Prediction Model for Microwave Radio Unit Testing

Master's thesis in Embedded Electronic System Design

Sivasenapathi Balasubramaniam

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2019

MASTER'S THESIS 2019

Prediction Model for Microwave Radio Unit Testing

SIVASENAPATHI BALASUBRAMANIAM

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2019

Prediction Model for Microwave Radio Unit Testing

SIVASENAPATHI BALASUBRAMANIAM

Supervisor: Hugo Pinto, Ericsson AB, Microwave PIM, Boras, SE
Advisor : Patrik Olesen, Ericsson AB, Microwave Transport SW, Lindholmen, SE
Supervisor: Lars Svensson, Department of Computer Engineering, Embedded Electronic system design, Chalmers university of technology
Examiner: Per Larsson-Edefors, Department of Computer Engineering, Embedded Electronic system design, Chalmers university of technology

Master's Thesis 2019
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: A Picture showing Ericsson's Radio Unit mounted on site.

Typeset in LaTeX
Gothenburg, Sweden 2019

Prediction model for Microwave Radio Unit Testing
SIVASENAPATHI BALASUBRAMANIAM
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

During radio unit testing, the radio unit is tested in different test flows in order to ensure its conformance. But testing radio unit is time consuming and increases the unit's production time. Prediction models are developed for the test flows to predict test points continuously by using the inputs from the previous test flows. Machine learning models presented in this thesis helps in the prediction of test point values before placing the radio unit in all testing stations. The developed prediction model helps in finding the measured value and their dependencies for failure. The model presented in this thesis helps to predict the future test point value and reduces the unit time to market. In addition, to obtain a more intuitive insight Graphical User Interface is built up to access the prediction model.

Keywords: Prediction model, Regressor, Radio unit, test flows, test points, Data, Features, Model, Modelling.

Acknowledgements

First of all I would like to thank God without whom nothing is possible.

I would like to express my special appreciation and thanks to my supervisor Hugo Pinto, you have been a tremendous mentor for me. I would like to thank you for encouraging my research and for allowing me to take independent decisions during the thesis work. Your advice on both thesis as well as on my career have been invaluable.

I would especially like to thank the senior developers Per Wenander, Åke Söderström for providing extended support during the thesis period.

I would also like to thank Patrik Olesen, for his excellent contribution in the field of data analytics.

I like to thank my manager Peter Hansson, who is the key driver for the whole work and for his meaningful assistance and tireless guidance.

I like to thank my academic supervisor Lars Svensson for his help, language support and feedback on the thesis and I finally thank our fellow students Prathamesh Moralwar and Kenneth Peter for acting as opponents for my thesis .

Sivasenapathi Balasubramaniam, Gothenburg, 2019

Contents

List of Figures

1 Introduction

In the recent years, data traffic has been increasing rapidly and solutions are developed to meet the requirements of speed, power and cost [1]. Ericsson is one among the leaders in providing solutions for growing traffic demands [2]. Ericsson's microwave solution MINI-LINK is an end to end microwave link which provides communication using radio waves in the microwave frequency [3].

Production test is an important part of system manufacture. Testing is done in MINI-LINK products to ensure proper functioning in real time condition with the conformance to standards. A set of test conditions are programmed to test the performance aspects of MINI-LINK.

Based on the equipment used, the test flow for testing the MINI-LINK differs. Test flow is a series of tests that are carried out to ensure that the product conforms to specification, but hardware parameters are dependent of test flow. Test faults are identified when the product is tested under test flows. In order to forecast faults *before* testing, predictive analytics can be applied with real test conditions to anticipate failures at an early stage [4].

The primary scope of the thesis is applying predictive analytics to the MINI-LINK test flows. Predictive analytics are related with techniques from machine learning and data mining [4]. Machine learning is a subfield of artificial intelligence which involves the use of statistical methods to build up predictive models. Predictive models utilize statistics to analyse complex patterns in past data and predicts the future. Data mining is used to extract the knowledge from data sets by finding relationships to solve the problems.

Machine learning is about using computers not only for calculations and data retrieval, but combining those two capabilities of a computer system to make it appear to be learning and making rational decisions according to previously observed circumstances and previous actions or reactions, and not only act according to a fixed plan.

There are numerous algorithms in machine learning, but the learning approach is always based on available data [5]. The supervised approach is one among these methods, in which there are many input data sets with the corresponding outputs, and the task of the algorithm is to predict the behaviour upon new inputs [6]. In this thesis work, the past data for the MINI-LINK test flows are available with outputs, hence supervised learning is a natural choice .

Use of other methods rather than machine learning might be an option, but the data-driven approach is efficient with machine learning [7], [8]. At present, there is no tailored approach for extracting features of hardware properties which help to predict the failure rate and this thesis will be a part in such development [9].

1.1 Problem Statement

Radio units are tested after production with different test flows. Test flow varies with respect to different hardware versions of MINI-LINK. The hardware variant used has the following test flows: Radio Modem Board testing, Radio Calibration testing, Temperature testing(TEMP), and Final testing(System). Test flows has many hardware parameters measured and measured values are stored in Ericsson's internal data server. To predict the radio unit's failure in test flows, a prediction model is developed in this thesis using machine learning. Hence identifying such kind of failures results in reduced lead time and reduced time to market.

1.2 Aim

The aim is to develop a prediction model for predicting the hardware test faults in MINI-LINK's 6363 test flows by analysing hardware parameters. It also helps in finding the hardware property that results in test point failure. Another aim is to identify which test flows can be replaced with the prediction model. Furthermore replacing a test flow with prediction algorithm will also reduce carbon foot print in terms of sustainability.

1.3 Limitation

Prediction models are developed only for specific variant and only critical hardware parameters are considered for designing prediction model. Not all the hardware parameters are included. Training data are obtained for specific product revision and the model developed does not support prediction for new product revisions.

1.4 Research Questions

The following are the research questions answered in this thesis.

- Which is the appropriate machine learning model to identify hardware test failures?
- What are the methods for developing prediction model for hardware failure?
- How is the prediction model evaluated ?

1.5 Related Works

The below recent findings are collected on focus for gaining tacit knowledge in the field of predictive analytics, but neither the learning techniques nor the analogy of learning approaches is exercised amply. The learning models and methods depends upon data. The data characteristics defines the methodology of machine learning models. They can have same methodology but the concepts depends strictly upon the data [10]. The data used in this thesis is unique to radio unit testing and no prediction models exists for this kind of application.

As to the discussions made above, predictive analytics is not entirely new, few works were already performed and some of them are addressed below. One of the fundamental predictive techniques is discussed by Jiexing Gu in his work “Dynamic meta-learning for failure prediction in large-scale systems: A case study”, which involves dynamic prediction of failures with continuous changing of training inputs and failure pattern changes according to the prediction accuracy [11]. In this thesis failure predictions are calculated on large scale systems using the fundamental techniques discussed by Jiexing Gu. Joseph F. Murray’s “Multiple instance framework” involves predicting failures in hard drives for specific test cases with approximations for rare events [12]. In this thesis, specific test cases are addressed for different test flows and they use the same kind of approximations as discussed by Joseph for rare occurrence. Seung-JunShin developed a “Predictive Analytics Model for Power Consumption in Manufacturing” which uses a supervised learning model to predict power consumption with dependent and independent factor [13]. This concept is used in this thesis for predicting the power consumption of radio unit.

2

Theory

2.1 Machine Learning

The concept of machine learning has evolved over the period of time. Alan Turing raised the question “Can a machine think?”. Instead of determining the actual meaning of the question, Turing suggested a gaming approach, where the human acts as judge to identify the difference in responses between another human and a computer that tries to act as human [14]. If the decider fails to see the difference between the computer and the human then machine wins the game. This test exists still now in schema of Loebner prize which is an yearly AI competition, awards top notched computer programs that looks like human conversations. This led to lot of discoveries in the field of artificial intelligence [15]. Machine learning is one among these.

Tom M.Mitchell, 1997 defined “A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if performance at tasks in T, as measured by P, improves with the experience E” where the importance is given to series of phenomenon that is defined in Machine learning [16].

2.2 Key Terminologies

2.2.1 Dataset

Dataset is an array representation of information from which the machine learning algorithm learns. The dataset usually consists of features and may or may not contain targets for predictions. Dataset are of two kinds, training dataset and testing dataset. Training dataset is one which is introduced to the model for learning. After completing this process, the model is tested with the testing dataset.

2.2.2 Instance

Instance usually refers to a data point in dataset. For example, an instance of value x^* has a target y_*.

2.2.3 Features

Features are usually the columns in dataset. They contain the metadata apart from having qualitative information. After selecting the important features, they are used to train the machine learning model.

2.2.4 Prediction

Prediction refers what the algorithm for-sees target value based on the input features given for learning. It is normally expressed as $\hat{f}(x)^i$.

2.2.5 Confusion Matrix

Confusion matrix usually describes the performance of machine learning classification model on set of data for which the target values are known. It is well explained with an example matrix shown in the figure 2.1.

n=165	**Predicted: NO**	**Predicted: YES**
Actual: NO	50	10
Actual: YES	5	100

Figure 2.1: Confusion Matrix

Figure 2.1 represents a total of 165 predictions, out of those classified into four different classes. They are true positive(actual yes and predicted yes), true negative(actual no and predicted no), false positive(actual no and predicted yes), false negative(actual yes and predicted no). From this terminology, there are some metrics that are computed. They are

- Accuracy: It determines the number of correct classifications by the model.
- Mis-classification: It determines the number of wrong classifications by the model.
- Precision: It determines the model's closeness of predicted value to the actual value.

2.2.6 Bias

The term bias was first presented by Tom Mitchell in 1980 in his paper titled, "The need for biases in learning generalizations" [17]. Having bias was about model offering significance to some of features so as to sum up better for the bigger dataset with different attributes. Bias in machine learning generalises model and make our model less prone to unnecessary feature change.

2.2.7 Variance

Variance, with regards to machine learning, is a kind of error that arises due to models prediction on small changes that the model has not experienced during training. High variance will make algorithm to learn the noises in the dataset. This is most generally called as model over-fitting. When talking about variance in machine learning, we additionally refer the bias as discussed in the section 2.2.6. High bias will make the algorithm to lose the relevant relationship between the input feature and the output. High bias in some case is referred as under-fitting. Relationship between bias and variance, they are usually seen as trade-off minimizing one would lead the other increasing. The most widely recognized factor that decides the bias/-variance of a model is its ability to do prediction(think about this as how complex the model may be). It can normally alleviated by introducing a regularization parameter, limiting the updates during training.

2.3 Algorithm Types

There are numerous algorithms in machine learning and the algorithm selection depends upon the the desired output. In the upcoming chapters,different algorithms are discussed and the one that fits the problem best is considered as the suitable one. Machine learning algorithms are classified into three main categories: supervised, semi-supervised and unsupervised learning [16].

2.3.1 Supervised Learning

Supervised learning is associated with labelled dataset. Labelled data set consists of input and desired output variable. Still for a better understanding, the learning is compared to that of student in classroom, under the supervision of tutor. In the supervised learning, the output target may be either qualitative or quantitative variable(target may be either in form of numerical values or labels with classes). In supervised learning, the learning is accomplished with the use of objective function called $f(x)$ from which the model output is predicted[16].

2.3.2 Semi-Supervised Learning

Semi-supervised learning resembles more like a supervised learning, where some of the training inputs have missing outputs.In this learning, the model learns the behaviour from labelled along with unlabelled data to predict the target. All the machine learning algorithms are non-probabilistic, it is better to use model which represents $p(Y|X)$ and $p(X)$. Here, $p(X)$ is weighted when compared to $p(Y)$, models which has correlated distribution, will have a chance for semi-supervised learning [18].

2.3.3 Unsupervised Learning

Unsupervised learning is associated with unlabelled dataset. Here the training data consists of only the input and not the desired output responses. It is also related to the classroom with students without the supervision of tutor. In this type of learning the algorithm finds own data pattern. Unsupervised learning algorithms helps in finding the hidden pattern that the dataset could accompany with outcomes [16].

2.4 Machine learning and knowledge discovery

2.4.1 Machine learning Regression Algorithms

Depending upon the type of learning methods, the field has wide range of learning algorithms. Kajaree introduced different learning algorithms based on the similarities [19]. I will discuss some algorithms that are compared and used in this thesis work.

2.4.2 Regression Algorithms

Regression algorithms are widely used in predictive analytics, which makes use of correlation between predictors and the target variable for prediction. Correlation is defined as the extent to which the variable varies in relation to one another. When there is a strong correlation between predictors and target, then the prediction model has many patterns that needs to be learned from the data. There are different algorithms in machine learning. The below section covers the list of regression models used in this thesis.

2.4.3 Linear Regression

Linear regression is a type of regression that is more suitable for linear models, which models the relationship between the predictors and target used in the data set. It uses the linear predictor function which makes use of coefficients and independent variables to predict the values of dependent variable. The linear regression is described in equation 2.1.

$$Y = a + bX \tag{2.1}$$

Where slope is b, a is intercept, Y is dependent, and X is explanatory variable.

2.4.4 Ridge Regression

Ridge regression is used when the dataset suffers from multi collinearity. Multi collinearity in statistics refers when a predictor variable in multi regression model can be computed in linear manner. Due to multi collinearity, large variance occurs in data set, ridge regressor adds bias to reduce the deviation of actual value as seen in the equation 2.2, bias b is added to reduce the deviation of predicted value.

$$\widehat{\beta}_{ridge} = \left\{ ||y - Xb||_2^2 + \lambda ||b||_2^2 \right\} \tag{2.2}$$

2.4.5 Decision Tree Regressor

Decision tree is tree based network, generally used for classification and regression problem. It uses decision rules based on the strong features that makes decision. For learning, the tree uses if-then-else rules. Complexity of model depends upon the depth of decision tree. The depth of trees must be chosen in the way that does not over fits the model. The main advantage of using decision tree regressor is that it creates a multi-way path for the feature to extract learning patterns for prediction. Tree depth helps in increasing the ability of models to respond to future data [20].

2.4.6 Gradient Boosted Tree Regressor

Gradient boosted tree regressor is used in case of models that have weaker predictions. It uses decision trees in order to boost the performance of models. It uses the principle of ensemble methods to boost the performance of models. Ensemble methods refers combining several techniques in prediction model to reduce the variance (referred as bagging) and biasing(referred as boosting) [20].

2.4.7 Extra Tree Regressor

Extra tree regressor is a ensemble based regressor in which an estimator is used to fit the trees in dataset. It averages the number of trees used in data set, which improves the prediction accuracy and avoids over fitting of data to model. The main difference of extra tree model to normal decision tree is that it uses random split method and chooses the best fitted features to predict the target. The randomization can be chosen by tuning according to the problem and evaluated accordingly. It has good computational efficiency rather than accuracy [21].

2.4.8 Kernel Approximation

Kernel approximation techniques consists of functions which estimates feature mapping with respect to kernels. It applies non linear transformation to the inputs, which serves as a basis for other regression and classification algorithms. Since, there are no existential methods to choose the kernels for modelling. The results can be compared between the kernels to have a better conclusion. Generally used kernels are Nyostream, Radial basis kernel, Chi-squared kernel(additive and skewed) [22].

2.4.9 Elastic Net Regressor

Elastic net regression is used when predictors are larger than than the observation. It introduces a grouping effect which determines the feasibility of correlated parameters in the model [23].

2.4.10 Random Forest Regressor

Random forest regressor is a type of ensemble technique, which uses decision trees, boosting and bagging methods. This method was created by Tin Kam in 1995. Random forest method builds multiple trees in random spaces of features in data. These randomly created trees in sub-spaces generalizes the model regression to improve performance. The models output depends on multiple decision trees rather than individual decision trees. Kam also proposed methods for inflating generalization accuracies [20].

3
Methods

The radio units contains distributed and integrated frequency components which facilitates the microwave communication. The radios are normally based on multi standard technology and operated in different cellular technologies. These radio units are installed close to antennas in cabinet or integrated with antennas.

3.1 Data Analysis

The hardware test data of radio unit is highly non-linear and complex, which requires proper understanding of data insights and theoretical knowledge. John W. Tukey wrote *Exploratory Data Analysis* (EDA), where he proposed the procedure for analysing the data in simple manner which includes many statistical relations derived from the data [24]. EDA helps in proper understanding of data insights. EDA is considered as the important part which results in discovering unknown knowledge from a database.

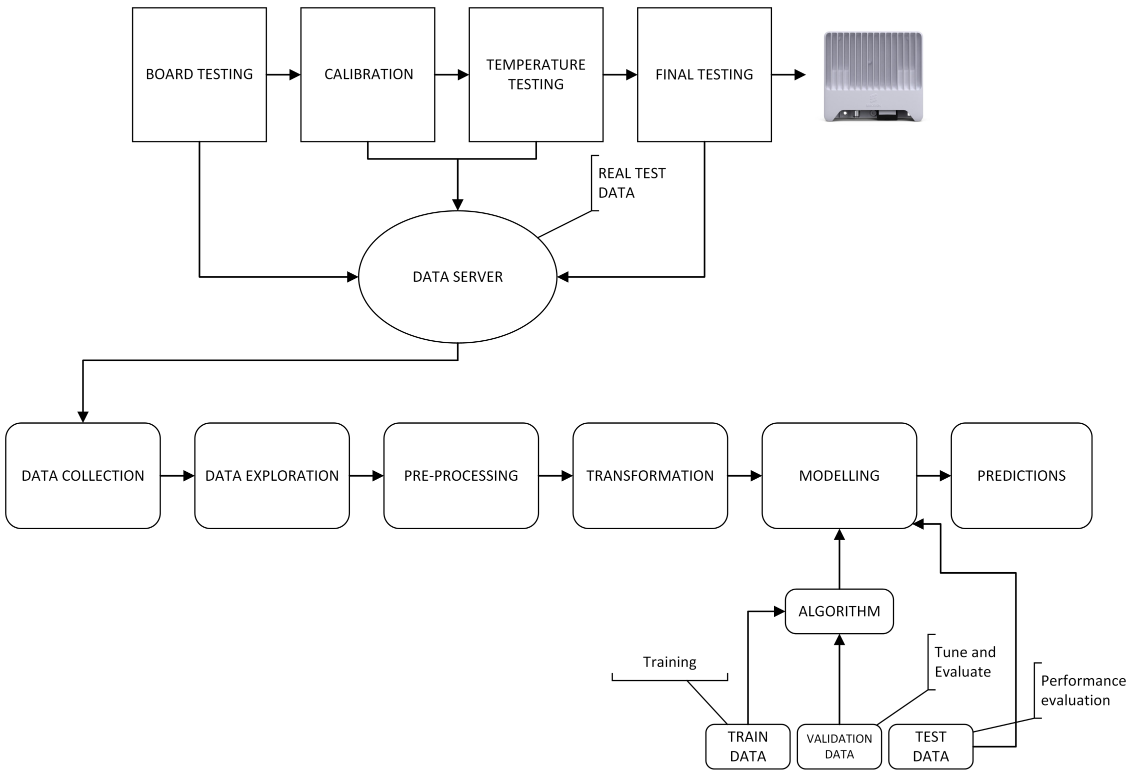

Figure 3.1: Radio unit fault prediction methodology

Radio unit included in this thesis consists of following test flows: radio board testing, calibration testing, temperature testing, final testing. When the radio unit is produced, it is tested under these flows in order to make sure of proper functioning. Each test flows include test points that need to be tested and the data from each test flow are stored in an online server. These data are used for the prediction model. The methodology is shown in figure 3.1.

It is important to analyze the data before modelling. EDA involves visualizing the outliers in the data, missing data, identify anomalies, and discover hidden patterns with numerical relationships.

3.2 Modelling Scenario

In this thesis, the machine learning model is built up for all the test flow predictions, such as predicting calibration test flow by using the data from board test, similarly temperature from board test and calibration, and final testing from board test, calibration and temperature testing. The prediction for final testing is further carried out in two stages; as discussed above it uses data from all three stations such as board test, calibration and temperature testing which is satisfactory in the way of machine learning perspective. But when it comes to technical perspective the data from calibration test has more relation to final testing, because the modem is calibrated with specific traffic bands and the same modem is tested in system testing with different traffic bands. Hence the data between calibration and system has dependencies. The above discussed scenario is explained in terms of flow chart in figure 3.2 .

3.3 Feature Selection

Feature selection is the most important process, where the useful features are selected from the dataset either manually or automatically [25]. Including unnecessary features will reduce the model training speed, predictability and model performance [26]. The benefit of performing feature selection will result in reducing the model sensitivity to noises and avoid the model using wrong data. In this work, the features are manually selected and are considered as the the critical points for each test stations and the critical test points are monitored for radio units after production. The features used for machine learning model are listed in table 3.1.

3.3.1 Data collection

The record of past test data were already available with Ericsson and are collected with respective data collection methods including IT infrastructure. The data used for analysis is obtained from real time online server, which keeps tracks of all test data for radio unit. This thesis utilizes only the production test data, and supervised methods are used with the test points to predict the values.

There are nearly three hundred test points in each test flows and the measured values are taken as features to respective test stations. Collecting larger amount

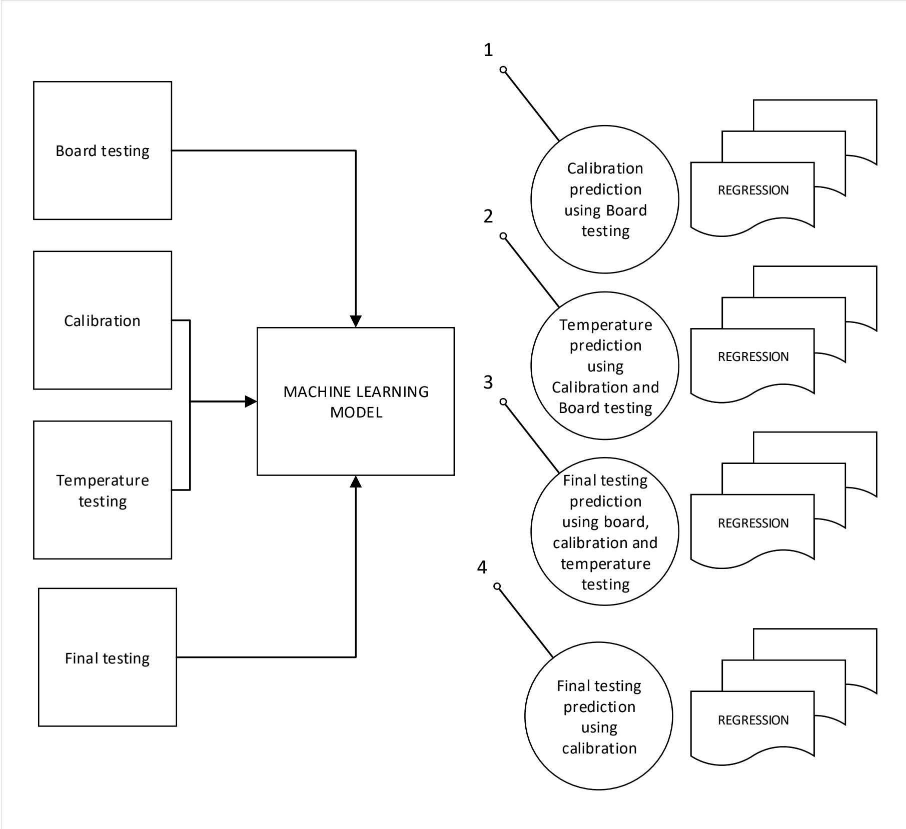

Figure 3.2: Modelling Scenario

of data results in building several common patterns, including wide spectrum of data distribution, but not all the features are considered important in prediction. Features that have less importance are removed with the help of feature reduction methodology.

Feature reduction methodology helps in reducing the number random variables for building machine learning model by obtaining the principal set of values needed for prediction. As the data dimension increases, it is hard to visualize data distributions and perform computations. Hence feature reduction methodology is used to remove redundant data dimensions. In this thesis, principal component analysis is used to remove the redundant data. Principal component analysis reduce large dataset to small dataset still holding most of the information in large dataset. Principal component analysis transforms number of correlated variables into a number of uncorrelated variables called as principal components [27].

Data collection depends upon the type of final prediction model. For example if it is a single class prediction model(predicting a single variable), or multi class prediction model(predicting multiple variables), etc,.. The data from the online server is collected as represented in raw format by building up queries. Applying the knowledge from domain experts, meta data are removed from the collected data. Hence only the informative features are considered.

Further collection proceeds in a way to extract information from the database. In this thesis, measured value of hardware parameter is considered as an object, where the test points are attributes to it. In this data collection, hardware parameters

Measure No	Board test	Calibration
1	Modem transmission level low 1	Transmission gain 1
2	Modem transmission level low 2	Transmission gain 2
3	Input output level set algorithm 1	Transmission gain 3
4	Input output level set algorithm 2	Transmitter Gain Calibration 1
5	Receiver control setting 1	Transmitter Gain Calibration 2
6	Receiver gain	Transmitter Gain Calibration 3
7	Receiver control setting 2	Transmitter Gain Calibration 4
8	Bandwidth gain	Transmitter power 1
9		Transmitter power 2
Measure No	**Temperature**	**Final**
1	Temperature extreme hot	Power amp high
2	Temperature extreme cold	Power amp low
3	Localized temperature 1	Power amp mid
4	Localized temperature 2	Amplitude Modulation(AM) coefficient
5	Localized temperature 3	Phase Modulation(PM) coefficient
6	Localized temperature 4	Mod volt
7	Localized temperature 5	Signal strength 1
8	Localized temperature 6	Signal strength 2
9	Localized temperature 7	Signal strength 3
10	Localized temperature 8	

Table 3.1: Features

measured are considered as attributes for the test flow(objects). The attributes consists of other metadata information which are not included in data collection as discussed above. Hence all the attributes with the listed objects are considered as features, usually represented in the fixed length of vectors used for machine learning.

3.3.2 Data Pre-processing and Transformation

The data included in the analysis is processed using the preliminary feature selection based on the expert's opinion, followed by outlier detection and inputing the missing values for the obtained data set.

The preliminary features are selected based on the experts opinion and the raw data distributions are viewed with normal distribution respectively. The figure below shows the normal distribution of raw data. The raw data for radio board test is collected and the distribution plot is plotted with kernel smoothing technique [28]. This technique is widely used in statistics and machine learning, which computes the real value function as the weighted average of the whole data. Hence this technique is adopted to get the actual distribution of raw data.

The distribution plot for radio board test is shown in figure 3.3 for understanding. For the rest of test flows, the plots are added in the appendix section. As you see in figure 3.3, the data are distributed in different scales in X axis. Z-score normalization is used which performs a linear transformation without spoiling the original data. Hence the information is retained by changing only the original data

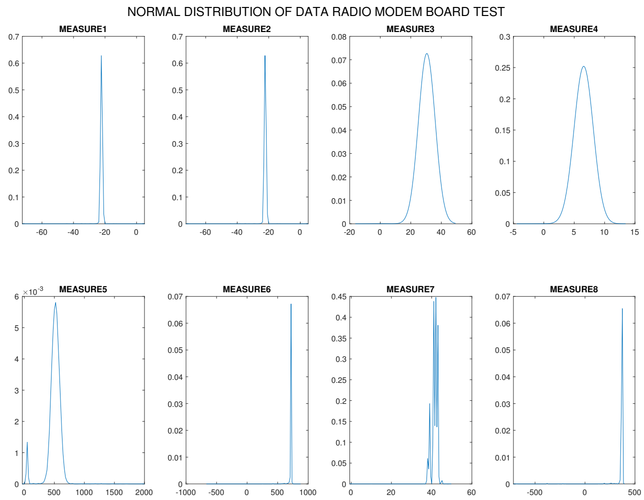

Figure 3.3: Normal distribution plot for radio board test

scale [29]. The main reason for normalization is to have a common scale for data.

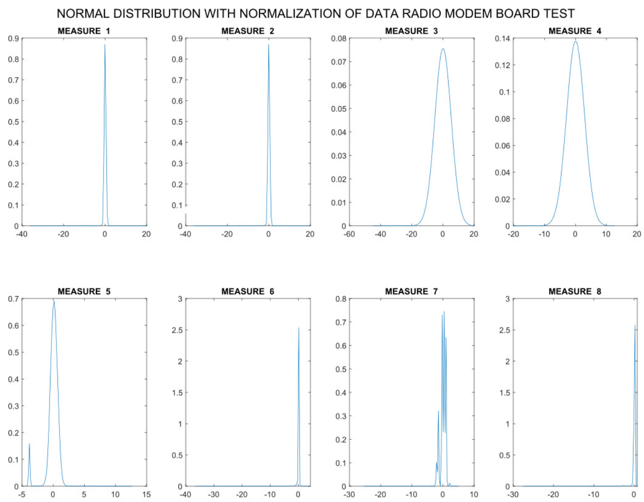

Figure 3.4: Normal distribution plot for radio board test with normalization

Z-score does this with the help of mean and standard deviation. Hence performing Z-score for the Radio board test data, will result in changing the scale to centred zero in x axis as seen in figure 3.4. Rest of the normalization results for other test flows are added in the appendix section. Detecting outliers in the data involves the use of different methodologies for detecting and removal. But this is one of the most critical part of data preprocessing [29]. Detecting the actual outliers and removing them from the existing data will result in enhancing the model. If not, outliers will ruin the model. The outlier here in the data may or may not contain useful information depending upon the model. Hence box plot is used to identify the outliers in the existing data. In this work, the outlier is detected and removed using isolation forest method [30] and the outputs before and after outliers are added for radio board testing station in figure 3.5 and the rest are added in the appendix section A.2

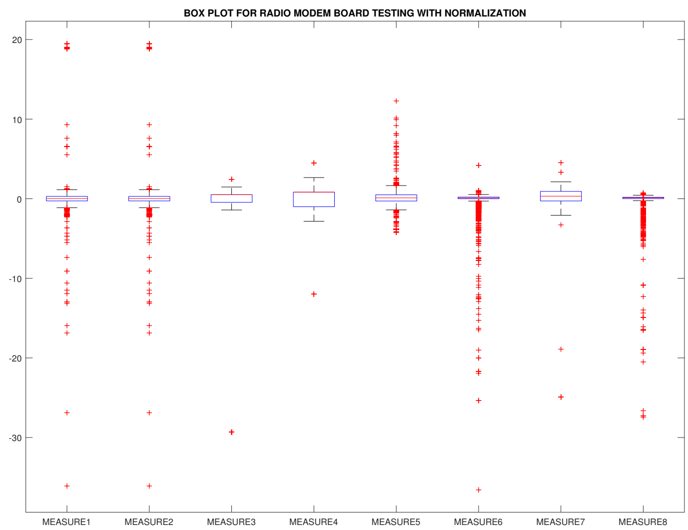

Figure 3.5: Box plot for outlier identification

In figure 3.5 the boxes refer to the probability distribution of data-sets with the centre median and the lines to top and bottom of boxes denotes the probability extent from median. The box plot is split up into three quantiles: first (Q1), second(Q2), and third (Q3). With the median line of distribution Q2, the box extends to right and left of normal distribution curves of data distribution. The data for the Radio board testing has lot of outliers in the raw data(markers in red color), it should be processed. The box has quantiles Q1 and Q3 from which the lower and upper fences for the dataset is calculated and the data lying outside the fences are considered as outliers. It is mathematically defined as follows [31].

$$\text{Lower fence} = Q_1 - 1.5(\text{IQR}) \tag{3.1}$$

$$\text{Upper fence} = Q_3 + 1.5(\text{IQR}) \tag{3.2}$$

In the above equation 3.1 and 3.2, IQR refers Inter-Quartile Range which is used as a measure of statistical data dispersion in mid spread region as seen in the figure 3.6. To extend these equations 3.1, 3.2 in plot with distribution, it is expressed in the figure 3.6.

These outliers might be errors in measuring equipment, human negligence, or system wear creep, or changes in system, or due to the normal distribution of data. According to the equations 3.1, 3.2 , the outliers are removed from dataset as shown in figure 3.7. Depending upon the prediction results, unsupervised outlier detection based algorithms are used.

The data obtained is not in linear representation, for example, decibel which is logarithmic value needs to be transformed into a linear value in terms of amplitude.

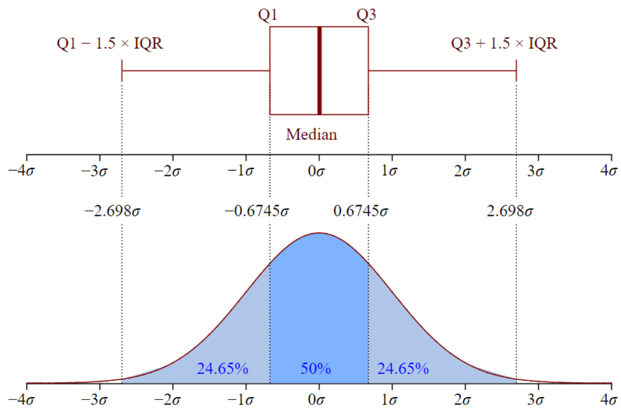

Figure 3.6: Box plot for outlier identification [31]

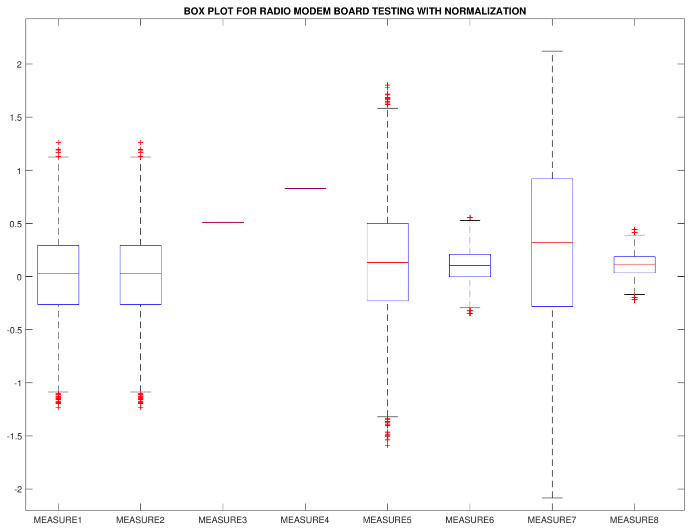

Figure 3.7: Box plot after outlier removal as same to dataset in figure 3.5

The features selected are based on the experts opinion for all the test flows initially, later to be replaced with Principal component analysis.

3.4 Modelling

Before modelling a machine learning algorithm, data visualization will help in better understanding of patterns and gives an outline of what type of algorithms can be used for prediction. Modelling algorithm requires the proper understanding of problem, having clear objective and desired outcomes.

3.4.1 Model selection and Evaluation

Machine learning models are selected based on model performance on input data. Normally, the test flow inputs(input data) collected from Ericsson's internal server after required preprocessing is fed into the machine learning model to obtain model predictions. The predicted data is compared against the real time data and the mean square error is computed to estimate model accuracy.

Explicitly, the above comparison doesn't estimate the model performance. In order to estimate the performance of model, new set of input data(unseen data from model) is given to the model to predict the performance. Hence different experiments are carried out with algorithm internal parameters(often termed as hyper-parameters) which are tuned later to improve model performance. The model selection using K-fold cross validation includes the following steps.

- The first step involves splitting of test data into testing data set and training data set. The testing data set is preserved for the model evaluation.

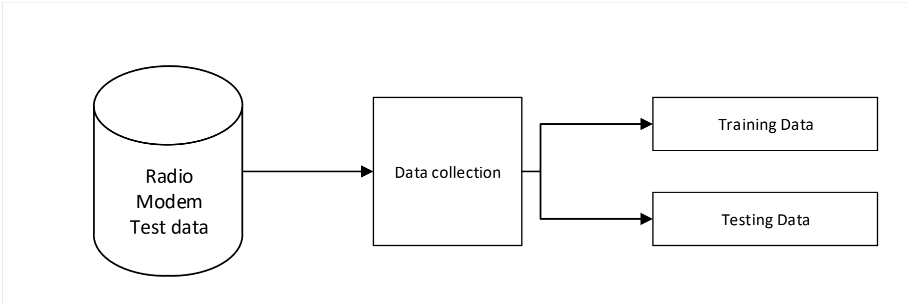

Figure 3.8: Model selection Step 1

- Second step involves in setting various hyper-parameters by using grid search or randomized search. For each hyper-parameter configuration K fold validation is applied to its training input. In this step, the configurations for hyper-parameters are selected using K fold. Similarly the hyper-parameters varies for different kinds of regression algorithms. Hence the flow chart for the second step is shown in figure for final testing using gradient boosted regressor.

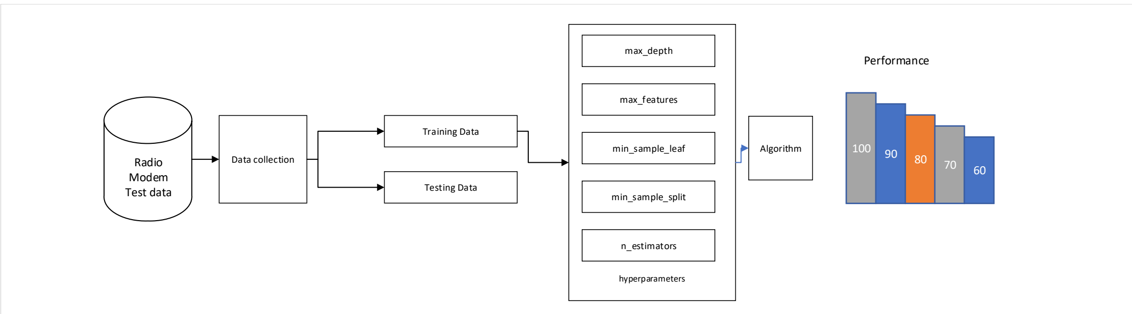

Figure 3.9: Model selection Step 2

- Third step is building the hyper-parameter with the best configuration obtained from the step2. The model is fitted with the training data.

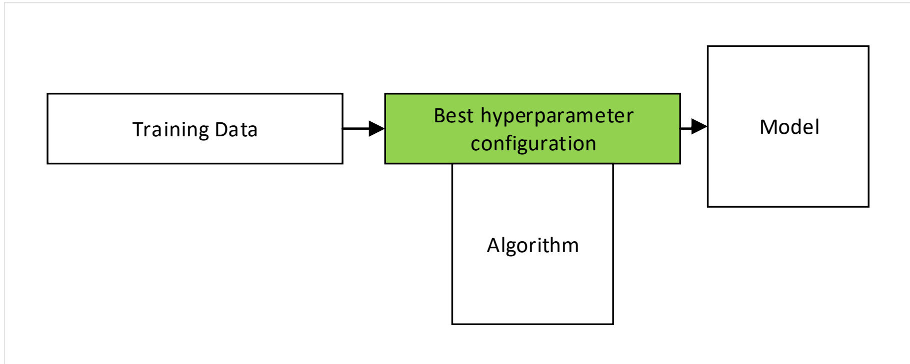

Figure 3.10: Model selection Step 3

- The fitted model is tested against the separated test data splitted from step 1. This test data is used to evaluate the fitted algorithm model with RMSE(Root mean square Error) metrics and validated. As per rule of thumb, if the model performances are better, then the algorithm uses informative features for prediction. Hence its obvious that the algorithm predicts using informative features.

3.5 Model optimization

The challenges faced during hyper-parameter optimization are

- Evaluating the optimization function is difficult because of large data set for radio units.
- Computing loss functions(such as mean square error to evaluate how well the algorithm models the training data) for test points is restricted because the target function suffers from mismatch of dataset convergence.
- Hyper-parameter configuration space is complex in case of these radio units.

3.5.1 Optimization using grid search

Selecting hyper-parameters for the selected machine learning algorithm by defining a grid of parameters. The values in grid are then set for all parameters. For example, while optimizing the hyper parameters for gradient boosting regressor, the hyper-parameter configurations for parameters are initialized in python code as shown in the figure 3.11.

3.6 Model performance stability

The base models are developed individually for each test point in final test station and the developed algorithm will find the patterns in the existing dataset. To find the hidden pattern, the algorithm should learn the features from training data set. If a model is trained with lot of input data, it suffers from over fit. The best method to prevent the model from over-fitting is to evaluate the model performance on the

```
In [27]: param_grid = {
             'max_depth': [1, 2, 3, 4],
             'max_features': [2, 3],
             'min_samples_leaf': [3, 4, 5],
             'min_samples_split': [8, 10, 12],
             'n_estimators': [100, 200, 300, 1000]
         }

         rf = GradientBoostingRegressor()
         grid_search = GridSearchCV(estimator = rf, param_grid = param_grid,
                                    cv = 5, n_jobs = -1, verbose = 2)
         grid_search.fit(x_train1, y_train1)

         Fitting 5 folds for each of 288 candidates, totalling 1440 fits

         [Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.
         [Parallel(n_jobs=-1)]: Done  25 tasks      | elapsed:   14.0s
         [Parallel(n_jobs=-1)]: Done 384 tasks      | elapsed:   18.7s
         [Parallel(n_jobs=-1)]: Done 1196 tasks      | elapsed:   33.6s
         [Parallel(n_jobs=-1)]: Done 1440 out of 1440 | elapsed:   37.9s finished

Out[27]: GridSearchCV(cv=5, error_score='raise-deprecating',
                estimator=GradientBoostingRegressor(alpha=0.9, criterion='friedman_mse', init=None,
                      learning_rate=0.1, loss='ls', max_depth=3, max_features=None,
                      max_leaf_nodes=None, min_impurity_decrease=0.0,
                      min_impurity_split=None, min_samples_leaf=1,
                      min_sampl...=None, subsample=1.0, tol=0.0001,
                      validation_fraction=0.1, verbose=0, warm_start=False),
                fit_params=None, iid='warn', n_jobs=-1,
                param_grid={'max_depth': [1, 2, 3, 4], 'max_features': [2, 3], 'min_samples_leaf': [3, 4, 5], 'min_samples_split': [8, 1
         0, 12], 'n_estimators': [100, 200, 300, 1000]},
                pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
                scoring=None, verbose=2)
```

Figure 3.11: Grid search cross validation using Python

data samples that have not been used in training. The use of K fold cross validation to check the model's performance for different data and hold out technique to test out of sample technique and ensure the model is not over-fitted [32]

The use of following method in base model insures that the over-fitting did not occur.

- Separating 20 percent of training data as holdout data, which is used to verify the model performance on data that has not been seen by the model during training.
- To evaluate the base model further, the data is divided into 5 cross folds(set of data split based on value of K) in partitions and the model is trained for a smaller part of data and a fold of data is used to evaluate the model performance. For the model which has highest performance, the sizes of folds can be increased and tested further with data folds and the mean value is calculated as shown below.

The data visualization in the figure 3.12 shows the data split for validation process. The one in blue with 80 percent is the data for training and rest is divided K- folds for cross validation. The data coloured with red in figure 3.12 indicates the hold out sample.

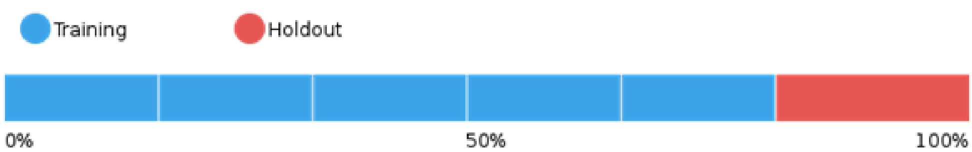

Figure 3.12: Data Split up

3.7 Interpretation

In many applications it is useful to interpret the derived approximation $F(x)$. This involves gaining an understanding of those particular input variables that are most influential in contributing to variation, and the nature of dependence of $F(x)$ on those influential inputs. To the extent of $F(x)$ at least qualitatively reflects the nature of target function $F^*(x)$, such tools can provide information concerning the underlying relationship between the inputs and the output variable. In this section, several tools are presented for interpreting the tree boosting approximations. The result of interpretation is shown in figure 4.13.

3.7.1 Relative importance of input variable

Among the most useful descriptions of an approximation $F(x)$ are the relative influences of individual inputs, on variation of $F(x)$ over joint input variable distributions. Piece-wise constant approximations are produced by decision trees and it can be approximated by a surrogate measure that reflects its properties. Breiman, Friedman, Olshen and Stone proposed function to over come the dependencies [20].

3.7.2 Partial dependence plots

Visualization is one of the most powerful interpretational tools. Graphical renderings of the value of $F(x)$ as a function of its arguments provides a comprehensive summary of its dependence on the joint values of the input variables. Unfortunately, such visualization is limited to low-dimensional arguments. Functions of a single real-valued variable $F(x)$ can be plotted as a graph of the values of $F(x)$ against each corresponding value of X.

Functions of a single categorical variable can be represented by a bar plot, each bar representing one of its values, and the bar height the value of the function. Functions of two real-valued variables can be pictured using contour or perspective mesh plots. Functions of a categorical variable and another variable(real or categorical) are best summarized by a sequence of trellis plots, each one showing the dependence of $F(x)$ on the second variable, conditioned on the respective values of the first variable [33].

Viewing functions of higher-dimensional arguments is more difficult. It is therefore useful to be able to view the partial dependence of the approximation $F(x)$ on selected small subsets of the input variables. Although a collection of such plots can seldom provide a comprehensive depiction of the approximation, it can often produce helpful clues, especially when $F(x)$ is dominated by low order interactions.

4
Results

There are four different modelling scenarios and the results for the test point prediction are added and the results for algorithm selection and evaluation are added and discussed below.

- Scenario 1: Calibration Prediction from Board testing
- Scenario 2: Temperature prediction from Calibration and Board testing
- Scenario 3: Final prediction from temperature, Calibration and Board testing
- Scenario 4: Final prediction from Calibration testing.

4.1 Modelling Scenario 1

In this scenario, Calibration test flow prediction is done by using the data from Board testing. There are different models built up for different test points in Calibration testing. The model selection for a test point in Calibration testing is illustrated from the figure 4.1.

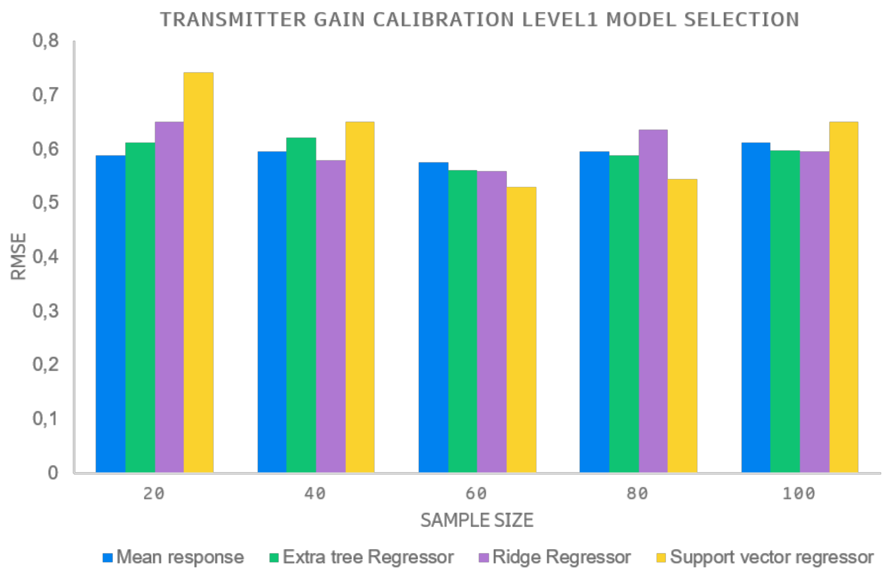

Figure 4.1: Model selection for Calibration testing

As discussed in the previous section 3.4.1, the machine learning models are selected using K fold cross validation, and the performance metric used is RMSE(Root Mean Square Error). From figure 4.1, Support vector regressor with sample size(split size for validation data in model selection process) of 60 produces lesser error when compared to mean response regressor, extra tree regressor and ridge regressor. From figure 4.1, the support vector regressor should be used to built up the prediction model for predicting test point transmitter gain Calibration level 1 as it has lesser RMSE. The training data splitted from k fold is then allowed to fit the model. Then the model is allowed to predict the values for test data. The prediction plot is shown in the figure 4.2.

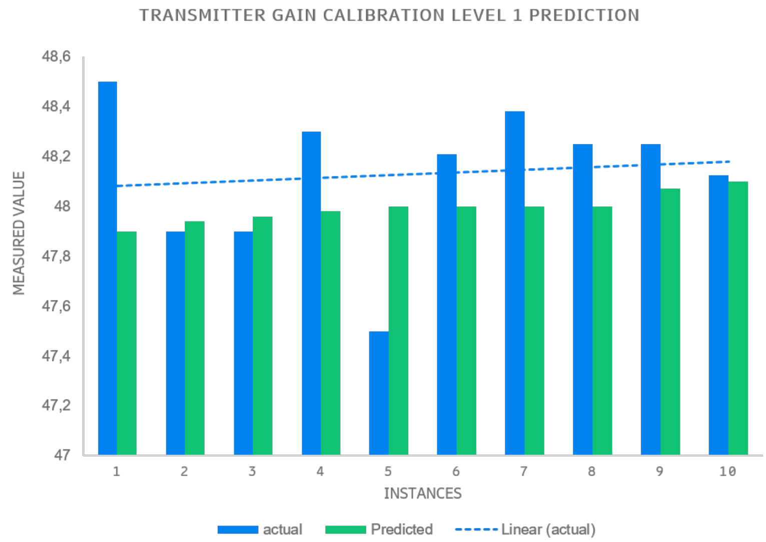

Figure 4.2: Transmitter gain calibration level 1 prediction

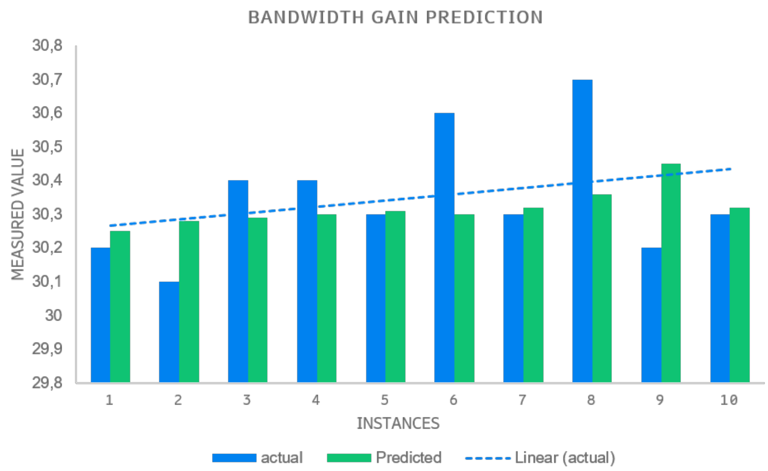

Figure 4.3: Bandwidth gain prediction

The linear value scale in the figure shows the prediction trend for all the instances used in prediction. The predicted values can be seen in green colour in figure 4.2. The linear trend keeps on increasing for the actual value and the predicted value trend also increases for the period of instances as seen in the figure 4.2. Comparing the data from board testing and calibration in terms of correlation, there exists a poor correlation between the test flow data. The correlation plot is added in the appendix section for reference. For example, while predicting Input output level set algorithm 1, the predicted output is shown in the figure 4.3.

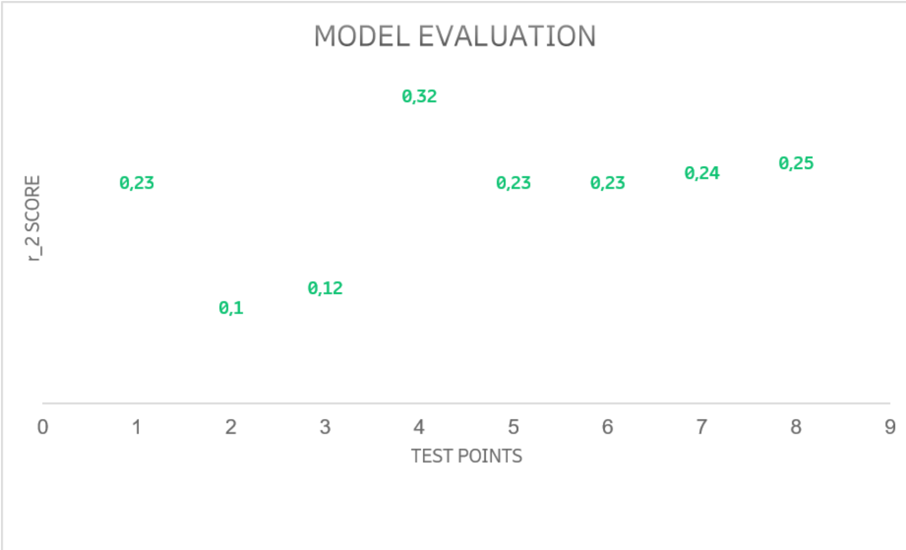

Figure 4.4: Model Evaluation

Similarly, the model is built up for the rest of test points for calibration prediction. To conclude with the modelling from scenario 1, the algorithm evaluation is done with the help of $r2\_score$ metrics and the results are plotted in the figure 4.4.

The models used for prediction doesn't have a good $r2\_score$ as seen from figure 4.4. A score with value closer to one denotes that the input features are cooperating for prediction and if a value that's lesser than 0.5, then the input features are not relevant for prediction. Hence its clear that the input from Board testing is not relevant in calibration prediction.

4.2 Modelling Scenario 2

As the scenario 1 produces disappointing results from evaluation, scenario 2 is used in case to predict the temperature test parameters by using the input features from board testing and calibration. As discussed from the above scenario, the models are built up for the different test points and the models are selected by using k fold cross validation as discussed in the previous section 3.4.1. The prediction results for extreme hot and cold are plotted in the figure 4.5 and 4.6.

Similarly the rest of test points are predicted from temperature testing. From figure 4.5, the prediction input tracks the changes in the input from the previous stations and it finds the relationship in predictions. But in technical perspective, there is no such kind of direct relationship for temperature prediction from board and calibration inputs. So there is mismatch in terms of data perspective, where the algorithm finds the pattern for hot temperature prediction. But when you see the figure 4.6, the technical perspective correlates with the data perspective. There should be a glitch in the data or from measurement equipment, for having a strange result in hot prediction. As there was no correlation between the data in scenario

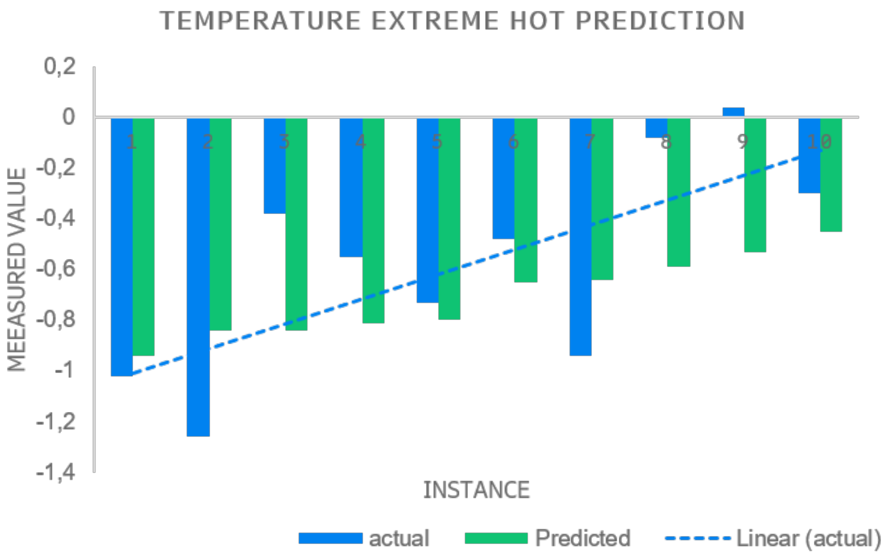

Figure 4.5: Temperature extreme hot prediction

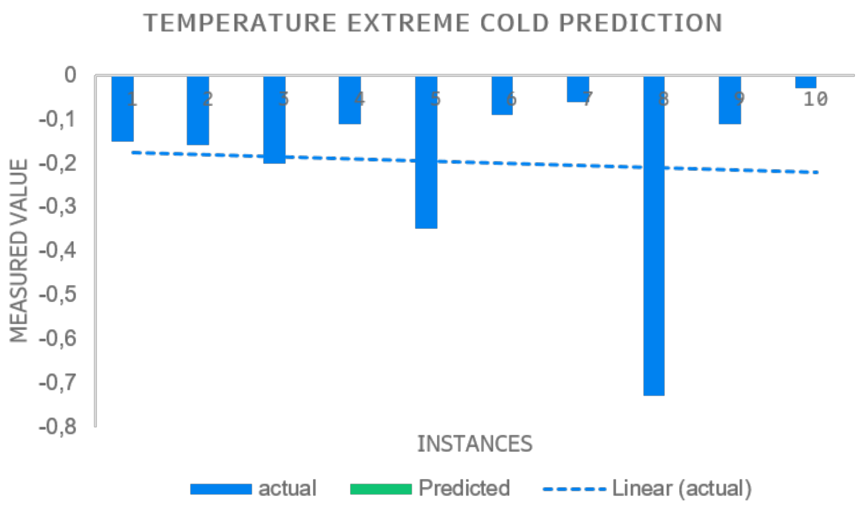

Figure 4.6: Temperature extreme cold prediction

1, it is the same as well in this scenario. The correlation plot is added in the appendix section for reference. Evaluation resulted in poor scores further making this prediction not practical in real-time. Hence temperature predictions are not possible with the data from board testing and calibration.

4.3 Modelling Scenario 3

Final prediction is modelled with the inputs from board testing, calibration testing and temperature testing. Hence the Final is modelled with the help of learning algorithms for different test points for gathering the most of information(informative features) from the inputs. An extensive model selection method is used to find out the appropriate models using k fold cross validation. The models selected for the test points are shown in figure 4.7.

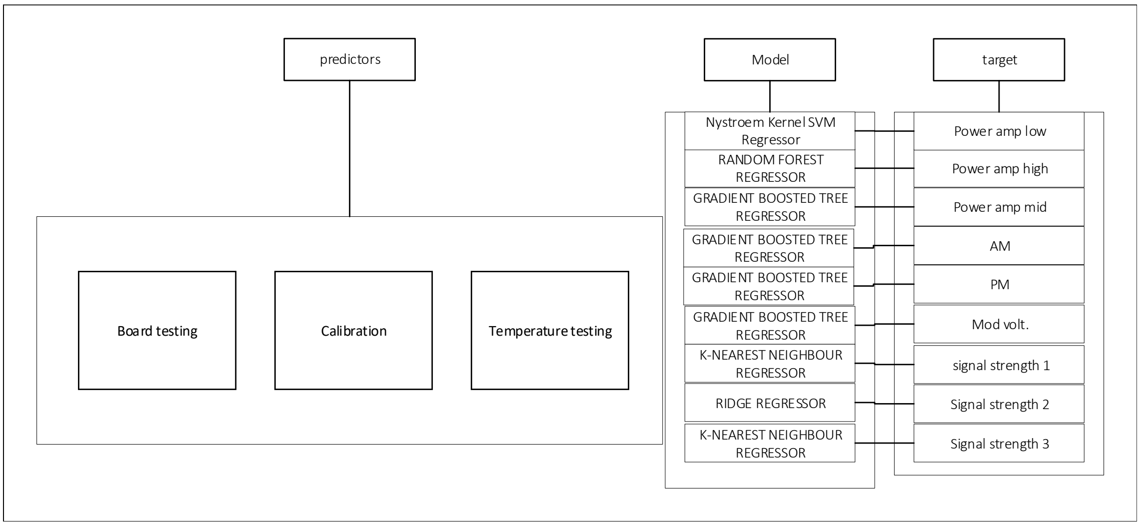

Figure 4.7: Models for Final testing scenario 3

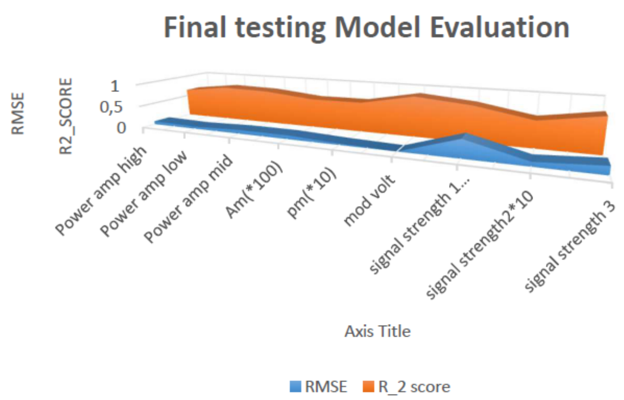

Figure 4.8: Model Evaluation for Final testing

The model is fitted against the testing inputs and the model is evaluated with RMSE and r2_score metrics in the figure 4.8. The fitted model has least RMSE and the model seems to have changes in relation to the changes from the input parameter. The predicted time series outputs are shown for consecutive ten instances in figure 4.9. The predictions are close to the actual value and the model is evaluated successfully. In this scenario, the model acquires input features from all the three stations, hence in-terms of training and testing, the time consumed are comparatively higher.

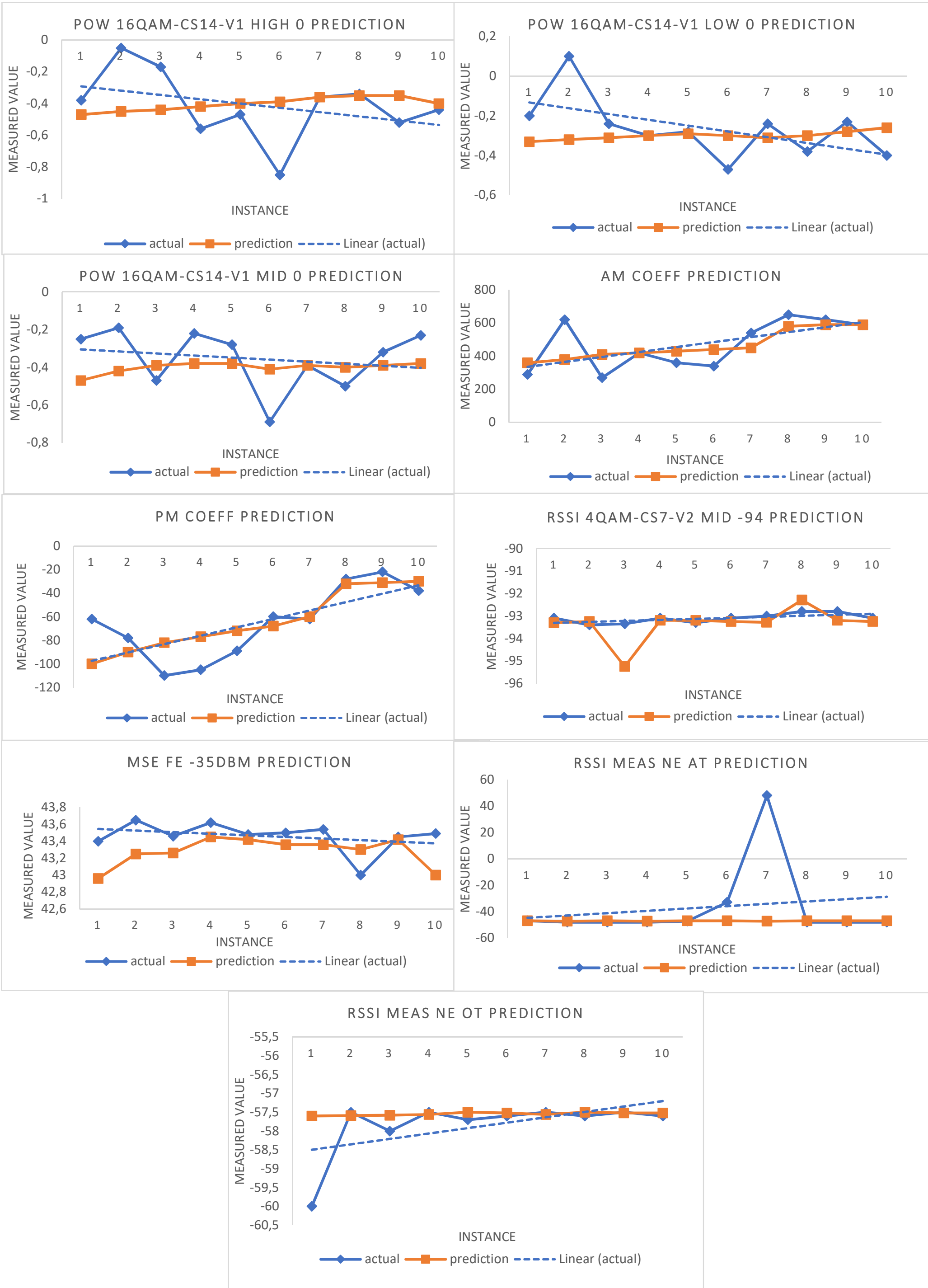

Figure 4.9: Model prediction for Final testing

4.4 Modelling Scenario 4

In this scenario, Gradient boosted regressor is used to predict all the test points in Final testing by using data from Calibration. An interesting relationship is found between the test parameters of Calibration and Final testing. Despite the use of different traffic bands between Calibration and Final, they have a strong correlation. Hence a model is built up by using only features from Calibration to test Final testing parameters. The model is trained and evaluated as in scenario 3 and the results are compared added in the figure 4.10.

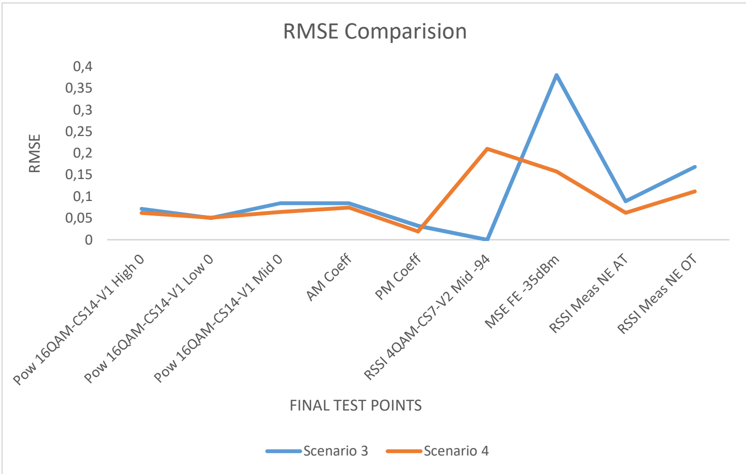

Figure 4.10: RMSE Comparison

From the figure 4.10, the RMSE in scenario 4 reduced significantly upto 10 points when compared to that in scenario 3. Hence the Calibration input features are more informative to the model prediction than the use of rest of three station inputs in previous scenario. The losses in this scenario might be of the fact that the test flow uses different traffic bands to test unit performance. Hence this input feature is more relevant in terms of Final test flow prediction, when compared to model using all three flows. The predicted output for Final testing with time series ten instances are shown in figure 4.11.

Figure 4.11: Model prediction for Final testing

4.4.1 Feature importance and Partial dependency

Feature importance for scenario 4, measure of POW 16QAM CS14 V1, works by altering input data and observing the effect on a models score. This technique is sometimes called Permutation Importance. The Feature Impact for a given column measures how much worse a models error score would be if predictions were computed after randomly shuffling that column (while leaving other columns unchanged). Model normalizes the scores so that the value of the most important feature column is first and the other subsequent features are normalized to it.

In the case of Gradient boosted regression, we can gain considerable insight into

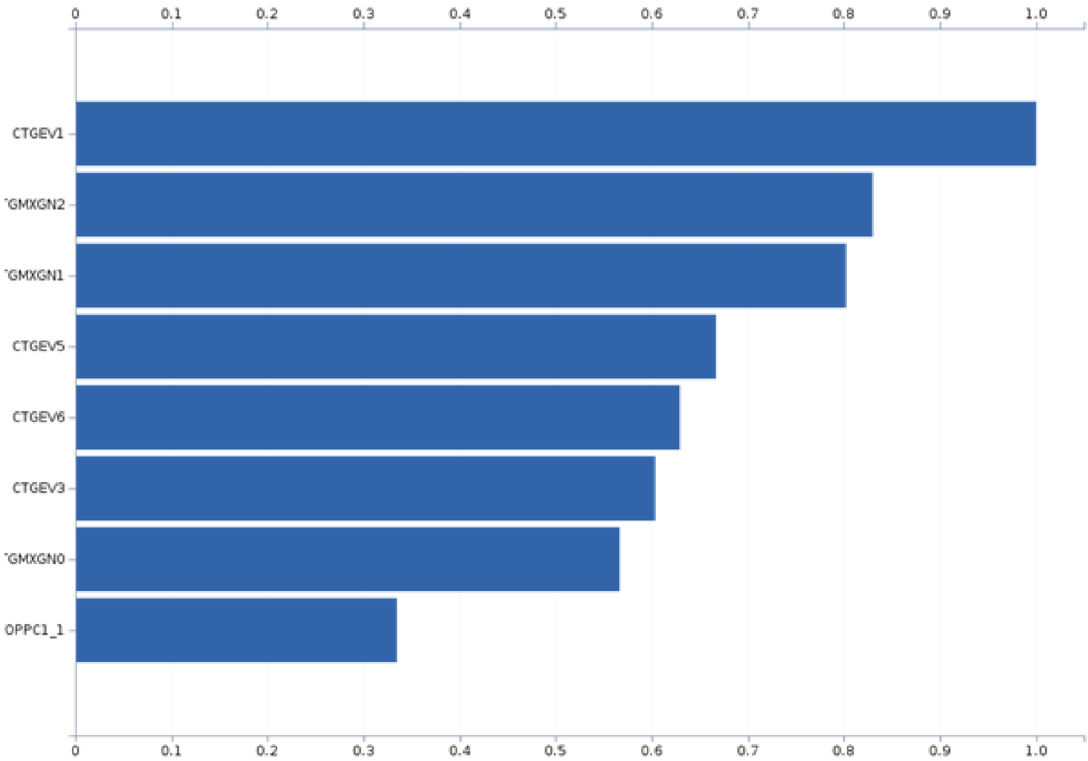

Figure 4.12: Feature importance of scenario 4 prediction- POW 16QAM CS14 V1 HIGH

the structure and interpretation of the model by examining its coefficients. For more complex models like support vector machines, random forests, or the blenders considered here, no comparably simple parametric description is available, making the interpretation of these models more difficult. To address this difficulty for his gradient boosting machine, Friedman proposed the use of partial dependence plots [34]. Partial dependence plots show the average partial relationship between a set of predictors and the predicted response. The partial dependence plots in the figure 4.12 capture the top features in our model, as measured by Feature Impact.

The orange circles depict, for the selected feature, the average target value for the aggregated feature values. The blue crosses depict, for the selected feature, the average prediction for a specific value. From the graph you can see that model also averages the predicted feature values. Comparing the actual and predicted points can identify segments where model predictions differ from observed data. This typically occurs when the segment size is small. In those cases, for example, some models may predict closer to the overall average.

The yellow partial dependence data points depict the marginal effect of a feature

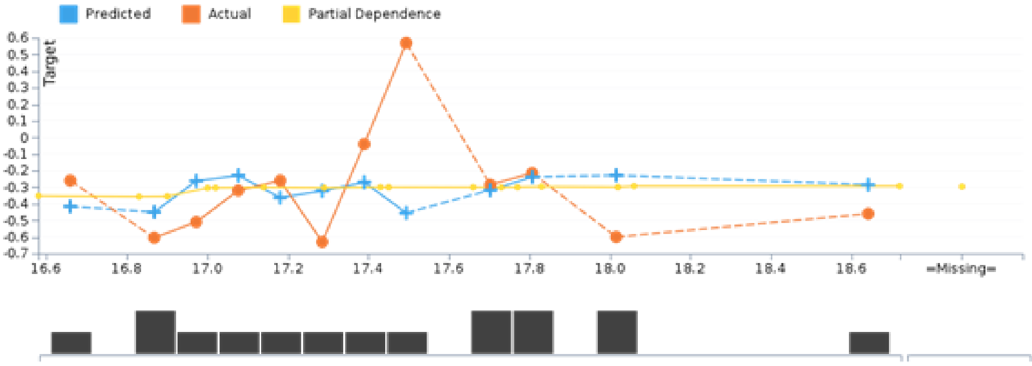

Figure 4.13: Partial dependency of CTGEV1 from Calibration to POW 16QAM CS14 V1 HIGH

on the target variable after accounting for the average effects of all other predictive features. It indicates how, holding all other variables constant, the value of this feature affects your prediction. Regression holds constant the values of all columns in the sample except the feature of interest. The value of the feature of interest is then reassigned to each possible value, calculating the average predictions for the sample at each setting. These values help determine how the value of each feature affects the target. The shape of the yellow data points describes the model's view of the marginal relationship between the selected feature and the target.

4.5 GUI

An interactive GUI platform has been built up to access the prediction model. GUI platform was built up using the the python package tkinter [35]. An example of GUI platform showing Final prediction using the data from Calibration is shown in the figure below.

Figure 4.14: GUI model

5
Conclusion

The success of machine learning in predicting test points relies on the good use of the data and machine learning algorithms. Selecting the right machine learning method for the right problem is necessary to achieve the best results. However, the algorithm alone cannot provide the best prediction results for Calibration prediction from Board testing, temperature prediction from Calibration and board testing and final prediction from temperature, calibration and board testing. Feature engineering, the process of modifying data for machine learning, is also an important factor in getting the best prediction results. This thesis helps in identifying which test flows in testing can be predicted with prediction model.

Hence different scenarios are discussed, in terms of their ability to improve the prediction results. Different radio unit's data sets were analyzed with different machine learning methods, and their results were compared using two evaluation measures. For the evaluation of feature engineering, machine learning methods were applied to the raw and modified versions of the data separately. The main method of feature engineering was feature selection (which is suggested in this thesis). In the case of classification and regression trees, additional feature engineering was done in the form of custom feature creation (such as adding distance to limits of measured variable as additional feature) which failed in prediction showing poor accuracy.

The accuracy reached in final prediction from calibration testing was higher than in final prediction from temperature, calibration and board testing. This can be attributed to the difference in dependent variables. As a result, generalizing the dependent variable made the predictions easier in this thesis which is well evident from the results of final prediction from calibration testing.

With the emergence of advanced data-driven approaches and technologies for collating the MINI-LINK data sets, the sophisticated machine learning models can be developed so as to alleviate the classical testing of units by replacing it with the machine learning models. With the help of data mining techniques, there is no need to define the functionality form of the model. This is a practical benefit particularly in the testing conditions where building an appropriate model is almost impossible. Furthermore, data-driven approaches give room for researchers to investigate the effect of new features for model prediction.

In this study, the Gradient Boosting of Regression Tree algorithm for scenario 4 showed excellent prediction and evaluation results when compared to the rest of all scenarios. These results support future replacement of some system testing with machine learning models.

This work answered all the research questions:

Which is the appropriate machine learning model to identify hardware test failures?
The appropriate machine learning models are selected as discussed in the methods section. Some problems are very specific and require a unique approach. E.g. if you look at a Scenario 4, Gradient boosting algorithm is used and it solves a very specific kind of problem.

Besides some of the decisions that we make when choosing a machine learning algorithm have less to do with the optimization or the technical aspects of the algorithm but more to do with final decisions.

What are the methods for developing prediction model for hardware failure?
While statistical tools might be the same, the study design aimed at causal explanation and theory building might be different from that aimed at prediction. This distinction in the goal of analysis may have impact on different steps of modelling process.

The model developed in this thesis elaborately described this difference on every step of prediction modeling process, i.e., goal definition, data collection, data preparation and analysis, choice of variables and methods, model selection and validation, and results. This conceptual difficulty might be minimized or avoided by careful selection of the studies to be used for prediction modeling(choosing the test points defines that). On the other hand, data-driven exploratory analyses even though not design for prediction might be more useful to be converted into prediction model. Therefore, for our purpose of developing prediction, algorithm for hardware test prediction the most adequate report is based on the study where multiple variables are included in the model and are reported in the thesis.

How is the prediction model evaluated?
Normally the model quality and performance should be assessed with the testing data and not with training data. Acting in a way that the training data does not over fits the model and not misleading in this project. Instead, this thesis uses the R2 score and RMSE, which readers are more likely to be able to use to understand whether the model is performing good or not. The value of R2 should be calculated for the modem test data and not from a regression predicted values. These are the methods discussed in this thesis to evaluate the machine learning models.

5.1 Recommendations

Recommendations include:

- Outstanding frameworks for replacing the machine learning model can be done with the help of data robot API(added in appendix section). This would allow the deployment of machine learning model in real time.

- Replacement of a physical test station in the MINI-LINK test will reduce the carbon foot print which have a positive effect on sustainability
- In terms of analysis, the analysis can also carried out for Temperature station by using the data only from calibration testing. I carried out this analysis, but without a time series data which is not good in terms of accuracy. But this can be tried out with time series data, in order to check the floor space of temperature predictions.
- Analysing data with different frequency band units which has more data concentration will help machine learning model to infer the patterns from data distribution.
- Data preparation is considered to be a time consuming task. Establish some efficient methods from data servers, so that the data collection and preparation can be made easy for test data in future.
- Studying different test points apart from the suggested one in this study and their dependencies with other test flows in terms of data, so that it joins hand in prediction.

Bibliography

[1] R. N. Clarke. (2014), Expanding mobile wireless capacity: The challenges presented by technology and economics, *Telecommunications Policy 38* ISSN 0308-5961.

[2] Ed.Gubbins. (2018), Microwave Back-haul:Competitive Landscape Assessment.

[3] MINI-LINK, *https://www.ericsson.com/en/portfolio/networks/ericsson-radio-system/mobile-transport/microwave*

[4] R.Sheposh. (2017), Predictive analytics, *Salem Press Encyclopedia of Science.*

[5] K.Lee. (2016), Machine Learning Approaches for Learning Analytics : Collaborative Filtering Or Regression With Experts ?, *Thirtieth Conference on Neural Information Processing Systems.*

[6] X. Xia, D. Lo, X. Wang, X. Yang, S. Li and J. Sun. (2013), A Comparative Study of Supervised Learning Algorithms for Re-opened Bug Prediction, *2013 17th European Conference on Software Maintenance and Reengineering*, ISSN-1534-5351, pages-331-334.

[7] M.Bauzá. (2018), A Data-Efficient Approach to Precise and Controlled Pushing, *CoRL.*

[8] M.Polese, R.Jana, V.Kounev, K.Zhang, S.Deb and M.Zorzi. (2018), Machine Learning at the Edge: A Data-Driven Architecture with Applications to 5G Cellular Networks, *arXiv:1808.07647.*

[9] E.Salahat and M.Qasaimeh. (2017), Recent Advances in Features Extraction and Description Algorithms:A Comprehensive Survey, *CoRR*, E-print 1703.06376.

[10] A.Agrawal and T.Menzies.(2018), “Is ‘Better Data’ Better Than ‘Better Data Miners’?” Proceedings of the *40th International Conference on Software Engineering-ICSE 18.*

[11] J. Gu and Z. Zheng and Z. Lan and J. White and E. Hocks and B. Park. (2008), Dynamic Meta-Learning for Failure Prediction in Large-Scale Systems: A Case Study, *37th International Conference on Parallel Processing*, ISSN-0190 3918, pages-157-164.

[12] J.F.Murray, G.F.Hughes, and K.Kreutz-Delgado. (2005), Machine Learning Methods for Predicting Failures in Hard Drives: A Multiple-Instance Application, *J. Mach. Learn. Res.*, Volume 6, ISSN-1532 4435, pages -783-816.

[13] S.J. Shin, J.Woo and S.Rachuri. (2014), Predictive Analytics Model for Power Consumption in Manufacturing, *Procedia CIRP*, Volume 15, ISSN-2212 8271, pages 153-158.

[14] A.M.Turing. (1950), Can a machine think. Mind, 59(236), pages 433-460.

[15] S.Muggleton. (2014), Alan Turing and the development of Artificial Intelligence, *AI Communications*, vol. 27, no. 1, pp. 3-10, DOI: 10.3233/AIC-130579.

[16] M.Awad, and R.Khanna . (2015), Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers, *Springer*, pages 1-18.

[17] T.M. Mitchell. (1980), The Need for Biases in Learning Generalizations, New Brunswick, New Jersey, USA: Rutgers University.

[18] J.Oh, K.Torisawa, C.Hashimoto, R.Iida, Ryu, M.Tanaka, and J.Kloetzer. (2016), A Semi-supervised Learning Approach to Why-question Answering, *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, amid-3016325, pages 3022–3029, AAAI Press.

[19] K.Das. (2017), A Survey on Machine Learning: Concept, Algorithms and Applications, *International Journal of Innovative Research in Computer and Communication Engineering*

[20] L.Breiman, J.H.Friedman, R.A.Olshen and C.J.Stone. (1984), Classification and Regression Trees, *Wadsworth International Group.*

[21] P.Geurts. (2006), Extremely randomized trees, *Machine Learning*, Volume 63, ISSN 1573 0565, pages 3-42

[22] A.Vedald. (2010), Efficient Additive Kernels via Explicit Feature Maps, *IEEE transactions on pattern analysis and machine learning*

[23] H.Zou and T.Hastie. (2005), Regularization and variable selection via the Elastic Net, *Journal of the Royal Statistical Society*, Series B, Vol:65, pages 301-320.

[24] J.W.Tukey. (1977). Exploratory Data Analysis. *Pearson.* ISBN 978-0201076165.

[25] A.L. Blum and P.Langley. (1997), Selection of relevant features and examples in machine learning, *Artificial Intelligence*,Volume 97, Issues 1–2,Pages 245-271, ISSN 0004-3702.

[26] G.Chandrashekar and F.Sahin. (2014), A survey on feature selection methods, *Computers and Electrical Engineering, ELSEVIER*, Volume 40, Pages 16-28, ISSN 0045-7906.

[27] I.Jolliffe. (2011), Principal Component Analysis, *International Encyclopedia of Statistical Science, Springer Berlin Heidelberg*, Pages 1094-1096, ISBN 978-3-642-04898-2.

[28] M.P.Wand, M.C.Jones, (1994). Kernel Smoothing. *Chapman Hall/CRC Monographs on Statistics Applied Probability (60).* Boca Raton, FL, U.S.: Chapman Hall.

[29] S.Patro, K.K.Sahu, K. K. (2015), Normalization: A preprocessing stage. arXiv preprint arXiv:1503.06462.

[30] F. T. Liu, K. M. Ting and Z. Zhou, "Isolation Forest,*2008 Eighth IEEE International Conference on Data Mining*, pages. 413-422.

[31] R.Hyndman and Y.Fan. (1996), Sample Quantiles in Statistical Packages *The American Statistician*, 50(4), 361-365.

[32] Y.Reich, S.V. Barai.(1999), Evaluating machine learning models for engineering. problems,*Artificial Intelligence in Engineering*,Volume 13, Issue 3,Pages 257-272,ISSN 0954-1810.

[33] A. Becker William, S. Clevel and M.J. Shyu.(1996), A Tour of Trellis Graphics Richard Bell Laboratories Murray Hill, New Jersey 07974.

[34] J.H.Friedman. (2001), Greedy function approximation: A gradient boosting machine,*Annals of statistics*, Pages 1189-1232.

[35] P.Hughes. (2000), Python and Tkinter Programming, *Linux J.*, Vol 2000, ISSN 1075-3583.

A

Appendix 1

A.1 Normal Distribution plot for testflow

The figure below shows normal distribution plot for rest of three tests flows.

A.1.1 Radio unit calibration testing

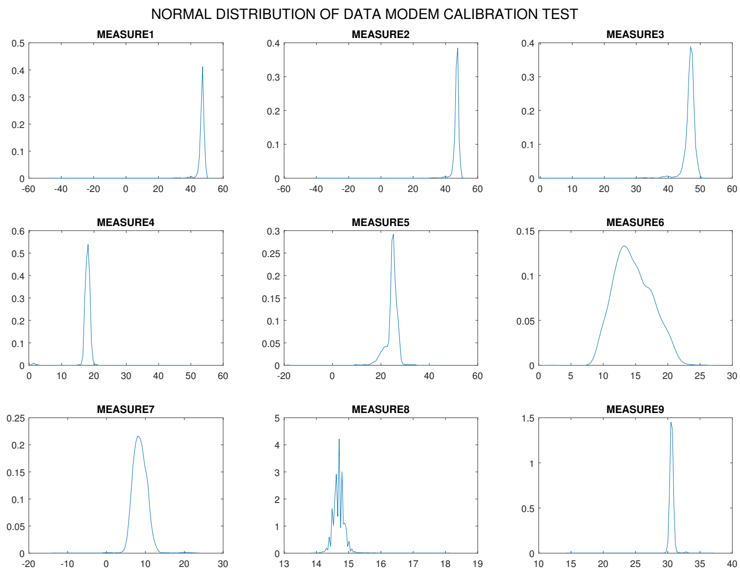

Figure A.1: Normal distribution plot for Radio unit calibration testing

A.1.2 Temperature Testing

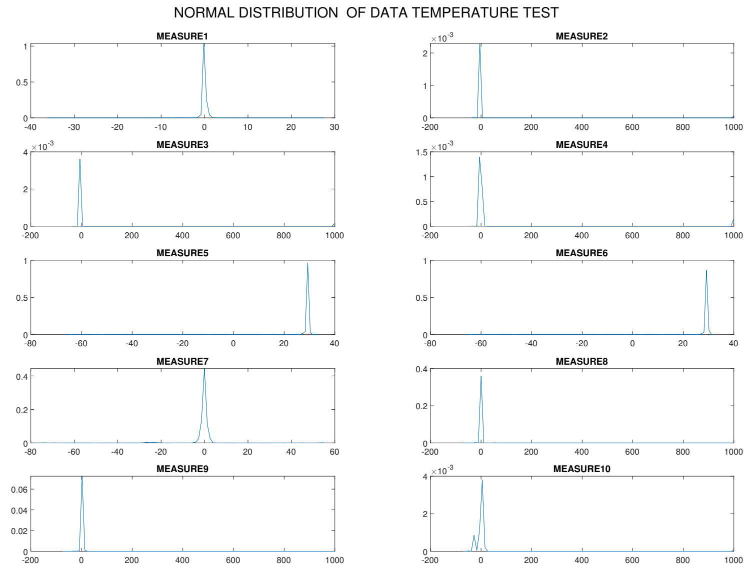

Figure A.2: Normal distribution plot for temperature testing

A.1.3 Final testing

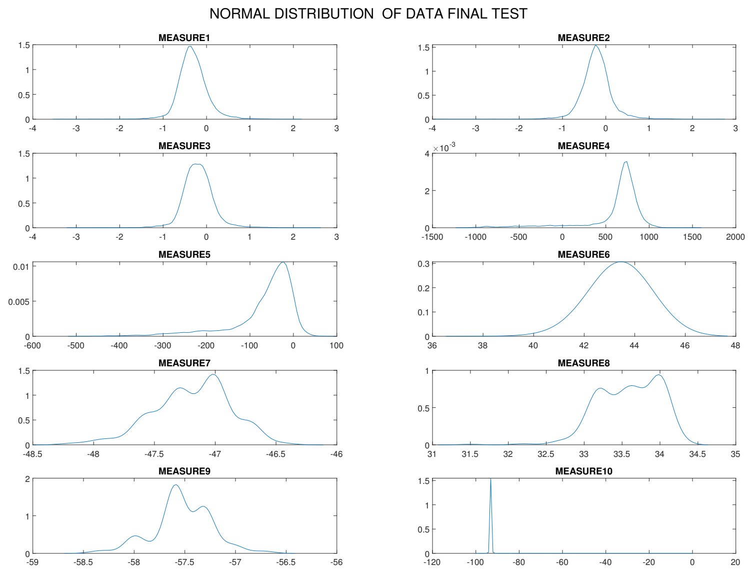

Figure A.3: Normal distribution plot for Final testing

A.2 Normal Distribution plot for test flow with normalization

The figure below shows normal distribution plot with normalization for rest of three tests flows.

A.2.1 Radio unit calibration testing

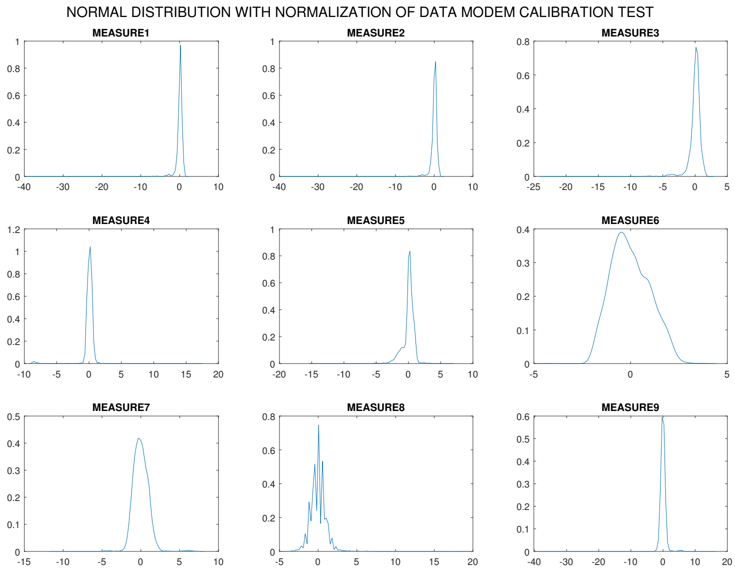

Figure A.4: Normal distribution plot for Radio unit calibration testing with normalization

A.2.2 Temperature Testing

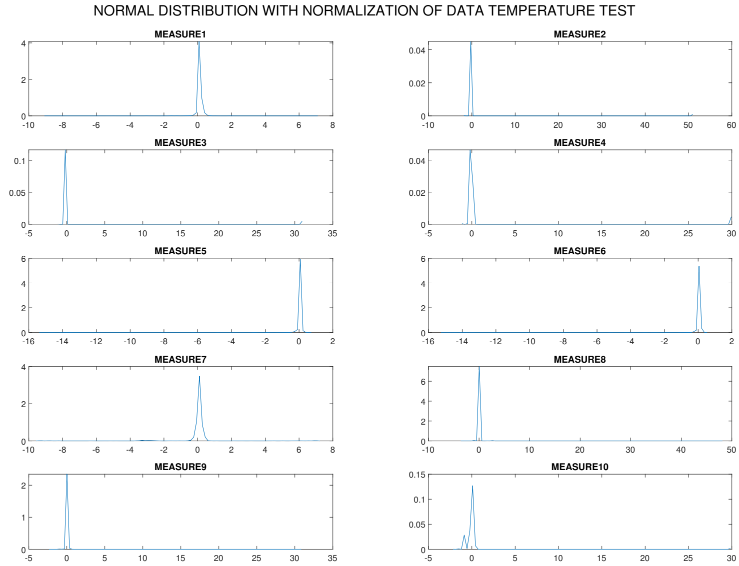

Figure A.5: Normal distribution plot for temperature testing with normalization

A.2.3 Final testing

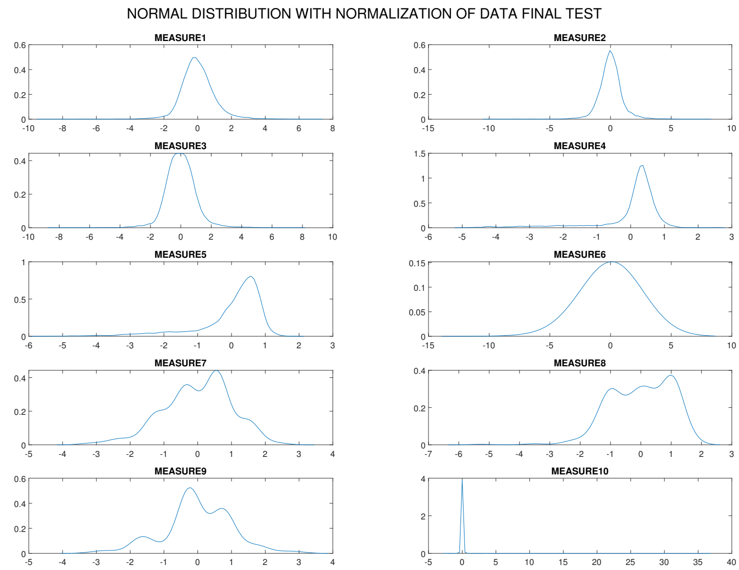

Figure A.6: Normal distribution plot for Final testing with normalization

A.3 Box plot for outlier identification

The figure below shows box plot for outlier identification for rest of three tests flows.

A.3.1 Radio unit calibration testing

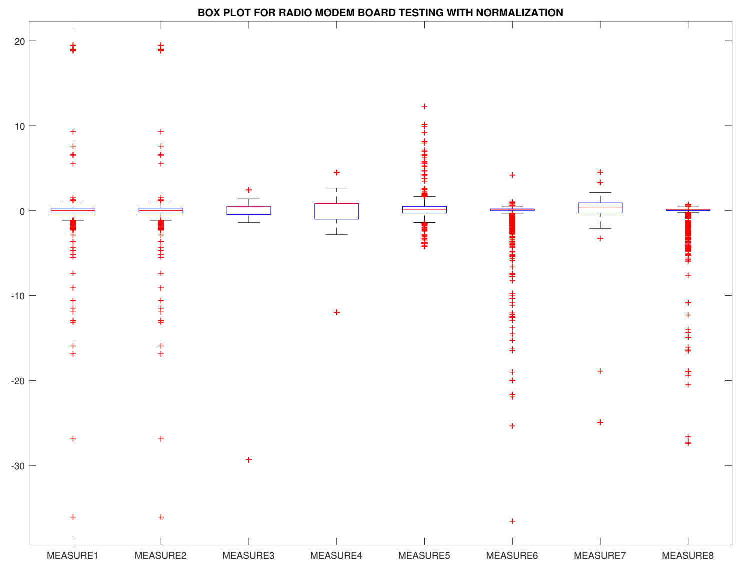

Figure A.7: Box plot for Radio unit calibration testing

A.3.2 Temperature Testing

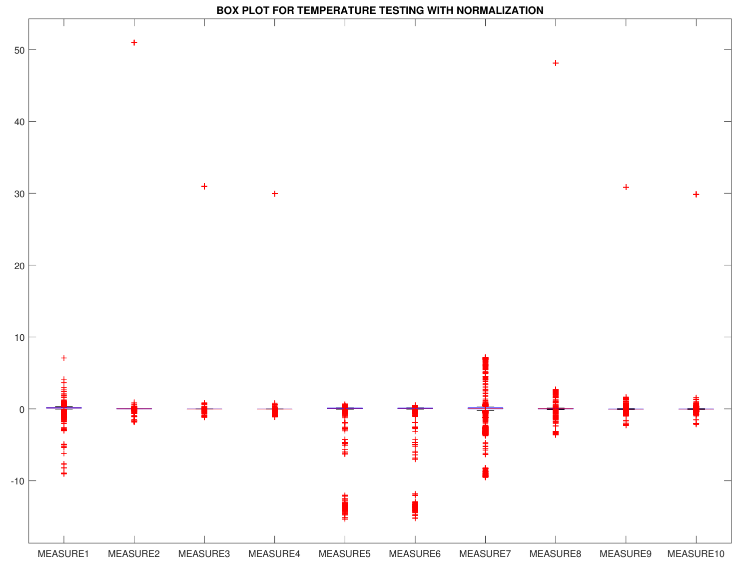

Figure A.8: Box plot for temperature testing

A.3.3 Final testing

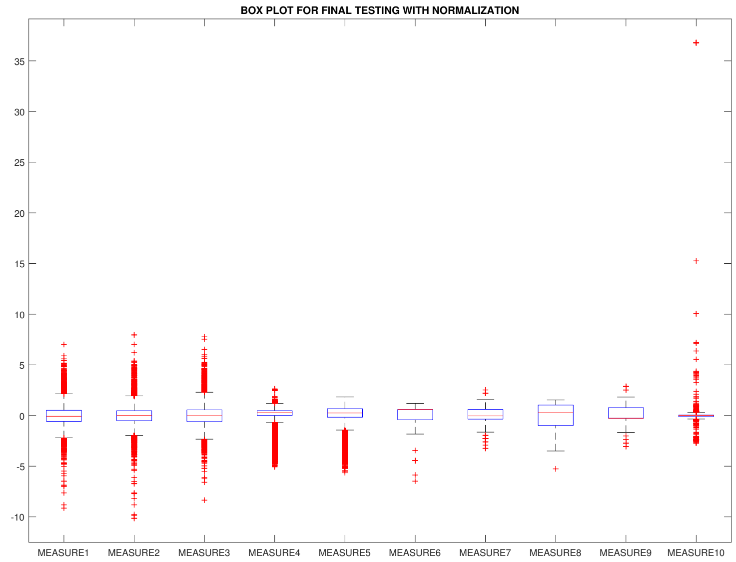

Figure A.9: Box plot for Final testing

A.4 Box plot after outlier removal

The figure below shows outlier removed box plot for rest of three tests flows.

A.4.1 Radio unit calibration testing

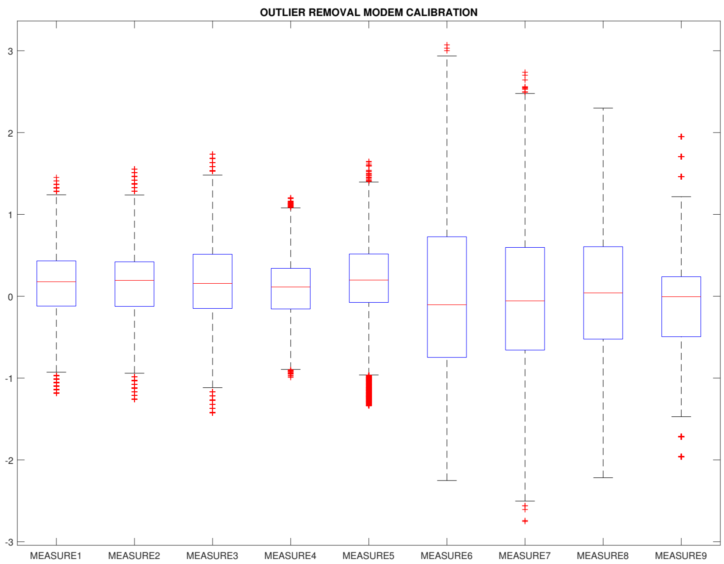

Figure A.10: Box plot after outlier removal for Radio unit calibration testing

A.4.2 Temperature Testing

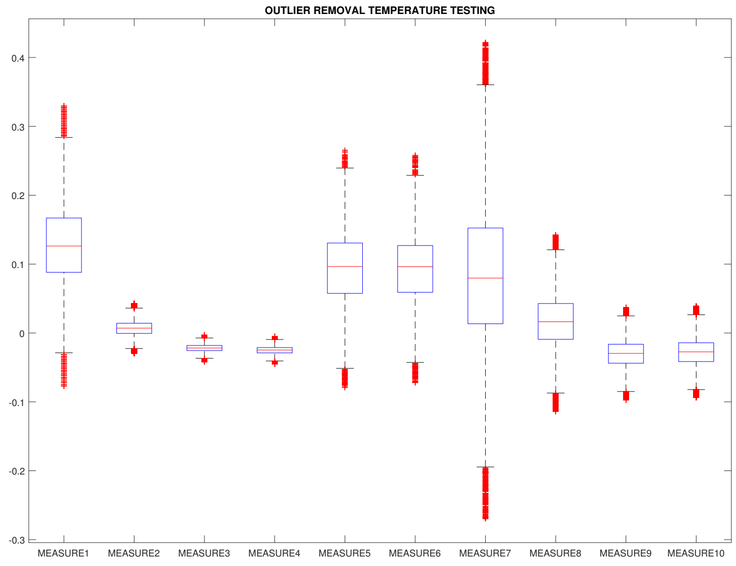

Figure A.11: Box plot after outlier removal for temperature testing

A.4.3 Final testing

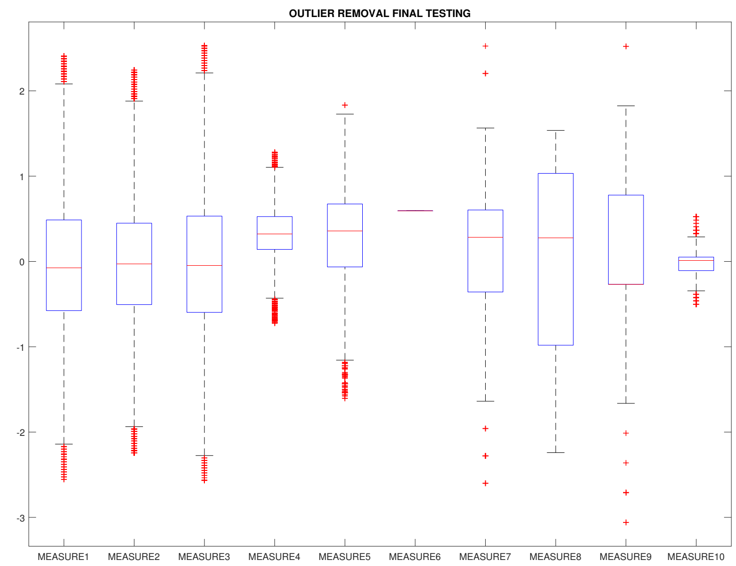

Figure A.12: Box plot after outlier removal for Final testing

A.5 Data Robot API

The model that can be added in data robot can be analysed as shown in the figure.

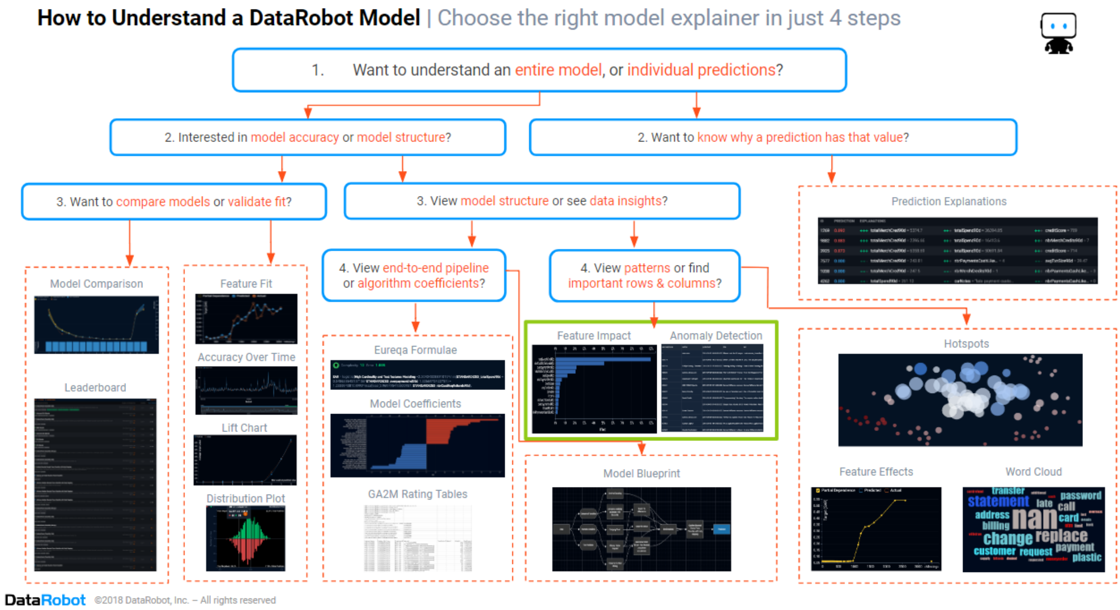

Figure A.13: Model in Data Robot