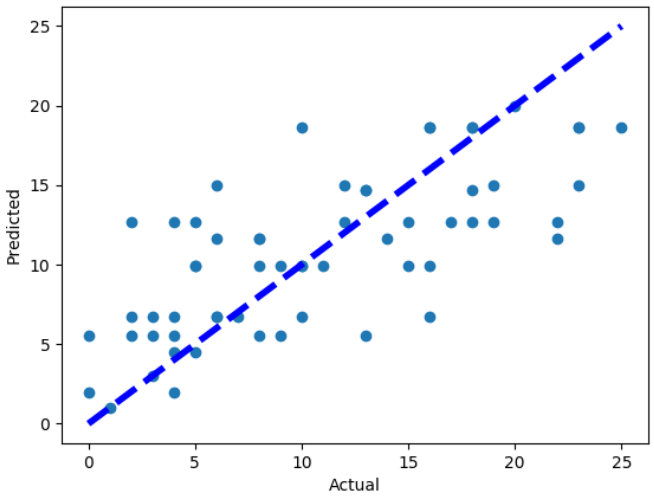

Figure 11: Scatter plot for mental anxiety predictions vs actual anxiety values.

Figure 12 shows positive correlation, especially in depression values below the higher range (above 25), where we can observe an underestimation of mental depression.

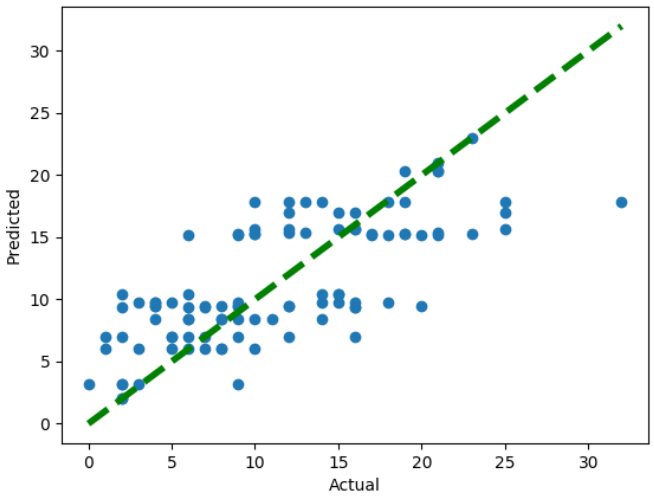

Figure 12: Scatter plot for mental depression predictions vs actual depression values.

R^2	Spearman	Pearson
0.139	0.225	0.247
0.208	0.346	0.369
0.217	0.351	0.375
0.297	0.480	0.526

Table 6: Experiments 3: Depression testing error metrics.

R^2	Spearman	Pearson
0.127	0.205	0.364
0.203	0.317	0.580
0.308	0.485	0.880
0.317	0.510	0.907

Table 7: Experiments 3: Apathy testing error metrics.

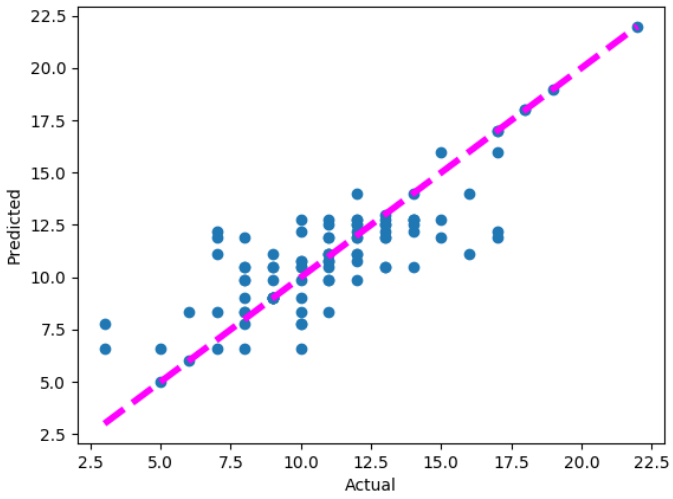

Figure 13: Scatter plot for mental apathy predictions vs actual apathy values.

Figure 13 shows a positive correlation, we also observed in the results overfitting indicating that a statistic model begins to describe a random error in the data instead of the relationship between variables[17]. In our regression pipeline, overfitting produced misleading R-squared values and regression coefficients. One of the problems is that the data has collinearity, and another problem is caused by the predictors, because of too many predictors chasing too little information.

6 Conclusion

The main insight of this Thesis is to propose an automated time-series regression pipeline as an effective and innovative model for predicting different time-series variables in unrelated domains, and for this purpose specifically tackling in the domains of medical (mental conditions prediction), earthquake and age predictions.
This task was undertaken to facilitate the resolution of problems present in predictions using time series regression, and the review of the outcomes from the pipelines and experiments developed show novelty problem solving approaches in this field.
The use of tsfresh proposes an effective approach to cover the feature extraction capabilities required for time series. As an example, its use with the EEG data was seeking to obtain a minimal or efficient set of features or variables due to the very large dataset utilized. Additionally, feature selection implemented through RFECV provides an additional layer of discrimination in the pipeline to pinpoint and extract the most relevant features. Moreover, the combination of the Hyperparameter optimization methodology Optuna and efficient modeling technique with the use of Random Forest Regressor introduced a novel blended technique.
Three different experiments were implemented to determine the suitability and validity of the automated regression pipeline, and also which combinations may offer the best precision. As a result, high prediction metrics were obtained in the first two experiments that indicate the effectiveness of the regression-based prediction model proposed. The experiments previously presented with their respective results in Section 5, give the reader a visual idea of the different experiments that were developed. The so-called experiments 1 (earthquake time-to-failure prediction) and 2 (age prediction) showed overall better prediction results and "goodness of fit" than experiment 3 (mental condition). Experiment 3 is a real data experiment from LUMC, where no data cleaning or data adjustments were performed before providing the data into the pipeline, with the except for dropping all the highly correlated variables identified.
From the outcome, it is reasonable to say that, even though some degree of correlation is obtained in experiment 3, the models built out of this experiment may be less robust than the ones built in experiments 1 and 2. Another reason for obtaining less correlated values may be the very nature of the data, as the process of obtaining mental condition values (anxiety, apathy and depression) over a large scale (0-25) may be less objective than e.g. using sensors, because these values come from clinician registries of responses to predefined questions in patient interviews that may be subjectively responded. Additionally, when it comes to the overall coefficient of determination results, it can be seen that the model achieves significantly better results than the experiments for LUMC, being evident that the best results overall are achieved by experiment 1 in terms of error metrics. The overall conclusion from the obtained metrics is that the models used in experiments 1 and 2 may be harder to use for predictions in the medical domain, at least with values obtained through interview questions, i.e. this may not be the case with accurate values provided by sensors such as in ECG studies.

More specific observations from the outcomes of the proposed pipeline are that, overall, values in the extremes of the range (e.g. earthquake predictions close to 0 i.e. imminent, or close to 16 i.e. happening in a long time; age predictions below 23 or over 67 years; and mental conditions estimated in the extremes of the scale e.g. above 25 when the maximum is 30) are much less correlated than the rest. This may be due to shortcomings of the pipeline proposed, or to the inherent data these values represent (as extreme values are more difficult to predict depending on the domain, such as age, or, in the case of earthquakes, estimating times that

goes to the extreme e.g. imminent or very late, which occurs less often). As the initial proposal of this study was to find an effective and novel automated time-series regression pipeline that could serve as an accurate model to predict real-world data specifically covering the medical domain, the research could not provide high enough accuracy for these predictions. However, the medical domain where the data was obtained (unlike in experiment 1) was psychiatry, i.e. values were not sensor-based as opposed to medical domains such as cardiology and ECG values. As indicated above, the possible subjectivity introduced in responses to interview questions may have an effect, so the application of our pipeline to other medical domains may be possible.
In summary, this automated time-series regression pipeline is an efficient and innovative regression analysis technique that can be implemented in a number of domains. A lot of these implementations are used in machine learning to predict numerical values in different areas, such as predicting sales or house prices, also to predict the number of sensor's failures, or even to predict medical conditions. Moreover, the large datasets utilised in some of the experiments include sufficient variable resolution to predict with better performance and less noise, avoiding the heteroscedasticity and multicollinearity in time-series models. Therefore, we figured out that time-series problems may require the identification of additional alternative approaches instead of a Random Forest Regressor that may generalize better in a wider variety of domains, such as Stack-Ensemble or Vector Regression Support (Figure 5). Also, it may be interesting to check the effect of the isolated relationship between each independent variable and the dependent variable also if the dependent variable changes significantly from the beginning to the end of the time series, which could not be considered in this study due to time constraints.

7 Future Work

Further work can be foreseen to improve the performances reported in the results of this Thesis. Given the nature of the indicated pipelines and datasets, additional experiments may be readily performed by simply employing more complex approaches for data sampling, feature selection or regression algorithms. Due to the time constraints in this study, and the computation times required for testing the present and discarded elements and techniques in the pipeline, we were not able to explore additional or more complex techniques. Based on the results, it would be advisable for future work to explore additional and more complex prediction models to better handle complex domains such as mental conditions from observed and annotated data of patient responses to interview questions. We further propose two different approaches for this: First, reconsidering some modules of the regression pipeline, specifically the part of the model that combines the feature selection with the Hyperparameter optimization; both are directly related to the best performance of the pipeline and therefore impacts directly on the metrics of the experiments, as this could provide major improvements in the outcomes of the metrics.
Our second proposition is to introduce a "black-box" based on an ensemble model, for example, Catboost or SHAP. The characteristics and dynamics of both approaches are open for the reader to work on. CatBoost has a high reputation and it is a contemporary gradient boosting engine. The author had little time to fully investigate it, and may have been impeded by a lack of familiarity with the algorithm. Spending time on Random Forest Regressor, Support Vector Machines or Nearest Neighbors Regression algorithms is probably not worth it; compared to other regression models, in the author's view, these are not state-of-the-art and appear to

underperform newer gradient boosting decision tree-based approaches such as LightGBM or XGBoost. The approach that we have developed can also be applied as a regression pipeline performing a quick Catboost with hyperparameter tuning to generate an accurate model that could later be used in complex domains such as the specific medical domain tested (mental conditions) or other applications.

A Appendix

All experiments in this thesis were run on the following system:

- Operating system: Ubuntu 20.04 LTS 64-bit
- Processor: Intel CoreTMi9©8950HK 2.9GHz
- Memory: 32 GiB DDR4 2933Mhz Non-ECC Memory
- Graphics: NVIDIA®Quadro P620

Here we provide the source code of the Automated Regression Pipeline for Time-Series, used in all experiments utilized in this Thesis as the main code, although the data cleaning part may differ, we will only include one of the variables predicted (Anxiety), hence the rest variables differ in their respective code sections.

```
# importing modules

import pandas as pd

import time

from datetime import datetime

import pickle as pkl

import numpy as np

import logging

import seaborn as sns

import optuna

import sklearn

from sklearn import ensemble

import matplotlib.pyplot as plt

from sklearn.metrics import confusion_matrix

from sklearn.metrics import r2_score

from scipy.stats import spearmanr, pearsonr

from sklearn.model_selection import train_test_split, cross_val_score

from sklearn.impute import SimpleImputer

from sklearn.preprocessing import LabelEncoder
```

```
from sklearn.utils.multiclass import type_of_target
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import RandomForestClassifier
from RFECV_feature_selection import RFECV_feature_selection
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import StratifiedKFold
from sklearn.metrics import r2_score, explained_variance_score, accuracy_score,
max_error, mean_absolute_error, mean_squared_error,
mean_squared_log_error, median_absolute_error


def objective(trial):


    X = data  # RFECV extracted data
    y = target  # labels
    # Invoke suggest methods of a Trial object to generate hyperparameters.

    rf_min_samples_leaf = trial.suggest_float('rf_min_samples_leaf', 0.01, 0.5,
    log=True)
    rf_min_samples_split = trial.suggest_float('rf_min_samples_split', 0.01, 1,
    log=True)
    rf_n_estimators = trial.suggest_int("rf_n_estimators", 10, 1200, log=True)
    rf_max_depth = trial.suggest_int("rf_max_depth", 10, 1200, log=True)

    regressor_obj = ensemble.RandomForestRegressor(max_depth=rf_max_depth,
                                                   n_estimators=rf_n_estimators,
```

```
                                                    min_samples_split=
                                                    rf_min_samples_split,
                                                    min_samples_leaf=
                                                    rf_min_samples_leaf)
    # Step 3: Scoring method:
    score = sklearn.model_selection.cross_val_score(regressor_obj, X, y, n_jobs=-1,
    cv=3)
    accuracy = score.mean()
    return accuracy

# Setting up the log file

suffix = time.strftime("%Y%m%d_%H%M%S")
logfile = './test/log_' + str(suffix)

logger = logging.getLogger('EEG')
logger.setLevel(logging.DEBUG)
formatter = logging.Formatter
('- %(asctime)s [%(levelname)s] -- ''[- %(process)d - %(name)s] %(message)s')

if logfile is not None:
    fh = logging.FileHandler(logfile)
    fh.setLevel(logging.DEBUG)
    fh.setFormatter(formatter)
    logger.addHandler(fh)

# Setting up the file for performance measure
```

```
# set suffix of filename:
description = '_log'

# Name of data set
name_dataset = 'EEG'
file_name = str(suffix) + '_' + name_dataset + '_performance_' +
description + '.txt'
file_name_inc = str(suffix) + '_' + name_dataset +
'_performance_incremental' + description + '.txt'
folder_name = 'experiments'
f_performance = open(folder_name + '/' + file_name, 'w+')
f_performance_inc = open(folder_name + '/' + file_name_inc, 'w+')

# Number of random forest iterations and CV
start_time = time.time()
iterations_rfr = 1
cv = 5  # Cross Validation 5.
counter = 0
cv_counter = 0
logger.info('Parameters are: CV=' + str(cv) + ', random forest iterations='
+ str(iterations_rfr))

# Loading data
logger.info('Loading data...')

X = pd.read_csv('eeg_ANX_extracted_TS.csv',index_col=[0])  # Tsfresh extracted data
X.fillna(X.mean(), inplace=True)
```

```
X.replace([np.inf, -np.inf], np.nan, inplace=True)
print(X)

X_RFECV = X
RF_SEED = 13
y = pd.read_csv('LABEL_ANX.csv', index_col=[0])  # labels

print(y)

def split_data_train_model(labels, data):
    # 20% examples in test data
    train, test, train_labels, test_labels = train_test_split(data,
                                                              labels,
                                                              test_size=0.2,
                                                              random_state=RF_SEED)
# training data fit

    return train, test, test_labels, train_labels
y_data = y
X_data = X

RFECV_features_per_split = []
feature_importance_per_rfr = []

data = X
target = np.isnan(y)
logger.info('Starting the cross-validation')
```

```
best_params_all = []

# Stratified split
skf = StratifiedKFold(n_splits=cv, random_state=np.random, shuffle=True)

best_params_all = []

for train_index, test_index in skf.split(X_RFECV, y['Anxiety']):

    cv_counter += 1
    logger.info('CV counter: {cv_counter}')
    logger.info('Started Feature Selection on the split')
    # print("Outer CV, TRAIN:", train_index, "TEST:", test_index)
    X_train, X_test = X_RFECV.iloc[train_index], X_RFECV.iloc[test_index]
    y_train, y_test = y['Anxiety'].iloc[train_index], y['Anxiety'].iloc[test_index]

    X_train, X_test, features_list = RFECV_feature_selection(X_train, X_test, y_train)
    feature_importances_ = RFECV_features_per_split.append(features_list)

    data = X_train
    target = y_train.values

    # Step 4: Running it
    study = optuna.create_study(direction="maximize")
    study.optimize(objective, n_trials=100)

    print(study.best_trial)
```

```
best_params_all.append(study.best_trial)
print(best_params_all[-1])
print(best_params_all[-1].params["rf_n_estimators"])
print()

for i in range(0, iterations_rfr):
    counter += 1
    logger.info('Counter is ' + str(counter))
    print('Counter is ' + str(counter))

    n_estimators = best_params_all[i].params["rf_n_estimators"]
    max_depth = best_params_all[i].params["rf_max_depth"]
    min_samples_leaf = best_params_all[i].params["rf_min_samples_leaf"]
    min_samples_split = best_params_all[i].params["rf_min_samples_split"]

    # We used the bestparams obtained in the objective
    # function in RandomForestRegressor
    rfr = RandomForestRegressor(n_estimators=n_estimators,
    max_depth=max_depth, bootstrap=False,
    min_samples_leaf=min_samples_leaf,
    min_samples_split=min_samples_split)

    rfr.fit(X_train, y_train)
    predictions_train = rfr.predict(X_train)
    print(predictions_train)
```

```
predictions_test = rfr.predict(X_test)
print(predictions_test)

test_score = r2_score(y_test, predictions_test)
spearman = spearmanr(y_test, predictions_test)
pearson = pearsonr(y_test, predictions_test)

print(f'Test data R-2 score: {test_score:>5.3}')
print(f'Test data Spearman correlation: {spearman[0]:.3}')
print(f'Test data Pearson correlation: {pearson[0]:.3}')
print('MSE: ', mean_squared_error(y_test, predictions_test))
print('MAE: ', mean_absolute_error(y_test, predictions_test))

confusion_matrix = pd.crosstab(y_test, predictions_test, rownames=['Actual'],
colnames=['Predicted'])
sns.heatmap(confusion_matrix, annot=True, vmin=0, vmax=40, linewidths=.2,
cmap="YlGnBu")
plt.show()
plt.savefig('confusion_matrixAnxiety.png')

fig, tx = plt.subplots()
tx.scatter(y_test, predictions_test)
tx.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--',
lw=4, color='green')
tx.set_xlabel('Actual')
tx.set_ylabel('Predicted')
plt.show()
```

```
plt.savefig('PLOTpredictAnxiety.png')


fig, tx = plt.subplots()
tx.scatter(y_train, predictions_train)
tx.plot([y_train.min(), y_train.max()], [y_train.min(), y_train.max()],
'k--', lw=4, color='green')
tx.set_xlabel('Actual')
tx.set_ylabel('Predicted')
plt.show()
plt.savefig('PLOTpredictAnxiety.png')


# predict y from the data
x_new = y_test
y_new = predictions_test


# plot the results
plt.figure(figsize=(4, 3))
ax = plt.axes()
ax.scatter(x_new, y_new)
ax.plot(x_new, y_new)
ax.set_xlabel('Actual')
ax.set_ylabel('Predictions')
ax.axis('tight')
plt.show()
plt.savefig('PLOTpredictAnxiety.png')


# regression plot using seaborn
```

```
fig = plt.figure(figsize=(10, 7))

sns.regplot(x=y_test, y=predictions_test, color='green', marker='+')


# legend, title, and labels.

plt.legend(labels=['Anxiety Test'])

plt.title('Prediction VS Actual', size=24)

plt.xlabel('Actual', size=18)

plt.ylabel('Predicted', size=18)

plt.show()

plt.savefig('RelationshipAnxiety.png')
```

References

[1] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama. Optuna: A next-generation hyperparameter optimization framework, 2019.

[2] O. Al Zoubi, C. Ki Wong, R. T. Kuplicki, H.-w. Yeh, A. Mayeli, H. Refai, M. Paulus, and J. Bodurka. Predicting age from brain eeg signals—a machine learning approach. *Frontiers in Aging Neuroscience*, 10:184, 2018. doi: 10.3389/fnagi.2018.00184.

[3] B. Bengfort and R. Bilbro. Yellowbrick: Visualizing the scikit-learn model selection process. *J. Open Source Softw.*, 4:1075, 2019.

[4] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. Algorithms for hyper-parameter optimization. In *NIPS*, 2011.

[5] G. Biau. Analysis of a random forests model. *Journal of Machine Learning Research*, 05 2010.

[6] G. Biau and E. Scornet. A random forest guided tour. *TEST*, 25, 11 2015. doi: 10.1007/s11749-016-0481-7.

[7] L. Breiman. Random forests. 45(1):5–32, Oct. 2001. ISSN 0885-6125. doi: 10.1023/A:1010933404324. URL `https://doi.org/10.1023/A:1010933404324`.

[8] L. Breiman and P. Spector. Submodel selection and evaluation in regression. the x-random case. *International Statistical Review / Revue Internationale de Statistique*, (3):291–319, 1992.

[9] K. Chen and M. B. Begum, Bagnall. The ucr time series classification archive. 02 2015. doi: www.cs.ucr.edu/~eamonn/time_series_data/.

[10] T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system. pages 785–794, 08 2016. doi: 10.1145/2939672.2939785.

[11] M. Christ, A. Kempa-Liehr, and M. Feindt. Distributed and parallel time series feature extraction for industrial big data applications. 10 2016.

[12] M. Christ, N. Braun, J. Neuffer, and A. Kempa-Liehr. Time series feature extraction on basis of scalable hypothesis tests (tsfresh – a python package). *Neurocomputing*, 307, 05 2018. doi: 10.1016/j.neucom.2018.03.067.

[13] K. R. Davenport T. The potential for artificial intelligence in healthcare. future healthcare journal, 03 2019.

[14] F. Degenhardt, S. Seifert, and S. Szymczak. Evaluation of variable selection methods for random forests and omics data sets. *Briefings in bioinformatics*, 20, 10 2017. doi: 10.1093/bib/bbx124.

[15] S. I. Dimitriadis and C. I. Salis. Mining time-resolved functional brain graphs to an eeg-based chronnectomic brain aged index (cbai). *Frontiers in Human Neuroscience*, 11:423, 2017. ISSN 1662-5161. doi: 10.3389/fnhum.2017.00423. URL `https://www.frontiersin.org/article/10.3389/fnhum.2017.00423`.

[16] N. U. F. Dosenbach, B. Nardos, A. L. Cohen, D. A. Fair, J. D. Power, J. A. Church, S. M. Nelson, G. S. Wig, A. C. Vogel, C. N. Lessov-Schlaggar, K. A. Barnes, J. W. Dubis, E. Feczko, R. S. Coalson, J. R. Pruett, D. M. Barch, S. E. Petersen, and B. L. Schlaggar. Prediction of individual brain maturity using fmri. *Science*, 329(5997):1358–1361, 2010. ISSN 0036-8075. doi: 10.1126/science.1194144. URL `https://science.sciencemag.org/content/329/5997/1358`.

[17] J. Frost. *Regression Analysis: An Intuitive Guide for Using and Interpreting Linear Models*. James D. Frost, 2020. ISBN 9781735431185. URL `https://books.google.es/books?id=1UPzzQEACAAJ`.

[18] L. Hansen, E. Lee, K. Hestir, L. Williams, and D. Farrelly. Controlling feature selection in random forests of decision trees using a genetic algorithm: Classification of class i mhc peptides. *Combinatorial chemistry high throughput screening*, 12:514–9, 07 2009. doi: 10.2174/138620709788488984.

[19] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001. doi: 10.1162/106365601750190398.

[20] N. J. Horton and K. P. Kleinman. Much ado about nothing: A comparison of missing data methods and software to fit incomplete data regression models, 2007.

[21] A. A. Imran, A. Rahman, H. Kabir, and M. S. Rahim. The impact of feature selection techniques on the performance of predicting parkinson's disease. *International Journal of Information Technology and Computer Science*, 10:14–29, 11 2018. doi: 10.5815/ijitcs.2018.11.02.

[22] K. Jamieson and A. Talwalkar. Non-stochastic best arm identification and hyperparameter optimization, 2015.

[23] S. Janitza, E. Celik, and A.-L. Boulesteix. A computationally fast variable importance test for random forests for high-dimensional data. *Advances in Data Analysis and Classification*, 2016. doi: 10.1007/s11634-016-0276-4.

[24] J. John Deegan. On the occurrence of standardized regression coefficients greater than one. *Educational and Psychological Measurement*, 38(4):873–888, 1978. doi: 10.1177/001316447803800404.

[25] P. A. Johnson, B. Rouet-Leduc, L. J. Pyrak-Nolte, G. C. Beroza, C. J. Marone, C. Hulbert, A. Howard, P. Singer, D. Gordeev, D. Karaflos, C. J. Levinson, P. Pfeiffer, K. M. Puk, and W. Reade. Laboratory earthquake forecasting: A machine learning competition. *Proceedings of the National Academy of Sciences*, 118(5), 2021. doi: 10.1073/pnas.2011362118.

[26] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *NIPS*, 2017.

[27] M. Koch, V. Geraedts, H. Wang, M. Tannemaat, and T. Bäck. Automated machine learning for eeg-based classification of parkinson's disease patients. 11 2019. doi: 10.1109/BigData47090.2019.9006599.

[28] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. 14, 03 2001.

[29] G. Kumar and P. Bhatia. A detailed review of feature extraction in image processing systems. 02 2014. doi: 10.1109/ACCT.2014.74.

[30] A. Liaw and M. Wiener. Classification and Regression by randomForest. *R News*, (3): 18–22, 2002.

[31] G. Louppe, L. Wehenkel, A. Sutera, and P. Geurts. Understanding variable importances in forests of randomized trees. NIPS'13, page 431–439, Red Hook, NY, USA, 2013. Curran Associates Inc.

[32] F. Maleki, N. Muthukrishnan, K. Ovens, C. Reinhold, and R. Forghani. Machine learning algorithm validation: From essentials to advanced applications and implications for regulatory certification and deployment. *Neuroimaging clinics of North America*, 30(4): 433—445, November 2020. ISSN 1052-5149. doi: 10.1016/j.nic.2020.08.004. URL https://doi.org/10.1016/j.nic.2020.08.004.

[33] C. Manzoni, D. Kia, J. Vandrovcova, J. Hardy, N. Wood, P. Lewis, and R. Ferrari. Genome, transcriptome and proteome: The rise of omics data and their integration in biomedical sciences. *Briefings in bioinformatics*, 19, 11 2016. doi: 10.1093/bib/bbw114.

[34] T. E. Mathew. A logistic regression with recursive feature elimination model for breast cancer diagnosis. 2019.

[35] A. Mignan and M. Broccardo. Neural network applications in earthquake prediction (1994–2019): Meta-analytic and statistical insights on their limitations. *Seismological Research Letters*, (4):2330–2342, May 2020. doi: 10.1785/0220200021.

[36] F. Nargesian, H. Samulowitz, U. Khurana, E. Khalil, and D. Turaga. Learning feature engineering for classification. pages 2529–2535, 08 2017. doi: 10.24963/ijcai.2017/352.

[37] E. I. Obilor and E. Amadi. Test for significance of pearson's correlation coefficient. 03 2018.

[38] H. Patel and P. Prajapati. Study and analysis of decision tree based classification algorithms. *International Journal of Computer Sciences and Engineering*, 6:74–78, 10 2018. doi: 10.26438/ijcse/v6i10.7478.

[39] P. Probst, A.-L. Boulesteix, and M. Wright. Hyperparameters and tuning strategies for random forest. 04 2018.

[40] J. R. Quinlan. Induction of decision trees. *MACH. LEARN*, 1:81–106, 1986.

[41] P. Roßbach. Neural networks vs. random forests – does it always have to be deep learning? 2018.

[42] B. Rouet-Leduc, C. Hulbert, N. Lubbers, K. Barros, C. Humphreys, and P. Johnson. Machine learning predicts laboratory earthquakes. *Geophysical Research Letters*, 44, 02 2017. doi: 10.1002/2017gl074677.

[43] L. A. Selvikvag A. An overview of deep learning in medical imaging focusing on mri. *Zeitschrift für Medizinische Physik*, 29(2):102–127, 2019. ISSN 0939-3889. doi: https://doi.org/10.1016/j.zemedi.2018.11.002. Special Issue: Deep Learning in Medical Physics.

[44] B. Skiera and J. Reiner. *Regression Analysis*. 09 2018. doi: 10.1007/978-3-319-05542-8_17-1.

[45] D. Steinkraus, I. Buck, and P. Simard. Using gpus for machine learning algorithms. pages 1115 – 1120 Vol. 2, 10 2005. doi: 10.1109/ICDAR.2005.251.

[46] C. Strobl, J. Malley, and G. Tutz. An introduction to recursive partitioning: Rationale, application, and characteristics of classification and regression trees, bagging, and random forests. *Psychological methods*, 14:323–48, 12 2009. doi: 10.1037/a0016973.

[47] Sullivan, White, a. P. R. Salter, and Lee. Should multiple imputation be the method of choice for handling missing data in randomized trials? *Statistical Methods in Medical Research*, 27:2610–2626, 2018. doi: 10.1177/0962280216683570.

[48] L. M. Sullivan. Correlation and linear regression. *Boston University School of Public Health*, 08 2019. doi: 10.1002/wmh3.347.

[49] F. R. Sullivan GM. Using effect size-or why the p value is not enough, 2012.

[50] S. Szymczak, E. Holzinger, A. Dasgupta, J. Malley, A. Molloy, J. Mills, L. Brody, and J. Bailey-Wilson. r2vim: A new variable selection method for random forests in genome-wide association studies. *BioData Mining*, 2016. doi: 10.1186/s13040-016-0087-3.

[51] M. Tsipouras. Spectral information of eeg signals with respect to epilepsy classification. *EURASIP Journal on Advances in Signal Processing*, 2019, 02 2019. doi: 10.1186/s13634-019-0606-8.

[52] G. Wang, J. Hao, J. Ma, and H. Jiang. A comparative assessment of ensemble learning for credit scoring. 04 2019.

[53] J. Wu, X.-Y. Chen, H. Zhang, L.-D. Xiong, H. Lei, and S.-H. Deng. Hyperparameter optimization for machine learning models based on bayesian optimization. *Journal of Electronic Science and Technology*, 17:26–40, 03 2019. doi: 10.11989/JEST.1674-862X.80904120.

[54] A. Zheng and A. Casari. *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists*. O'Reilly Media, Inc., 1st edition, 2018. ISBN 1491953241.

[55] L. Zhou and H. Wang. Loan default prediction on large imbalanced data using random forests. *TELKOMNIKA Indonesian Journal of Electrical Engineering*, 10:1519–1525, 10 2012. doi: 10.11591/telkomnika.v10i6.1323.