ANALYSING THE EFFICACY OF FEATURE ELIMINATION AND ADAPTIVE SAMPLING IN META-MODEL BASED EXPLORATION OF AGENT-BASED SIMULATION MODELS

by

Ecemnaz Yıldız

B.S., Industrial Engineering, Middle East Technical University, 2016

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Industrial Engineering

Boğaziçi University

2022

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my advisor, Gönenç Yücel for his consistent guidance and support. This thesis could not be completed without his endless patience and precious feedback. He sparked my curiosity and made this study very enjoyable, even in the most stressful times.

I would also thank Mert Edalı and Mustafa Gökçe Baydoğan for attending my thesis jury and sharing their valuable comments. I am indebted also to Pelin Yurdadön for the inspiration and courage she gave to me, as well as the academic contribution she provided.

Words cannot express my gratitude to my family for their support during this journey. Eren Yıldız, I am deeply grateful to have you nearby me in all the darkest and brightest times. The never-ending support you give me is the expression of true love. My mother Firdes Bay and my father Muzaffer Bay, you are the light of my way. Without you, I could have never achieved where I am. Last but not least, Elif Şahin and Ezgi Yerlikaya Ay, thank you for your unconditional love and belief in me. After every storm in my life, you were and will be there like a rainbow that gives hope. I could not have been able to complete this journey without your emotional support.

ABSTRACT

ANALYSING THE EFFICACY OF FEATURE ELIMINATION AND ADAPTIVE SAMPLING IN META-MODEL BASED EXPLORATION OF AGENT-BASED SIMULATION MODELS

In this thesis, an advanced procedure is constructed to investigate agent-based simulation models. A meta-modeling approach, that utilizes adaptive sampling and feature elimination methods is used in the proposed procedure. The procedure aims to build a machine learning model that replicates the input-output relationships of the original agent-based simulation model and accurately predicts the output of interest. Thanks to feature importance measurements, the proposed procedure also enables researchers to analyse the relationships between the agent-based simulation model parameters and the output of interest. The Random Forest algorithm is used for building the meta-model. The adaptive sampling method is utilized to create a high-quality data set to train the meta-model. The feature elimination process is applied to enable meta-model to prevent the curse of dimensionality and keep the focus on important features regarding the output of interest. The proposed procedure is applied to a complex agent-based meta-model to evaluate its performance. A recent agent-based simulation model, that is analyzing socio-dynamic systems, is selected for application considering its probabilistic nature and wide range of parameters. Moreover, previously proposed meta-modeling approaches in the literature are reviewed and performance comparisons are assessed with the proposed procedure. Both the accuracy of output predictions and the validation of feature elimination decisions are analysed in detail. The conducted experiments and analysis showed that the proposed advanced procedure estimates the output of the original simulation model in an accurate and efficient way, and it outperformed the previously proposed meta-modeling approach in terms of accuracy.

ÖZET

ETMEN-TABANLI BENZETİM MODELLERİNİN META MODELLEME YOLUYLA İNCELENMESİNDE DEĞİŞKEN SEÇİMİ VE UYARLANABİLİR ÖRNEKLEME YÖNTEMLERİNİN FAYDA ANALİZİ

Bu tezde, etmen-tabanlı simülasyon modellerini incelemek için gelişmiş bir prosedür tasarlanmıştır. Bu prosedürde, değişken seçimi ve uyarlanabilir örnekleme tekniklerinden faydalanan bir meta-modelleme yöntemi kullanılmıştır. Prosedürün amacı, özgün etmen-tabanlı simülasyon modelini taklit edip çıktılarını doğru bir şekilde tahmin edebilecek bir makine öğrenmesi modeli oluşturmaktır. Tasarlanan prosedür, meta-model değişkenlerine ait önem değerlerinin hesaplanması sayesinde, etmen-tabanlı model para-metrelerinin çıktı üzerindeki etkilerinin analiz edilebilmesini de sağlar. Meta-modelin oluşturulmasında Rastsal Orman algoritması kullanılmıştır. Meta-modeli eğitmekte kullanılacak yüksek kaliteli veri setinin oluşturulması için uyarlanabilir örnekleme yön-teminden faydalanılmıştır. Boyutsallığın yarattığı sorunları önlemek ve meta-modelin odağını çıktı açısından önemli değişkenler üzerinde tutmak amacıyla değişken eleme yöntemi uygulanmıştır. Önerilen prosedür, performans değerlendirmesi için, karmaşık bir etmen-tabanlı simülasyon modeli üzerinde uygulanmıştır. Uygulama için, olasılıksal doğası ve geniş parametre yelpazesi göz önünde bulundurularak, sosyo-dinamik sistemleri analiz eden güncel bir etmen-tabanlı simülasyon modeli seçilmiştir. Ayrıca tasarlanan prosedürün performansı, literatürde daha önce önerilen meta-modelleme yöntemleriyle karşılaştırmalı olarak değerlendirilmiştir. Çalışmada hem öngörülen çıktı değerlerinin hem de yapılan değişken seçimlerinin doğruluğu detaylı bir şekilde analiz edilmiştir. Deney ve analizler, önerilen prosedürün orijinal simülasyon modeline ait çıktıları doğru ve verimli bir şekilde tahmin ederken daha önce önerilmiş meta-modelleme yöntemine göre daha doğru sonuçlar ürettiğini göstermiştir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

$check\_elim$	A variable to activate the feature elimination process
$elim\_iter$	Iteration number of the feature elimination process
$iter*$	Iteration number at which the feature elimination process is activated
$iter$	Current iteration number of the procedure
k	Number of features
M	Meta-model
MSE_{min}	Reference value of Mean Squared Error
MSE_i	Mean Squared Error value after permuting each feature value randomly
N	Number of instances
$ntree$	Number of trees in the random forest
$\text{N}(\mu, \sigma^2)$	Normal distribution with mean and standard deviation
o	Out-of-bag error allowance rate
OOBerror	Out-of-bag error
p*	A fraction defining the number of features to be evaluated for elimination
p	A fraction defining the maximum number of features for elimination
P(e)	Perceived expertise
P(t)	Perceived trustworthiness
P(h)	Prior belief in a hypothesis
P(h|rep)	Posterior probability of a hypothesis
T	Training data set
$\hat{y}$	Agent-based model output value that is predicted by the meta-model
y	Actual output value of the agent-based model observed via simulation run
μ	Mean value of the distribution

σ^2	Standard deviation of the distribution
%IncMSE	Percent increase of Mean Squared Error

LIST OF ACRONYMS/ABBREVIATIONS

ABM	Agent-Based Model (i.e., Agent-Based Simulation Model)
MSE	Mean Squared Error
OOB	Out-of-bag
RMSE	Root Mean Square Error

1. INTRODUCTION

Agent-based models (i.e., ABMs) are used for simulating and analyzing complex systems which contain interacting individuals making decisions and determining their own behaviors. Autonomous individuals, named agents, interact with other individuals and the environment in the agent-based models. Agents follow certain rules which determine their behavior routines. Thus, ABM is a very powerful approach while studying systems with emerging behaviors according to individual decisions and interactions.

Agents have specific properties such as states, perceptions of the environment, level of resources, and locations. The ability to put that diversity, which arises due to those distinctive properties, among agents into the model enables researchers to simulate systems with heterogeneous interacting individuals who make decisions about their behaviors. Thus, problems containing social interactions under biophysical constraints can be simulated [1] and complex dynamics, which can be easily overseen with other modeling approaches can be captured using ABMs.

As a result of those benefits, ABMs are chosen as an appropriate tool by modelers to work on problems from a wide range of areas from water use to traffic simulation and can be applied to modeling proteins as agents to modeling interactions among nation-states [2].

Agent-based modeling is accepted as a powerful tool to explore the dynamics of social systems and nonlinear relationships within them. That is because ABMs can represent these kinds of systems better than mathematical models like linear programming models or statistical forecasters. A previous study states that the ABM approach has its greatest power in situations with unpredictable future scenarios [3]. That study also suggests that, despite the power of ABM in unpredictable situations, when the uncertainty increases the effectiveness of analytical methods decreases, and decision-making gets difficult using analytical methods. In order to represent the complexity of real systems, models get larger by including lots of variables. In that case, construct-

ing and understanding the model becomes an exhaustive task. Moreover, the size and complication of the output increase with the increasing size of the model. Output becomes harder to understand and analyse, thus ABMs' interpretability gets limited by the complexity of real-life problems. In that manner, researchers suggest utilizing analytical approaches together with ABMs to improve decision-making abilities and to get the maximum benefit from ABMs [3].

Previous studies are conducted on building a meta-modeling procedure combining an analytical approach with ABMs to analyse relationships between the parameters and the output of simulation models [4]. A procedure incorporating adaptive sampling with feature elimination is proposed to figure out relationships between the parameters and the output of simulation models. Results of the developed procedure are applied to an agent-based segregation model. Initial results of the procedure are evaluated in that study. Despite showing promising results, many aspects of the proposed approach remain unevaluated as it is not applied to a large-scale simulation model in that study. Increasing scale is expected to lead to efficiency and accuracy problems. This study aims to search for improvement opportunities in the suggested procedure and then analyse the behavior and performance of the improved procedure on a more complex model. The efficiency and accuracy of the meta-model that is generated via the proposed procedure to replicate the original ABM are examined considering a more complex simulation model with a larger set of parameters.

2. LITERATURE REVIEW

In the literature, studies combining analytical and statistical tools with ABMs are mainly focused on two topics. One is the estimation or tuning of parameters that are used in the simulation. Another focus is to use the analytical and statistical tools to explore the dynamics of the emerged model behavior of the system during the simulation in a more efficient way. In this manner, machine learning and sampling techniques are used with ABMs.

Supervised machine learning and intelligent sampling are combined in the design of a surrogate meta-model in a previous study for agent-based model calibration and parameter space exploration [5]. An approach is developed and evaluated to explore the parameter space of the agent-based model, using a non-parametric machine learning surrogate in the study. In that approach, an iterative sampling algorithm is used which intelligently searches the response surface taking limiting conditions (i.e., model constraints) into account. The approach is constructed as an iterative algorithm that searches for a good approximate of a surrogate model that can be used for different agent-based models using a predefined number of agent-based model runs. The approach starts with creating an unlabeled sample pool from the parameter space. Then a sample is drawn from the pool, the initial surrogate learning algorithm is trained and labels of the sample predicted over the pool. If the predefined iteration budget is not reached yet, the predicted labels are added to the pool. The method stops and records the last predictive model as a good surrogate when the predefined iteration budget is reached. To apply the algorithm, preliminary decisions need to be taken. Firstly, the machine learning algorithm should be selected to be used as a surrogate of the original agent-based model. Secondly, a sampling method should be selected to work on parameter space to create the training set for the selected machine learning algorithm. Thirdly, a performance metric should be selected to evaluate the surrogate machine learning model's performance. After taking these three decisions, the algorithm can be applied iteratively. In that study, researchers report that the machine learning surrogate provides a good representation of the original agent-based model in

terms of accuracy while reducing the parameter space exploration time.

As stated before, meta-models are also used to explore the emergent dynamics of complex systems. Emergent behaviors are caused by the interaction of variables within the system. Interactions between variables are embedded in models as complex and nonlinear relationships. Meta-models are used to imitate agent-based models and find out how these relationships affect the emerging behavior. A good meta-model can be used as a representation of an agent-based model, and it can collaborate with the agent-based model for helping to explore the behavior space [6]. Moreover, a meta-model that is balancing accuracy and interpretability would be very useful to understand the input-output dynamics of models with lots of parameters and nonlinear relationships. If the meta-model is proven to be accurate, it is a good way of validation and verification of ABMs to observe the effects of inputs on output. In a previous study [7], Random Forest is used as a meta-modeling approach imitating an agent-based model. The meta-model is used to predict the output of different combinations of variable values, and a simple procedure is constructed for investigating the behavior space of the agent-based model and estimating the model outputs. A crucial step for that procedure is the selection of training data to train the meta-model. The study aims to work in the least computationally costly way without loss of accuracy. To train the meta-models effectively and to speed up the fitting process, active learning techniques are proposed instead of random sampling. An adaptive sampling strategy is utilized to achieve a good level of prediction accuracy with a lower number of simulation runs in the study.

Despite many examples using meta-models on agent-based models together with analytical and statistical tools, that approach becomes inapplicable for lots of examples in case of the existence of a large number of parameters. Agent-based model parameters are used as features in the representative meta-models. Feature selection is choosing the most relevant features from the entire feature set according to a feature selection criterion and removing the irrelevant features to improve accuracy, reduce time, and simplify results [8]. Feature selection is utilized in ABMs together with the meta-modeling approach in a previous study to select the most relevant features to keep in the model [4]. Simplification of the model and avoiding the curse of dimensionality are

the main aims while using the feature selection in that study. A procedure is suggested using Random Forest meta-modeling to represent the agent-based model. The meta-modeling approach is used incorporating adaptive sampling with feature elimination to figure out the relationships between parameters and the output of simulation models in case of insignificant parameters. In that study, the proposed procedure is applied to an agent-based segregation model and analyses are made on the output. Results show that the suggested procedure can be used as an efficient way to construct meta-models which give insights about the dynamics embedded in the simulation model.

The segregation model has many benefits for the purpose of testing the approach thanks to its simplicity. Yet, it did not enable researchers to explore all elements of the proposed procedure, because it has a small number of parameters that can be separated easily in terms of their effects on output. Thus, evaluating that procedure on a large-scale, more complex model would provide very useful insights about the procedure. Potential improvement areas can be pointed out after the application of that procedure on a more complex model.

3. PROBLEM DEFINITION

Agent-based models are mostly used to model complex systems which contain dynamic and non-linear relationships like real-life problems. Thus, large numbers of parameters are included in agent-based models to represent the real problem context. However, many drawbacks arise when more parameters are included in an agent-based simulation model. With an increasing number of parameters, the computational cost of running the ABM increases. Simulation run duration also increases with the increasing number of parameters. Larger and more complex models result in high dimensional outputs which require sophisticated analytical approaches for analyzing and reporting [9].

Agent-based model parameters are used as machine learning model features in the approach of representing agent-based models with meta-models. Not all agent-based model parameters would have a significant effect on the output of interest. Machine learning models tend to overfit and become less comprehensive in case of including irrelevant features [10]. Moreover, the analysis of meta-model results gets complicated as well with the increasing number of features. Thus, keeping a fewer number of features in the meta-model has great importance from the computational efficiency and interpretability point of view. Features that are significant regarding to the output should be selected carefully to be kept in the meta-model. It should be noted that the significant meta-model features correspond to the ABM parameters that have a significant effect on the output in the simulation model.

In a former study, the abovementioned improvement areas are discussed, and an advanced meta-modeling procedure is defined using adaptive sampling and feature elimination [4]. A meta-model is used to imitate the original agent-based model. The Random Forest technique is applied for meta-modeling. The adaptive sampling method is used to create a good quality dataset to train the meta-model. That method enables researchers to select valuable samples which bring important information about the dynamics of emergent behavior. It is well known that a high number of parameters

and nonlinear interactions between variables make interpretation of complex systems harder. An increasing number of parameters increases the dimensionality of parameter space and size of output thus, it contributes to the curse of dimensionality. To reach a simpler meta-model, insignificant parameters are removed with feature elimination in that former study [4]. The elimination process provides dimensionality reduction preserving meta-model accuracy and promoting interpretability. After eliminating redundant features, adaptive sampling works on a lower dimension space. That improves the performance of the meta-model by enabling adaptive sampling to focus more on significant parameters.

Despite providing a very promising approach, the procedure is not applied to a large-scale and complex model in the previous study. Thus, downsides and improvement areas could not be searched in detail. Some elements of the procedure could not be examined due to the small number of model parameters which have easily been observed with their different effects on output. It is expected that efficiency and accuracy problems will arise with application to a larger scale and more complex model. Multimodality and continuous output may also lead to an unsuccessful approximation of the simulation model. Prediction performance is expected to deteriorate when the procedure is applied to a simulation model with categorical variables and class imbalance.

Application of previously constructed meta-modeling procedures to a larger scale, more complex agent-based model and make performance analysis is the primary purpose of this study. The approaches discussed in the literature review are very promising and have many advantages, yet they have many points to discuss and improve. After application to a large-scale model, this thesis focuses on findings and improvement opportunities that are already expected. Moreover, the predictive power and efficiency performance of the previously proposed procedure in the literature highly depend on user-defined parameters. Technical knowledge is required to comprehend these user-defined parameters, and the design of experiments is needed to assign proper values to them. Herewith, this thesis aims to develop a simplified approach by removing these complex user-defined parameters of the previously proposed procedure.

Simulation models which are constructed to investigate socio-dynamic systems are a good candidate to implement the previously defined meta-modeling approach with feature elimination, as they usually have high complexity and a wide range of parameters. Therefore, a literature review is conducted for current studies on socio-dynamic systems including agent-based simulation models. During the literature review, it is remarked that investigating the spreading of misinformation is an interesting topic for researchers. That topic got more popular over the past decade since social media became much more preferable to mass media. Social media platforms provide space to people for connecting others who think like themselves. People share their ideas via posts and content with like-minded others. As a result, people are inclined to form clusters around common interests, worldviews, and narratives. Users tend to spread information that is taken from friends having a similar profile and that creates polarization which results in the formation of homogeneous clusters [11]. Those homogenous and polarized communities are also known as echo chambers. An echo chamber can also be defined as an environment in which a person can only come across ideas and information which reflect and reinforce their own. People's tendency to give more credit to information that is in favor of their existing opinion creates a confirmation bias, and that also feeds the construction of echo chambers. In case of confirmation bias, people pass over fact-checking and choose immediately to accept the given information. Thus, polarization and confirmation bias have a significant impact on misinformation spreading on online social media [12].

A recent study investigates the emergence of echo chambers in social networks using an agent-based simulation model [13]. Researchers develop a model which considers the probabilistic nature of social media users' belief updating mechanism. The model takes into account that the communicating user's perceived credibility and trustworthiness level on other users have an effect while convincing them. In order to point out sufficient causes of echo chamber formation with a robust and realistic model, a wide range of parameters are included in the model. As a result, the model is chosen as a good candidate to implement the previously constructed meta-modeling procedure in terms of topic, scale, and complexity manners.

4. BACKGROUND

Agent-based models are widely used in problems that contain non-linear and complex interactions between their parameters and outputs. Interactions between parameters and outputs are analysed through many simulations which are designed with changing parameter values. These parameters include not only policy parameters that can be determined by policymakers but also model parameters that are defined by the model's nature and taken as constant by policymakers. Thus, the design of experiments for investigating relationships between parameters and outputs is a challenging task all by itself. Besides this, making detailed analysis and finding out the interactions is much more compelling than designing experiments. In such a straightforward way, the design of experiments and result interpretations depend on the researcher's level of expertise.

As time goes by agent-based modeling becomes used more in various new areas which have complicated problems with large parameter sets. Thus, models get more complex, including more parameters interacting with each other, and understanding the dynamics of the models becomes harder for even models' own developers. Sometimes it is not clear whether a significant result emerged due to the effects of parameters and assumptions or simply because of errors and artifacts in the model design [14]. As a result, conducting manual analysis and interpreting results become an exhaustive task. That problem is addressed in a former study "Analysis of Agent-Based Simulation Models Through Meta-modeling" [6]. An advanced analysis procedure is proposed utilizing machine learning tools and sampling methods in that study.

4.1. Agent-Based Simulation and Meta-modeling Approach

When an agent improves its performance after making observations on some tasks, that means the agent is learning. The concept of machine learning comes into the picture when the agent becomes a computer (or model). When a computer observes data, constructs a model based on the observed data, and uses it to interpret and solve

a problem, that concept is called machine learning [15]. Machine learning methods are overperforming the human researchers to foresee and evaluate possible future situations as those models are trained using massive datasets that no human can analyse and learn in detail. Moreover, the design of a solution is way harder than constructing the problem most of the time. For example, recognizing faces is a task that is solved subconsciously in the human mind, and it cannot be expressed as a computer program except in machine learning algorithms. Machine learning models search for meaningful relationships and patterns within a problem using examples and observations [16]. These algorithms are applied iteratively to the training data to learn and improve their performances. During that process, models find out the hidden insights and meaningful patterns without being programmed by developers [17]. Thus, these algorithms are good alternatives to be utilized for investigating complex interactions between inputs and outputs of agent-based simulation models.

Edali proposed that a machine learning model can be used as a representation of the original agent-based simulation model, and investigations can be made through that model [6]. That approximate representation of the agent-based simulation model is a meta-model. The meta-modeling approach shortens the modeling cycle and analysis time as researchers work on estimated outputs found by a meta-model, instead of the real simulated output of each parameter combination. In addition to shorter modeling and execution times, an interpretable meta-model can bring advantages to a better understanding of input-output relationships as researchers suggest. However, the meta-model should be built carefully to maintain the balance between accuracy and interpretability as a simple model may miss out on some relations between inputs and outputs. On that concern, Edali conducted a comprehensive discussion on alternative representations of agent-based simulation models that can capture input and output relationships in a simple yet accurate way [6]. They also compare different data selection methods to train meta-models. By doing so, researchers come up with a comprehensive approach to represent and analyse agent-based simulation models with a well-trained meta-model in an effective way.

4.1.1. Random Forest Meta-modeling

Different machine learning models may yield different outputs with the same input as their prediction processes are different from each other. Ensemble Learning is using multiple models together as a single one to increase the predictive power by reducing variance and bias [18]. Decision tree learning is a predictive method that is easy to implement and interpret, but not so competitive as other learning approaches in terms of accuracy. The Random Forest method is built to increase the decision tree learning method's predictive performance by aggregating many decision trees as an ensemble model [19]. The decision trees, constructing the forest, are grown with two random moves. The first one is that each tree is trained using a bagging method. Bagging (bootstrap aggregation) makes each tree be trained with different data sets which are created from the original training sets by random sampling with replacement. The second one is that the feature set is selected randomly at each node for splitting. Random Forest models are applicable to both classification and regression problems. In Random Forest classification, majority voting is applied to select the most frequent class to return as the predicted label. In Random Forest regression, the average output of each tree is defined as the predicted result [19].

Edali discussed the selection of meta-modeling techniques according to research objectives [6]. As a result of their detailed analysis and comparisons with other methods, they suggested that Random Forest stands as a good technique when the aim is to grasp the input-output relationships of agent-based models with acceptable accuracy performances. Despite the minimal loss of accuracy, the Random Forest meta-modeling technique presents interpretable results for investigating the inner dynamics of agent-based simulation models.

4.1.2. Meta-model Training with Active Learning

Machine learning models learn from a training data. Models discover relationships between inputs, gain understanding, make predictions, and evaluate the confidence of these predictions based on the provided training data. Therefore, machine learning

model performance depends on the quality and quantity of training data as much as the algorithm used. To imitate an agent-based model with a representative meta-model, the meta-model should be trained with the output of many agent-based simulation runs. That requires running the agent-based model many times which takes a long time in most cases. Thus, selecting a proper sampling method for training data set construction would be very beneficial for training the meta-model efficiently.

Different methods can be selected to take samples from parameter space according to computational power constraints. Latin hypercube sampling is one of these methods that can give an improved coverage of parameter space in case of a fixed number of samples which is independent of the dimension of the space [20]. Latin hypercube sampling is a preferred method in model fitting thanks to its advantages in cases when dominant components exist. The method ensures that each of the components is represented regardless of being one of the most important ones [21].

Despite having many advantages, even the improved Latin hypercube sampling methods are categorized as one-shot sampling techniques. The reason is that, those samples are generated with a specific number of input parameter combinations which is decided before training the meta-model [6]. Thus, Edali states that there is no known way to determine a sufficient sample size to ensure the desired accuracy level before fitting the meta-model according to their literature review. In the literature, they see sequential sampling techniques, which are also known as adaptive sampling and active learning, are used as an alternative to one-shot sampling. Sequential sampling techniques are chosen as powerful alternatives as they have the advantage of starting with smaller sample set sizes and expanding them iteratively to increase model accuracy by using the knowledge that is gained in earlier iterations.

As a result of previously stated interpretations, Edali chooses to start their meta-modeling approach with the generation of several training instances (parameter combinations of agent-based models) having a specific size by using Latin hypercube sampling and finding their outputs (labels of training data sets) via simulation runs [6]. After checking the modality of distribution within the output of the training data set they

apply sequential sampling to each training data set independently. They explained the sequential sampling method in detail as expanding the initial training data in a certain number of iterations by taking a certain number of instances (parameter combinations of the agent-based model) from the unlabeled pool (that is previously generated using the Latin hypercube sampling technique) at each iteration and finding their output by using the simulation model. Depending on the modality of the output, their procedure continues with adding either the numerical results or class labels to the training set. At the end of each iteration, the training dataset is expanded, the meta-model is re-trained, and the accuracy of the meta-model is reported. The procedure returns the latest trained meta-model.

4.2. Feature Selection

Selecting the most informative features is crucial for building a successful predictive machine learning model. The model analyses how the output changes with changing values of features and learns the hidden relationships through selected features, thus it is very important to build the model with the best features. Constructing a machine learning model with features that significantly impact the output helps researchers solve dimensionality problems by removing irrelevant and redundant features [8]. As a result of that, computation time reduces, learning accuracy increases, and interpretability of the machine learning model (itself and its output) increases.

Previous research state that feature selection can be applied to find the important meta-model model features regarding a specific meta-model output variable in the context of meta-modeling of a simulation model [4]. As the meta-model features represent the relative ABM parameters, the feature selection also gives insight into the importances of the ABM parameters regarding the simulation model output variable. The advantages of feature selection are discussed thoroughly in that study. At first, the computational efficiency of the meta-model, thanks to less memory and analysis requirements with faster meta-model training and utilization, is discussed. Moreover, prediction power improvement with mitigation of the curse of dimensionality is remarked as insignificant simulation model parameters are being eliminated and

meta-model focuses more on the significant simulation model parameters through feature selection. Furthermore, it is indicated that understanding embedded relationships within data and making visualizations becomes easier when including fewer and more meaningful meta-model features.

Yurdadön proposes an extension to the "Analysis of Agent-Based Simulation Models Through Meta-modeling." [6] study by using feature selection to find the simulation parameters that significantly affect the agent-based model behavior [4]. That study aims to reach the stated advantages of feature selection by training the meta-model using those selected simulation parameters as meta-model features. An approach is developed combining adaptive sampling and feature selection to achieve a sufficiently comprehensive and accurate meta-model of an agent-based simulation model. In that approach, the meta-model is used to find out the hidden relationships between agent-based model parameters and outputs, adaptive sampling provides good quality training data, and feature selection prevents the complexity in meta-model training by eliminating insignificant meta-model features regarding the output variable.

The proposed procedure has two sub-procedures which are stated in Yurdadön's study as adaptive sampling procedure and feature elimination procedure. The adaptive sampling procedure selects informative instances to feed the meta-model and ensure the quality of the training data. The feature elimination procedure keeps the meta-model to be focused on its significant features and allows adaptive sampling to work in a lower-dimensional space. The procedure starts with the data generation phase utilizing the Latin hypercube sampling method. A relatively small (sufficient to cover the parameter space, but small enough to benefit from the adaptive sampling approach) set of agent-based simulation model parameter value combinations is generated in that step. The procedure continues to obtain the outputs (to be used as labels of the training data) of these agent-based simulation parameter combinations via simulation runs. The meta-model is trained after creating the initial training set. Afterward, the adaptive sampling procedure works iteratively for up to a specific number of iterations (i.e., *Elimination Start Iteration*) during the meta-modeling procedure. By doing so, both the size and quality of the training data set are improved thanks to added instances,

so that the training set becomes good enough to estimate feature importance scores successfully. After completing the predefined number of iterations with the adaptive sampling procedure and reaching *Elimination Start Iteration*, the feature elimination procedure starts. From that iteration, the feature elimination procedure applies together with the adaptive sampling as long as the model finds features that can be eliminated. In order to end the overall process, researchers define an *Iteration Budget*, and the process is terminated when that number of iterations is completed. The flow is summarized in Figure 4.1 which is provided in Yurdadön [4].

Researchers conduct experiments with the segregation model [22], which is a well-known agent-based simulation model. That model is a very simple and interpretable one, and it has only two parameters. That selection provides a good insight into the procedure's performance, but the requirement for application on a larger and more complex system remains. As both parameters of the segregation model are already known to be significant, four new insignificant parameters are added to the model to test whether the procedure is able to eliminate them. Researchers investigate the performance of the proposed procedure in detail and show its power of eliminating the insignificant feature and improving the meta-model accuracy (by eliminating the negative effects of insignificant meta-model features and focusing on the important ones). However, that application does not provide insights into the procedure's ability to retain the features that have a slight impact on the output.

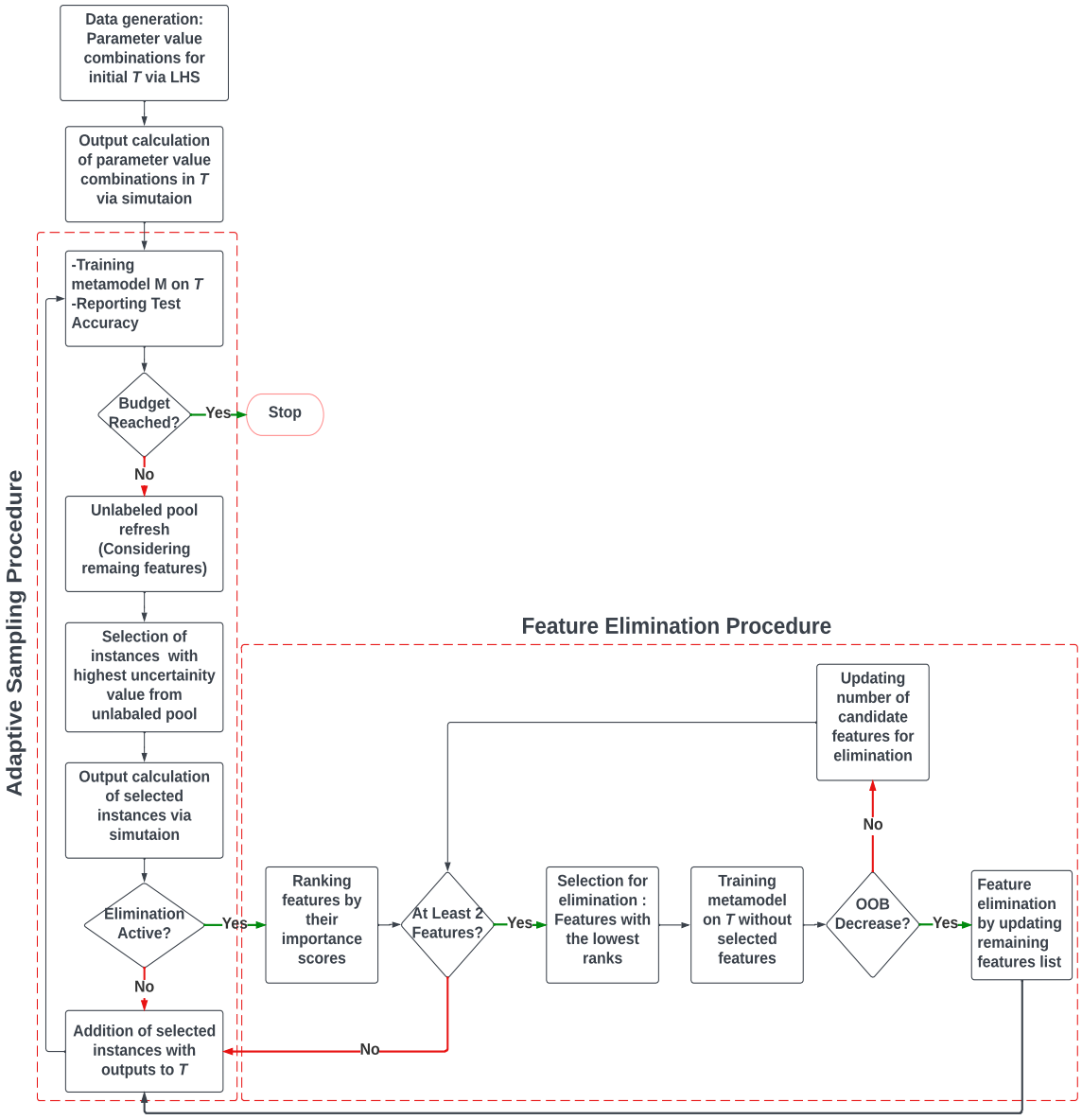

Figure 4.1. Flowchart of the previously proposed meta-modeling procedure.

4.3. Formation of Echo Chambers: Sample Agent-Based Model

As stated in the previous section Fränken and Pilditch's agent-based model, investigating the construction of echo chambers in social networks, is selected to evaluate the meta-modeling procedure with feature elimination [13]. Using that agent-based simulation model, their research investigates a system with an idealized social network user population. The findings show that social media users' ability to instantly share information with each other through a single cascade can be enough to create

echo chambers when that ability is combined with positive credibility perceptions of a communicating source. Moreover, the effects of psychological justifications like bias and individual distinctions are examined in that study, and it is stated that they have no significant impact on echo chamber formation. Besides psychological justifications, researchers also remarked that repeated actions are not required for echo chamber formation as well.

In the agent-based model, the echo chamber formation as a result of a single interaction between generations of network users is analysed. The term "generation" can be explained with an example social network where every member has a specific number of friends (connections) who have no connection with other members' friends. Assume that a member shares an opinion with their friends and friends of them also share the opinion with their own friends. The second-level connections of the initial member are named "second-generation". So, even with a single interaction opinion of the initial member spreads to more members than their friends in the social network. The model considers the social media users' selective acceptance of information from a communicating source according to the source's perceived credibility. A belief update mechanism is introduced in the model to take the trustworthiness of the communicator (source) according to the communicatee (target) into account during the belief formation process. The source credibility is introduced into the agent-based model with a Bayesian perspective. According to that perspective, when an agent encounters a claim that is opposite to her belief, her subjective reliability to that source decreases, but her consideration of the truth of that claim (opposite to her prior opinion) is also revised with that new information; especially in the positive side if she thinks that the source is reliable [23]. So, the target agent's belief and the source agent's perceived credibility are updated with the statement in that process if the source's credibility is not known in advance (by the target agent). Furthermore, alteration of the perceived credibility of the communicating source (according to the target agent) and the belief in the communicated claim (on the target agent's side) depends on the prior statuses of those values according to the target agent, and both of them are uncertain.

The Bayesian source credibility model used in the agent-based model is summarized in Figure 4.2. By using the Bayesian source credibility model in the agent-based simulation model, a cognitive aspect is added to the belief update mechanism. That aspect allows agents to use the combination of credibility perceptions of others during the belief formation process instead of only personal perceptions.

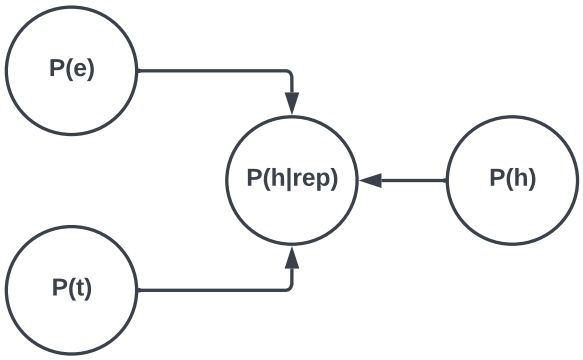

Figure 4.2. Bayesian Source Credibility Model.

In the agent-based model, a social network is simulated with a predefined number of agents. During the initialization of each run, those agents are assigned to random coordinates in a two-dimensional environment and those locations are preserved during the run. After defining the locations, static links (which are not changing during the simulation run) are formed between agents. The links are formed with the nearest neighbors, represent the social network connections, and are used for the intercommunication of opinions between agents. The distances between agents are defined by Euclidean distance. The prior beliefs of agents are sampled from a univariate Gaussian distribution of $N(= 0.5, \sigma^2 = 0.2)$. In a similar vein, subjective expertise and trustworthiness values of agents are sampled with the same distribution. Those values are used for finding estimations of perceived expertise and trustworthiness of a communicating source during the belief update process. An example setup and the distributions of prior belief, subjective expertise, and subjective trustworthiness are shown in Figure 4.3.

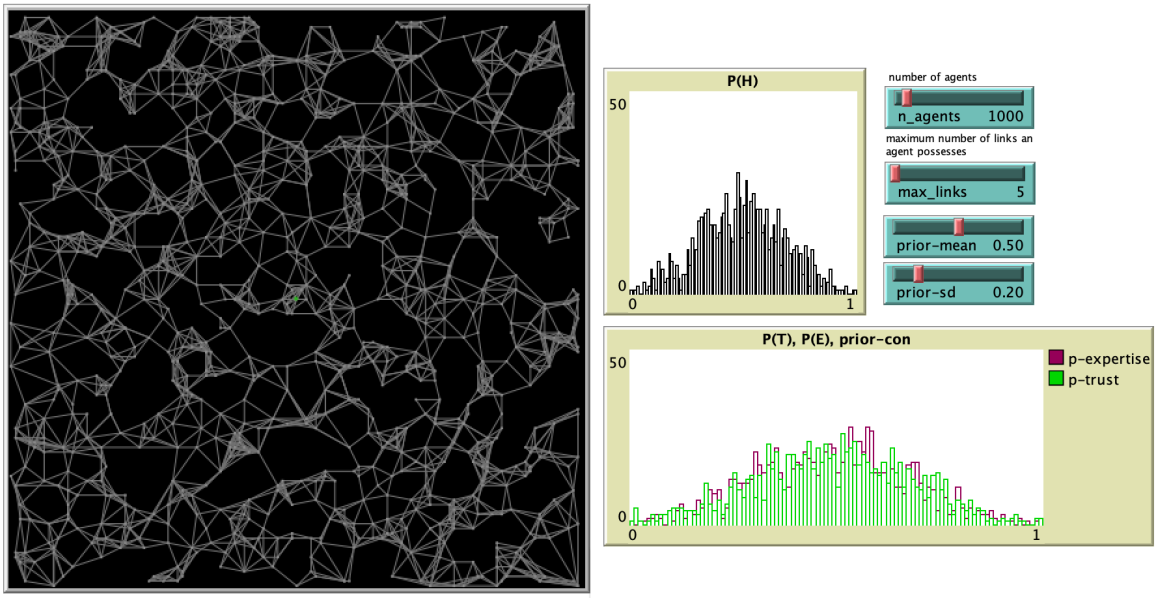

Figure 4.3. An example environment setup with prior belief, subjective expertise, and subjective trustworthiness distributions at the beginning of the run.

Agents of the model have the same behavior routines and they follow the same cognitive processes during the simulation runs. Target (communicatee) agents receive and check messages coming from source (communicator) agents at each time step. There must be a social link formed at the beginning of the simulation run between target and source agents to allow their communication. The incoming message can be either supporting or rejecting the target agent's prior belief. Source agents share messages (supporting or rejecting the target's opinion) according to their own belief declaration in the previous time step. As stated previously, the Bayesian source credibility model is introduced in the model for the belief update mechanism. Correspondingly, target agents update their initial beliefs considering the incoming message according to the Bayesian source credibility model.

In order to take the effect of the source expertise level into account, the *Expertise-Influence* parameter is defined in the agent-based simulation. That parameter is used to determine how does the expertise level of the source influences opinion formation on the target's side. Simply, a source with stronger expertise has higher persuasive power on the target than a source with a lower expertise level.

Perceived expertise and perceived trustworthiness values of source agents are also updated for target agents during the simulation run. To calculate these values, target agents check the incoming messages from the sources and compare them with the opinions of their neighbors (the ones that have social links with the target agents). The target agent then checks the subjective expertise and trustworthiness values of its neighbors related to the source agent. According to the ratio of subjective expertise (and trustworthiness) value of the neighbors that have the same opinion of the source to the total subjective expertise (and trustworthiness) value of all neighbors, a perceived expertise (and trustworthiness) value of the target is calculated.

The probability of opinion declaration is also considered in the agent-based simulation model with the *Prop-Likelihood* parameter. That parameter is used to describe the population's tendency to share their opinions. Each agent has a probability of making its opinion public as much as the *Prop-Likelihood* parameter of the agent-based model. For example, the *Prop-Likelihood* parameter of 0.1 means that each agent has a 10% probability of making its opinion public.

The agent-based simulation run is initiated with an agent that is placed in the middle of the environment and shares a random opinion (support or reject) with her neighbors (agents having social links with her). Besides other agents are shown as neutral ones as they did not express any opinion yet, a prior opinion is assigned to them at the initialization according to the belief distribution which is defined by ABM parameters. After receiving a message from the initial agent, its neighbors make their opinion public according to the predefined opinion declaration probability. These agents are described as the first generation as they are the first group that receives messages and makes opinion declarations. Then target agents of the first generation follow the same belief update and declaration process with the messages they received. Agents cannot become targets once again after declaring their beliefs. The message transmission (communication between linked agents) continues between generations until either all agents have declared their beliefs or the number of agents that are believing (or rejecting) did not change for two periods.

The agent-based model has two main outputs that are analysed in Fränken and Pilditch's research study. The first one is the global proportions of belief within the network. The second one is the average proportion of agents having the same opinion within neighborhoods (linked agents). A complex agent-based model is constructed with many parameters (and relations between them) to analyse these outputs. For example, the value of *Prop-Likelihood* parameter influences the effect of other parameters as it defines the opinion declaration tendency of the individuals. If the *Prop-Likelihood* value is so low, then the effect of *Expertise-Influence* decreases as individuals does not share their opinions. Similarly, as the parameters that control the number of agents and maximum number of links defines the network connectivity density, they influence the effect of other parameters on the outputs as well. Moreover, several probabilistic processes and interactions between these processes are added to the agent-based model to simulate real-world social networks. Therefore, the agent-based model stands as a good candidate to apply the previously mentioned meta-modeling with a feature elimination procedure. A large number of parameters and model complexity enable analyzing the performance of meta-modeling with a feature elimination procedure and point out the possible improvement areas.

5. PROPOSED APPROACH

As stated in the previous sections, collaboratively using meta-modeling, adaptive sampling, and feature selection are claimed to be efficient for investigating agent-based models [4]. In that context, the data is generated through simulation runs. Good quality training and test data sets are created using the adaptive sampling method. The machine learning meta-model is used to indicate embedded dynamics and input-output relationships within the original agent-based model. Feature selection helps to reduce the complexity and prevent the curse of dimensionality by keeping the most significant agent-based model parameters which have an impact on the output of interest. The framework of Yurdadön's approach is defined in the background section and the flow is shown in Figure 4.1. That approach contains several user-defined parameters which directly affect the feature elimination decisions and so the predictive power of the meta-model that is built to represent the ABM. Moreover, these procedure parameters require technical knowledge and the design of experiments to be defined for the application of the procedure to different ABMs. In this thesis, an improved approach is aimed to be constructed in a more user-friendly way, it is applied with a complex ABM and a comparative analysis is made against the previously proposed procedure.

5.1. Previously Proposed Feature Elimination Process

In the previously proposed procedure, before starting the feature elimination process, the training data is enlarged by applying adaptive sampling iteratively. The aim of that design is to reach a data set that enables a better importance score calculation for meta-model features (the model parameters of the original agent-based simulation model) before starting the feature elimination and by doing so, prevent the elimination of important features. In order to enable that design, a user parameter of *Elimination Start Iteration* is introduced to the procedure which represents the iteration number that the feature elimination starts. The selection of *Elimination Start Iteration* value is discussed in detail with the experiments in the previous study of "Adaptive sampling with feature elimination for agent-based models" [4]. Before reaching *Elimination Start*

Iteration, only adaptive sampling is applied, and more samples are added to training data. After reaching *Elimination Start Iteration*, the feature elimination process is applied as long as there exists enough number of features. The feature elimination process stops the elimination when all of the remaining features are significant. The feature elimination process is designed to find out the insignificant features in an iterative way and eliminate them from the meta-model as long as there is a satisfactory change in the meta-model's predictive power.

For measuring the meta-model performance Out of Bag (OOB) score is used. OOB score enables users to assess the predictive power of the model without using any external validation methods. Random Forest model is constructed as an ensemble of a specific number (ntree) many decision trees. When the training set of each tree is constructed by sampling with replacement, roughly one-third of the data is left out of the sample. That portion is named OOB data and used as a test data set to calculate the error. In this process, the same number of OOB data sets as decision trees are created as well. The predictions of OOB data are made by the trained tree, and the mean prediction error is calculated (Mean Squared Error for regression, Classification Error for classification problems) as an estimated test error.

Feature importance scores are obtained by using the variable importance calculation method of Random Forest itself as it is a more efficient and interpretable way than using external variable importance measures. The permutation-based importance measure (Mean Decrease in Accuracy) of Random Forest is used. Mean Decrease in Accuracy is the preferred importance method for regression problems. It is also named as the Percent Increase in MSE (Mean Squared Error). That method uses OOB data to evaluate the importance of features on the accuracy level of the Random Forest model. The calculation starts with computing the MSE using the OOB for each tree. The calculated MSE is stored as a reference value. Afterward, the method continues by permuting each feature value randomly and calculating the new MSE values which are also stored for comparison. Then, the percent increase of MSE is calculated for

each feature as 5.1;

$$\frac{100 \times (MSE_{min} - MSE_i)}{MSE_{min}}. \tag{5.1}$$

In this calculation (5.1) MSE_{min} represents the initial accuracy level calculated before the permutation, and MSE_i represents the accuracy level after the permutation of the respective feature for a single tree. In the end, the average percent increase of MSE level is calculated using the values computed from all trees. It is expected that the accuracy level decreases more when the significant features are permuted. Thus, the average percent increase of MSE is used as an estimator of the feature importance.

There are two main user parameters introduced to the procedure to make elimination decisions. First of all, the proportion of features that are evaluated for elimination is defined by the user. Based on the calculated importance scores the candidate features for elimination are determined and the OOB error of the meta-model is evaluated assuming these features are eliminated. If that assumed elimination does not lead to an adequate change in the OOB error of the Random Forest model a smaller set of the features are evaluated as candidates for elimination at the next iteration. The acceptable change amount of OOB error is defined by a user parameter of the procedure, which is introduced as the *OOB Allowance*. The procedure is designed so that the number of features reviewed for elimination is decreased at each iteration exponentially until a satisfactory change is reached according to (5.2);

$$OOBerror_{iter} < OOBerror_{iter-1} \times (1 + o)\,. \tag{5.2}$$

In (5.2) $OOBerror_{iter-1}$ is the OOB error that is calculated at the previous iteration and $OOBerror_{iter}$ is the OOB error of the previous iteration. The user-defined parameter o represents the acceptable increase rate of OOB error for applying the elimination (i.e., *OOB Allowance*).

The elimination speed of the procedure is determined by another user-defined parameter which is *Elimination Proportion*. As mentioned before, that parameter controls the number of features that are considered for elimination. With an increasing

Elimination Proportion, a larger number of parameters are reviewed for elimination, so the procedure works more aggressively. On the other hand, the procedure terminates quickly without eliminating all insignificant features with a small *Elimination Proportion* value. At each iteration, the number of features that are candidates for elimination decreases exponentially and the calculation of that number is explained in Figure 5.1 in step 5. Let us assume that the procedure works for an agent-based model with 10 parameters, which means the initial random forest meta-model has 10 features at the beginning. Different *Elimination Proportion* values lead to elimination considerations as Table 5.1.

Table 5.1. Number of features that are reviewed for elimination at each iteration with 10 features at the beginning and different *Elimination Proportion* values.

		Elimination Iteration Number					
		1	**2**	**3**	**4**	**5**	**6**
Elimination Proportion	0.2	2	0	0	0	0	0
	0.5	5	2	1	0	0	0
	0.8	8	6	4	3	2	1

In Figure 5.1 feature elimination process of the previously proposed procedure is summarized. While the speed of feature elimination is controlled by the *Elimination Proportion*, the decision of elimination is determined by the *OOB Allowance* value. Thus, for the procedure to work efficiently and effectively, the selection of these values is crucial. That makes usage of the procedure harder as it needs lots of prework and experiments to define these values. To prevent this requirement, a simplified approach is designed in this study.

5.2. Improved Feature Elimination Process

In this study, the previously proposed feature elimination procedure is aimed to be revised in a more user-friendly way. Instead of defining the *Elimination Propor-*

tion at the beginning and selecting a subset of the meta-model features as elimination candidates, all of the features are evaluated upon their own feature importance values in the improved version of the feature elimination process. Meta-model features that are under a predefined importance value are eliminated, and the meta-model is trained without the eliminated features in the next iteration. The importance values are calculated by Random Forest itself (as mentioned in the previous section) and so no external importance calculation is introduced into the procedure. In the following iterations, the meta-model is trained without the eliminated features and feature importance values are recalculated. Elimination and training continue until the number of iterations reaches the *Iteration Budget* if the number of remaining features is enough for elimination (greater than or equal to 2).

Given that $iter^* \leq iter$ and $2 \leq k$;

1: Calculate permutation-based importance scores if RFE is used.
Otherwise, calculate only when $iter = iter^*$;
2: Set $elim\_iter$ to 1;
3: $check\_elim \leftarrow TRUE$;
4: **while** $check\_elim$ **do**
5: Calculate the elimination fraction by using Eq. 5.3, and the number of features considered for elimination by using Eq. 5.4;

$$p^* = p^{elim\_iter} \tag{5.3}$$

$$k' = \lfloor k_{iter} * p^* \rfloor \tag{5.4}$$

, where p is a predefined fraction determining the maximum number of features that can be eliminated at once and k_{iter} is the number of features at $iter$.
6: Train M on T without k' features having the lowest importance scores;
7: **if** Cond. 5.2 holds **then**
8: Update k_{iter} by eliminating the selected k' features from the feature set;
9: $check\_elim \leftarrow FALSE$;
10: **else**
11: **if** all features are significant **then**
12: $check\_elim \leftarrow FALSE$;
13: **else**
14: $elim\_iter = elim\_iter + 1$;
15: **end if**
16: **end if**
17: **end while**
18: Continue with the adaptive sampling procedure.

Figure 5.1. Feature elimination process adapted from the previously proposed procedure where iter*= *Elimination Start Iteration*.

The improved design removes the need to re-evaluate the meta-model's OOB error rate after possible elimination. As a result, the requirement for the *OOB Allowance* user parameter of the procedure is removed as well. The feature elimination decisions are made upon features' own importance values within the existing version of the meta-model at the current iteration. The comparison is made between the current importance values and the pre-defined importance threshold, and the features are chosen for elimination according to that comparison. Moreover, in the previous feature elimination process design, the meta-model is retrained unnecessarily within iterations even if no features are eliminated. The process is designed in that way to enable OOB error comparison. In the improved version, the unnecessary training operations are also eliminated.

In Figure 5.2 feature elimination process of the improved procedure is summarized. The new elimination process is easier to apply than the previous version as two user-defined procedure parameters are removed. Moreover, the newly introduced parameter of *Feature Importance Threshold* is much more interpretable than the removed parameters of *Elimination Proportion* and *OOB Allowance*. Users can easily understand that the procedure tends to eliminate more meta-model features if the *Feature Importance Threshold* is set to higher values and keep only the features having very high importance on the meta-model output. Whereas if the *Feature Importance Threshold* is set to lower values, it will be harder to eliminate meta-model features for the procedure, and only the least important meta-model features will be eliminated from the feature set. As a result, that change makes parameter tuning easier for the elimination process as the number of parameters is decreased and the introduced parameter is so straightforward.

Given that $iter^* \leq iter$ and $2 \leq k$;

Where $iter^*$ is the iteration number defined for starting the elimination process, iter is the current iteration number and k is the number of metamodel features at the beginning.

1: **while** iter ≤ iteration budget **do**
2: Train the metamodel on the training data with the current feature set;
3: **if** $iter^* \leq iter$ and $2 \leq k_{iter}$ **then**;
Where k_{iter} is the number of metamodel features at the current iteration.
4: Calculate permutation-based importance scores;
5: Choose the features with an importance score less than the pre-defined feature importance threshold. Let's denote that subset of features as k′.
6: **if** k′ > 0 **then**;
7: Train the metamodel on the training data without k′ features;
8: Update k_{iter} by eliminating the selected k′ features from the current feature set;
9: **else if** continue without making any elimination;
10: **end if**
11: **end if**
12: Continue with the adaptive sampling procedure
13: **end while**

Figure 5.2. Feature elimination process of the improved procedure.

5.3. Improvement Opportunities on the Proposed Approach

The efficiency and accuracy of the proposed approach are highly dependent on *Elimination Start Iteration* where the feature elimination procedure starts and the *Iteration Budget* until which the features are re-evaluated for elimination decisions and the meta-model continues to be trained. The adaptive sampling process continues until *Elimination Start Iteration* to expand the training data set and to enable the improvement of the meta-model. Selecting the right *Elimination Start Iteration* is important for efficiency because starting the elimination later in the procedure means taking less advantage of the simpler meta-model and tackling more with the high dimensional meta-model. However, starting to feature the elimination process too

early means to take elimination decisions with a less saturated model and a narrower training data set. Thus, a comparison between different values of *Elimination Start Iteration* is made in this section.

After starting the feature elimination process, the meta-model continues to be trained without the eliminated features (which also means with only the important features). Thus, the *Iteration Budget* directly affects the predictive power of the final meta-model. The *Iteration Budget* should be selected large enough to advance the meta-model with lowered dimensions and the expanded training data set. However, it should be small enough to prevent lingering at the same accuracy level without any improvements despite training to ensure efficiency. In order to manage that trade-off, the user of the procedure is expected to make experiments to define the proper *Iteration Budget*. To reduce the dependency of the proposed approach's performance on the user-defined parameters, alternative designs that eliminate the *Iteration Budget* parameter are considered in this section.

As stated before, *Elimination Start Iteration* is a user-defined procedure parameter that should be selected according to the selected ABM. To assess feature elimination starting points against the sample ABM that is used in this thesis, 7 alternative values are selected, and 10 distinct repetitions of the procedure are generated by using these values (Selected values are 2,3,4,5,6,7,8). For each repetition of the procedure, the final meta-model RMSE values for different *Elimination Start Iteration* values are compared. The minimum RMSE value and the *Elimination Start Iteration* values providing the minimum RMSE value are recorded. The RMSE value comparisons can be found in Table 5.2.

Table 5.2. RMSE values of the final meta-models using different *Elimination Start Iteration* values.

Procedure Repetitions	**Elimination Start Iteration**							**Min RMSE**	**Selected Start Point**
	2	**3**	**4**	**5**	**6**	**7**	**8**		
1	6.35	6.44	6	6.26	6.37	6.32	6.2	6	4
2	6.2	6.02	6.21	6.14	6.1	6.17	5.94	5.94	8
3	5.91	5.95	6.02	6.02	6.07	6.11	5.96	5.91	2
4	5.96	6.01	6.12	6.05	6.03	6.02	6.16	5.96	2
5	6.31	6.42	6.19	6.2	6.27	6.58	6.47	6.19	4
6	6.41	6.15	6.3	6.15	6.46	6.34	6.36	6.15	3
7	6.39	6.08	6.23	6.54	6.53	6.52	6.41	6.08	3
8	6.08	6.3	6.19	6.31	6.21	6.26	5.89	5.89	8
9	6.1	6.18	6.38	6.34	6.25	5.88	5.84	5.84	8
10	6.3	6.39	6.17	6.33	5.91	5.97	6.32	5.91	6

In order to determine the best *Elimination Start Iteration*, it is examined how many times each *Elimination Start Iteration* value gives the lowest RMSE value during the repetitions. Moreover, to see if there exists any completion time improvement, the average duration of the procedure (which is estimated at 10 repetitions) is compared for each *Elimination Start Iteration* value. These comparisons can be found in Table 5.3.

Table 5.3. Comparison of *Elimination Start Iteration* values for providing the best RMSE and procedure completion times for 10 repetitions.

	Elimination Start Iteration						
	2	**3**	**4**	**5**	**6**	**7**	**8**
How Many Times It's the Best?	2	2	2	0	1	0	3
Total Duration of 10 Repetitions	5:58:14	5:22:53	5:48:26	5:26:08	5:45:47	6:12:22	5:33:43
Average Duration of 1 Repetition	35:49	32:17	34:51	32:37	34:35	37:14	33:22

The change in the average duration of the procedure seems random. It can also be seen in Figure 5.3 that; the average completion time fluctuates while starting the feature elimination process in the later iterations. It decreases when the starting iteration is changed to 3 from 2, but it increases back when starting to the elimination at iteration number 4. The same fluctuation is seen in the experiments that are conducted with higher elimination starting points. Thus, it is concluded that changing the *Elimination Start Iteration* value does not have a significant effect on the procedure duration.

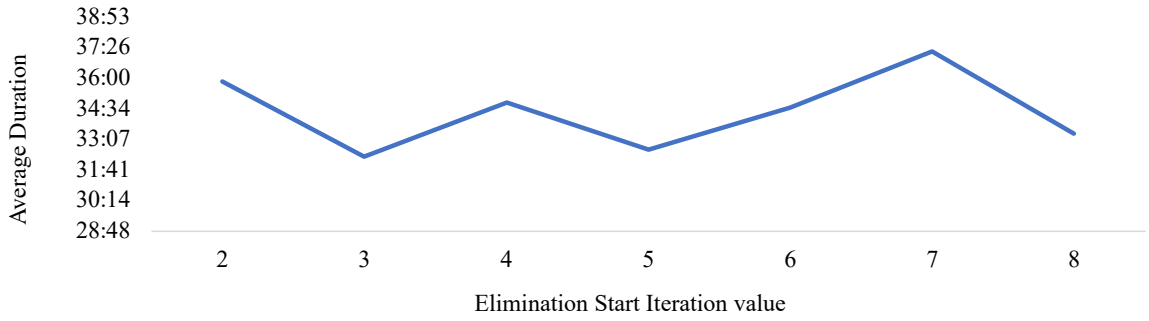

Figure 5.3. Average duration change over changing *Elimination Start Iteration*.

Despite showing no specific effect on procedure duration, using different *Elimination Start Iteration* values affects the procedure performance in providing the minimum

RMSE value. The lower half of the values (2,3, and 4) and the highest value (8) seem to have outstanding performances. Thus, further experiments are designed to compare the performances of procedures that are starting the elimination process at iterations 2, 5, and 8. In order to conduct more detailed investigations and eliminate the randomness effect, the number of procedure repetitions is increased to 30 in that set of experiments. Similar to the first set of experiments, RMSE values of the final metamodels are compared for each of 30 repetitions. RMSE values are reported in Table 5.4.

Table 5.4. Final RMSE values using different *Elimination Start Iteration* values.

Procedure	**Elimination Start Iteration**			**Min**	**Selected**
Repetitions	**2**	**5**	**8**	**RMSE**	**Start Point**
1	6.25	6.22	6.27	6.22	5
2	6.09	6.26	6.19	6.09	2
3	5.91	6.14	6.1	5.91	2
4	6.14	6.15	6.21	6.14	2
5	6.42	6.22	6.46	6.22	5
6	6.27	6.1	6.45	6.1	5
7	6.15	6.28	6.46	6.15	2
8	5.93	6.35	5.88	5.88	8
9	6.21	6.28	5.96	5.96	8
10	6.41	6.53	6.27	6.27	8
11	6.49	6.26	6.38	6.26	5
12	6.3	5.92	6.04	5.92	5
13	6.18	6.31	6.16	6.16	8
14	5.82	6.21	6.25	5.82	2
15	5.94	6.21	6.37	5.94	2
16	6.7	6.74	6.57	6.57	8
17	6.86	6.61	6.46	6.46	8
18	6.66	6.47	6	6	8

Table 5.4: Final RMSE values using different *Elimination Start Iteration* values. (cont.)

Procedure Repetitions	**Elimination Start Iteration**			**Min RMSE**	**Selected Start Point**
	2	**5**	**8**		
19	6.37	6.25	6.06	6.06	8
20	6.52	6.41	6.08	6.08	8
21	6.45	5.96	6.04	5.96	5
22	6.13	6.16	6.22	6.13	2
23	6.34	6.29	6.47	6.29	5
24	6.26	6.38	6.87	6.26	2
25	6.33	6.5	6.73	6.33	2
26	6.62	6.86	6.74	6.62	2
27	6.27	6.7	6.39	6.27	2
28	6.59	6.42	6.56	6.42	5
29	6	6.03	6.18	6	2
30	5.84	5.99	6.13	5.84	2

Similar to the analysis made for 10 repetitions, different values of *Elimination Start Iteration* are compared in terms of providing the minimum RMSE value of the repetition and average completion time of the procedure. The comparison is summarized in Table 5.5. Results of the average duration are similar to the experiments made with 10 repetitions and more alternative *Elimination Start Iteration* values. The average duration value fluctuates and does not show any specific patterns. Despite the consistency in average duration, the comparison on providing the minimum RMSE value in the repetition results differently. Starting the feature elimination process at the very beginning of the procedure seems to result in better final meta-models in almost half of the repetitions.

Table 5.5. Comparison of *Elimination Start Iteration* values for providing the best RMSE and procedure completion times for 30 repetitions.

	Elimination Start Iteration		
	2	**5**	**8**
How Many Times It's the Best?	13	8	9
Total Duration of 30 Repetitions	18:29:51	16:17:56	17:28:26
Average Duration of 1 Repetition	50:59	37:48	44:51

That is an understandable result as the *Iteration Budget* is not changed between these experiments. The meta-model is trained more times after eliminating the less important features and considering only the significant features. Moreover, the training data set is expanded between iterations as the adaptive sampling procedure continues. Thus, meta-model has been improved more when starting the feature elimination process in earlier iterations. Improvements in the meta-models through iterations can be seen in Figure 5.4.

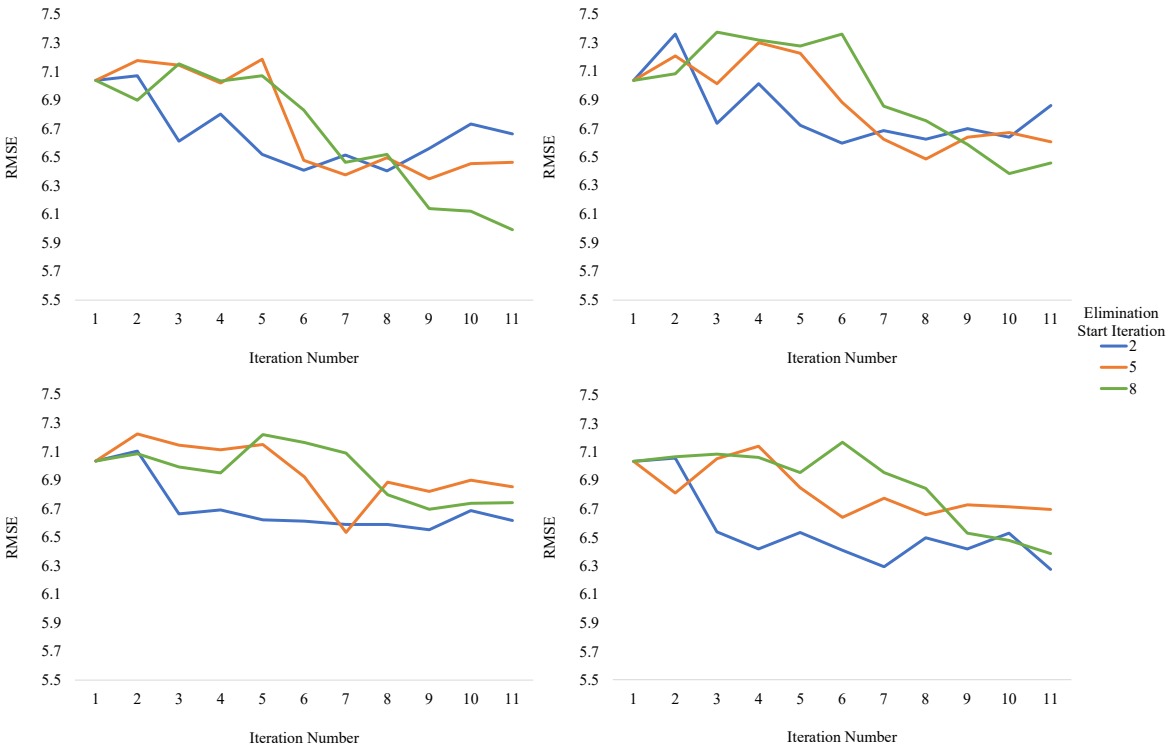

Figure 5.4. Example RMSE values throughout the procedure for different *Elimination Start Iteration* values.

4 examples out of 30 repetitions are chosen and RMSE changes between iterations of these runs are drawn in Figure 5.4. The figure shows that the RMSE value fluctuates throughout the procedure. That means the final meta-model does not necessarily have the lowest RMSE value. A further analysis is made by selecting the meta-model having the minimum RMSE value (within 11 iterations) in 30 repetitions for 3 different *Elimination Start Iteration* values instead of recording only the last meta-model. Table 5.6 shows the minimum RMSE values that are reached in each repetition of the procedure performed with different *Elimination Start Iteration* values. They are not necessarily belonging to the last iteration of the procedure; they can be calculated for any iteration after starting the elimination. The raw data of all RMSE values generated during the experiments can be found in Figure A.1 in Appendix A.

Table 5.6. Minimum RMSE values using different *Elimination Start Iteration* values.

Procedure Repetitions	**Elimination Start Iteration**			**Min RMSE**	**Selected Start Point**
	2	**5**	**8**		
1	6.25	6.1	6.17	6.1	5
2	6.09	6.26	6.18	6.09	2
3	5.78	6.09	6.1	5.78	2
4	6.01	6.15	6.21	6.01	2
5	6.28	6.22	6.38	6.22	5
6	6.24	6.1	6.37	6.1	5
7	6.07	6.28	6.36	6.07	2
8	5.93	6.34	5.87	5.87	8
9	6.16	6.27	5.92	5.92	8
10	6.4	6.46	6.27	6.27	8
11	6.37	6.03	6.26	6.03	5
12	6.22	5.92	6.04	5.92	5
13	6.18	6.25	6.16	6.16	8
14	5.82	6.14	6.25	5.82	2
15	5.92	6.09	6.34	5.92	2
16	6.4	6.48	6.4	6.4	8
17	6.6	6.49	6.39	6.39	8
18	6.41	6.35	6	6	8
19	6.37	6.18	6.06	6.06	8
20	6.39	6.21	6.08	6.08	8
21	6.18	5.96	6.04	5.96	5
22	6.13	6.09	6.22	6.09	5
23	6.25	6.29	6.47	6.25	2
24	6.18	6.38	6.74	6.18	2
25	6.33	6.38	6.59	6.33	2
26	6.56	6.54	6.7	6.54	5
27	6.27	6.64	6.39	6.27	2

Table 5.6: Minimum RMSE values using different *Elimination Start Iteration* values. (cont.)

Procedure Repetitions	**Elimination Start Iteration**			**Min RMSE**	**Selected Start Point**
	2	**5**	**8**		
28	6.37	6.42	6.37	6.37	8
29	6	6.03	6.18	6	2
30	5.84	5.99	6.13	5.84	2

Prior comparison on the number of times for providing the best RMSE is repeated with this scenario as well. As seen in the summary provided in Table 5.7, the comparison result does not change much from the scenario that selects directly the last generated meta-model. The result is consistent with the previous idea that the procedure which starts the elimination process later has less opportunity to improve the meta-model by training with the lower dimension space that consists of significant features. While the procedure has nine iterations ahead to reach its minimum RMSE value when the elimination procedure starts at the second iteration, it has only three iterations to develop an advanced meta-model after feature elimination when the elimination process starts at the eighth iteration.

Table 5.7. Comparison of *Elimination Start Iteration* values for providing the best RMSE for 30 repetitions with the minimum RMSE value observed.

	Elimination Start Iteration		
	2	**5**	**8**
How Many Times It's the Best?	12	8	10

That design is not so advantageous in terms of efficiency as well, because the *Iteration Budget* still needs to be defined by the user and the minimum RMSE value is searched within the predefined number of iterations. In addition to keeping the

Iteration Budget in procedure design, all of the iterations need to be completed to find the minimum RMSE value. So, that change does not provide efficiency improvement. For these reasons, an alternative design is attempted to be built. Instead of completing a specific number of iterations to find the minimum RMSE value, the procedure stops at the first iteration where meta-model RMSE increases and the meta-model of the previous iteration is recorded as the final meta-model. The number of iterations, until which the RMSE increase will be searched, is limited to a static number (11 iterations) to gain insight quickly.

In Table 5.8 the iterations that are selected for stopping the procedure (while using different *Elimination Start Iteration* values) are shown. While procedures starting to feature elimination at iterations 2 and 5 tend to stop earlier than the *Iteration Budget* the ones which are starting to feature elimination at the 8th iteration use nearly the whole *Iteration Budget*. It is reasonable because the RMSE increase is seen within a few iterations after stating the feature elimination. In a similar way, the procedures starting the elimination process at the eighth iteration tend to stop at the ninth iteration.

Table 5.8. Stopping iterations using different *Elimination Start Iteration* values.

Procedure Repetitions	**Elimination Start Iteration**		
	2	**5**	**8**
1	4	7	9
2	5	7	9
3	4	8	11
4	3	6	9
5	3	6	9
6	4	7	10
7	4	11	9
8	3	7	10
9	4	6	10
10	3	6	9

Table 5.8: Stopping iterations using different *Elimination Start Iteration* values. (cont.)

Procedure Repetitions	**Elimination Start Iteration**		
	2	**5**	**8**
11	4	6	9
12	4	8	9
13	3	6	11
14	6	6	11
15	4	6	9
16	3	9	10
17	3	8	10
18	3	7	11
19	4	8	11
20	3	6	9
21	4	7	11
22	4	6	9
23	3	6	10
24	5	6	9
25	6	10	9
26	3	7	9
27	4	6	11
28	5	6	10
29	6	7	11
30	3	6	11

The same comparison between *Elimination Start Iteration* values is made in the design of the procedure with dynamic stopping conditions. Table 5.9 shows the RMSE values recorded for the selected meta-models prior to the first RMSE increase. In the same table, the minimum RMSE value obtained in respective repetition among three alternative feature elimination starting points is also provided to evaluate which one

performs better in terms of accuracy.

Table 5.9. RMSE values recorded using different *Elimination Start Iteration* values.

Procedure Repetitions	**Elimination Start Iteration**			**Min RMSE**	**Selected Start Point**
	2	**5**	**8**		
1	6.56	6.21	6.17	6.17	8
2	6.5	6.3	6.18	6.18	8
3	5.94	6.19	6.1	5.94	2
4	6.25	6.23	6.21	6.21	8
5	6.51	6.71	6.38	6.38	8
6	6.63	6.31	6.37	6.31	5
7	6.39	6.28	6.36	6.28	5
8	6.51	6.34	5.87	5.87	8
9	6.26	6.41	5.92	5.92	8
10	6.54	6.47	6.34	6.34	8
11	6.48	6.14	6.26	6.14	5
12	6.22	6.12	6.33	6.12	5
13	6.5	6.4	6.16	6.16	8
14	6.1	6.54	6.25	6.1	2
15	6.18	6.4	6.34	6.18	2
16	6.81	6.48	6.4	6.4	8
17	6.74	6.49	6.39	6.39	8
18	6.61	6.38	6	6	8
19	6.61	6.31	6.06	6.06	8
20	6.53	6.32	6.19	6.19	8
21	6.18	6.23	6.04	6.04	8
22	6.28	6.2	6.34	6.2	5
23	6.47	6.36	6.47	6.36	5
24	6.29	6.64	6.82	6.29	2
25	6.4	6.38	6.59	6.38	5

Table 5.9: RMSE values recorded using different *Elimination Start Iteration* values. (cont.)

Procedure Repetitions	**Elimination Start Iteration**			**Min RMSE**	**Selected Start Point**
	2	**5**	**8**		
26	6.66	6.54	6.7	6.54	5
27	6.42	6.64	6.39	6.39	8
28	6.37	6.51	6.37	6.37	8
29	6.38	6.61	6.18	6.18	8
30	6.6	6.68	6.13	6.13	8

In that procedure design, the comparison between *Elimination Start Iteration* values results in a different way than completing a predefined number of iterations. Table 5.10 shows the number of repetitions for which the respective feature elimination starting points provide the minimum RMSE value. As the procedure stops at the first increase of RMSE value, the procedure starting the feature elimination at the second iteration does not have time to improve the meta-model. Nevertheless, the procedure starting the feature elimination at the eighth iteration is already trained until the feature elimination process and gained a better predictive power. Thus, starting the feature elimination process later performs better in that design. However, starting the feature elimination process in the eighth iteration results in stopping at the ninth iteration for half of the repetitions. So, that means the advantage of the feature elimination is not utilized well. That trade-off should be considered when the design change is assessed. The raw data of all RMSE values recorded during the analysis and comparisons can be found in Figure B.1 in Appendix B.

Table 5.10. Comparison of *Elimination Start Iteration* values for providing the best RMSE value when stopping at the previous iteration of RMSE increase.

	Elimination Start Iteration		
	2	**5**	**8**
How Many Times It's the Best?	4	8	18

To gain a comprehensive interpretation, the difference between RMSE values obtained from three designs is evaluated. Table 5.11 shows the average differences that are calculated from 30 distinct repetitions for three designs of the procedure that are starting the feature elimination process at stated iterations in rows of the table. The designs are indicated in the table as "*Last Iteration*", "*Minimum RMSE*" and "*Chosen Iteration*". "*Last Iteration*" denotes the design that records the final meta-model within the *Iteration Budget*. The procedure gets user-defined parameters of *Elimination Start Iteration* and *Iteration Budget* and completes all iterations until reaching the budget. "*Minimum RMSE*" denotes the design that chooses the minimum RMSE within the *Iteration Budget*. The procedure gets user-defined parameters of *Elimination Start Iteration* and *Iteration Budget* and completes all iterations until reaching the budget. "*Chosen Iteration*" denotes the design that seeks the first RMSE increase and uses the meta-model of the previous iteration as the final meta-model. The procedure gets user-defined parameters of *Elimination Start Iteration* only. To compare that design with alternatives, the number of iterations to search the RMSE increase is limited to a specific number that is equal to the *Iteration Budget* value that is used in other designs.

Table 5.11. The average RMSE differences between procedure designs.

Elimination Start Iteration	**Chosen Iteration Compared to Last Iteration**	**Minimum RMSE Compared to Chosen Iteration**
2	2.5%	-3.6%
5	1.5%	-2.4%
8	-0.3%	-0.3%

In Table 5.11, the column named "Chosen Iteration Compared to Last Iteration" shows the average percent difference between *Chosen Iteration* design over the *Last Iteration* design. It shows that the *Last Iteration* design performed better when starting the elimination process earlier. However, *Chosen Iteration* design performed slightly better when starting the elimination process later. Furthermore, the column named "Minimum RMSE Compared to Chosen Iteration" shows the average percent difference between the *Minimum RMSE* design over the *Chosen Iteration* design. It is seen that on average 2% of RMSE improvement opportunity is missed when the procedure stops at the first increase of RMSE value. The missed opportunity is higher for the procedures starting the feature eliminations earlier. As a result of these analyses, to utilize the advantages of the feature elimination process, and not miss the opportunity to improve the predictive power of the meta-model it is preferred to continue with the current design of the process and start the feature elimination process in the middle of the procedure. Hereby, in this chapter, an intuitive design choice of the previous approach is validated through detailed experimental analysis.

6. COMPARATIVE PERFORMANCE ANALYSIS

Experiments are planned in two sets to select the improved procedure's feature importance parameter value and compare the performance of the improved procedure with the previously proposed version. The first set of experiments is conducted to find the proper *Feature Importance Threshold* value to be used in the improved model with the selected agent-based model. Different values of *Feature Importance Threshold* values are compared with each other in multiple runs. After choosing the *Feature Importance Threshold*, the second set of experiments is conducted to compare the improved version and the previously proposed version in terms of accuracy and completion time.

Fränken and Pilditch's agent-based model, "Cascades Across Networks are Sufficient for the Formation of Echo Chambers" is used for the experiments [13]. Details of the agent-based model are explained in chapter 4.3. The output of interest is selected as the average proportion of agents having the same opinion within neighborhoods (linked agents). NetLogo version 6.0.4 is used for the agent-based simulation runs. R is used for meta-modeling.

6.1. Data Generation

As the procedure is applied to an agent-based simulation model, the instances of training and test data set consist of the parameter combinations of the agent-based model and labels are the simulation model's output values. These instances are selected using the Latin hypercube sampling method. The details about Latin hypercube sampling can be found in chapter 4.1.2. 10 different training data sets are generated with 500 instances in each to construct robust performance evaluation experiments. The sample pool is refreshed in each iteration of the procedure, and 30 new unlabeled instances are generated using the Latin hypercube sampling method. 5 instances are selected among these 30 new instances and added to the training set to enhance the training set through the procedure and improve the meta-model. Moreover, a test set

having 215 instances is generated with the same approach. Labels of the training and test instances (outputs of agent-based simulation model parameter combinations) are gathered through simulation runs. To consider the randomness, the simulation runs are replicated 30 times for each instance (i.e., agent-based simulation model parameter combinations). The outputs of 30 simulation runs are averaged and recorded as the label of the respective instance.

Before starting the data generation phase, the agent-based simulation model is analysed through individual runs, and it is seen that the output barely changes after the 30th time step. Hence, simulation runs are set to 30 time steps at maximum to ensure efficiency and prevent time loss.

The data generation approach used in this thesis is the same as the one used in previous studies of "Analysis of Agent-Based Simulation Models Through Meta-modeling." [6] and "Adaptive Sampling with Feature Elimination for Agent-Based Models."[4]. The used data generation approach is discussed and assessed thoroughly in those previous studies.

6.2. Performance Evaluation

As mentioned in the previous section, a test data set is generated to evaluate the performance of the improved and previously proposed versions of the procedure. Root Mean Squared Error (RMSE) is chosen as the performance measure as it is suitable for the selected output of interest (which is continuous) and used in the previously proposed procedure too.

The calculation of RMSE is as 6.1;

$$RMSE = \sqrt{\sum_{i=1}^{N} \frac{(\hat{y}_i - y_i)^2}{N}}. \tag{6.1}$$

In this calculation (6.1) i represents the individual instances (a specific combination of agent-based model parameters). $\hat{y}_i$ is the predicted value of the output of interest which is calculated by the meta-model. y_i is the real value of the output of interest

which is obtained by the simulation runs. The prediction performance is calculated using the square root of the mean squared difference between the real and the predicted values where N is the number of instances in the data set.

6.3. Selection of the Feature Importance Threshold

Before starting to compare the previously proposed version with the improved version, the most proper value of the newly introduced user-defined procedure parameter is searched for the selected agent-based simulation model. The *Feature Importance Threshold* defines the procedure's tendency when taking elimination decisions (whether to keep only the most important meta-model features or eliminate only the less important ones). The proper value depends on the agent-based simulation model because the meta-model features are the parameters of the original agent-based model and feature importances are calculated according to the importance of these parameters on the selected output of interest.

To find the value to use, 4 levels of the *Feature Importance Threshold* values are evaluated. The improved version of the procedure is applied 10 times to a parameter combination of 500 instances for each of those 4 levels. Other parameters of the procedure are kept as same during the experiments.

The definitions of these parameters are;

- *Agent-Based Simulation Run Repetitions*: Number of agent-based simulation runs to get the label of an instance (agent-based model parameter combination)
- *Iteration Budget*: Number of iterations to complete before stopping the procedure
- *Number of Instances in Training Data Set*: Number of the agent-based model parameter combinations in the training data set
- *Number of Instances in Test Data Set*: Number of the agent-based model parameter combinations in the test data set
- *Number of New Instances Added in Each Iteration*: Number of the agent-based model parameter combinations added to the training data set at each iteration

- *Elimination Start Iteration*: The iteration number that the feature elimination process is activated
- *Feature Importance Threshold*: The threshold at which features with a lower importance value will be eliminated

Used values for these parameters are summarized in Table 6.1. *Iteration Budget* and *Elimination Start Iteration* parameter values are used as they are in the previously proposed procedure. These values are also discussed in Yurdadön's study in detail [4]. Other procedure parameters, except the *Feature Importance Threshold*, are determined considering the number of meta-model features (agent-based model parameters).

Table 6.1. Parameter values used in the experiments for selecting the *Feature Importance Threshold*.

Parameter	**Value**
Agent-Based Simulation Run Repetitions	30
Iteration Budget	11
Number of Instances in Training Data Set	500
Number of Instances in Test Data Set	125
Number of New Instances Added in Each Iteration	5
Elimination Start Iteration	5
Feature Importance Threshold	0.4 / 0.25 / 0.1 / 0.05

The boxplot in Figure 6.1 shows the distribution of RMSE scores at the end of the improved procedure in the case of using different levels of *Feature Importance Threshold* Values. Results of 10 repetitions are plotted in the boxplot.

There seem no outliers and the distributions seem suitable to draw interpretation. The experiments made with 0.25 and 0.4 have taller boxplots than 0.05 and 0.1 *Feature Importance Threshold* values. It can be understood that the procedure performance is affected by the randomness more while using the higher values. Moreover, the RMSE distributions are set to higher (both for the median and maximum points) values for 0.25 and 0.4 values. Thus, lower *Feature Importance Threshold* values seem to be more suitable for the selected agent-based simulation model.

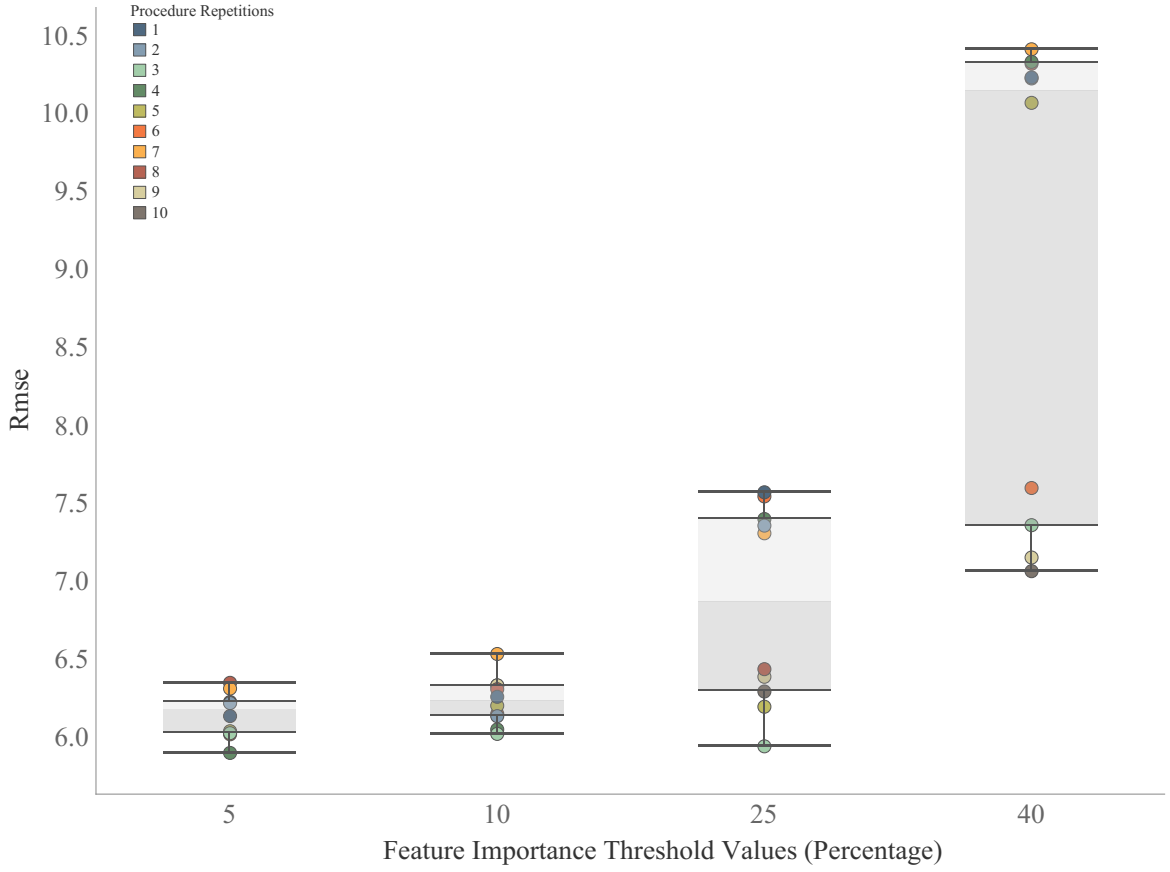

Figure 6.1. Boxplot of RMSE scores of final meta-models with different *Feature Importance Threshold* values.

Figure 6.2 shows the change of average RMSE values for 10 replications of the improved procedure with different *Feature Importance Threshold* values during the procedure iterations. The procedures with different *Feature Importance Threshold* values show similar accuracy performances before starting the eliminations (5th iteration), so the final difference is not related to the initial training set or sampling process. The

RMSE increase after starting the elimination process indicates that two higher values of the parameter (0.25 and 0.4) result in the less accurate performance of the procedure. Similar to boxplot results, 0.05 and 0.1 values of the parameter show similar performances.

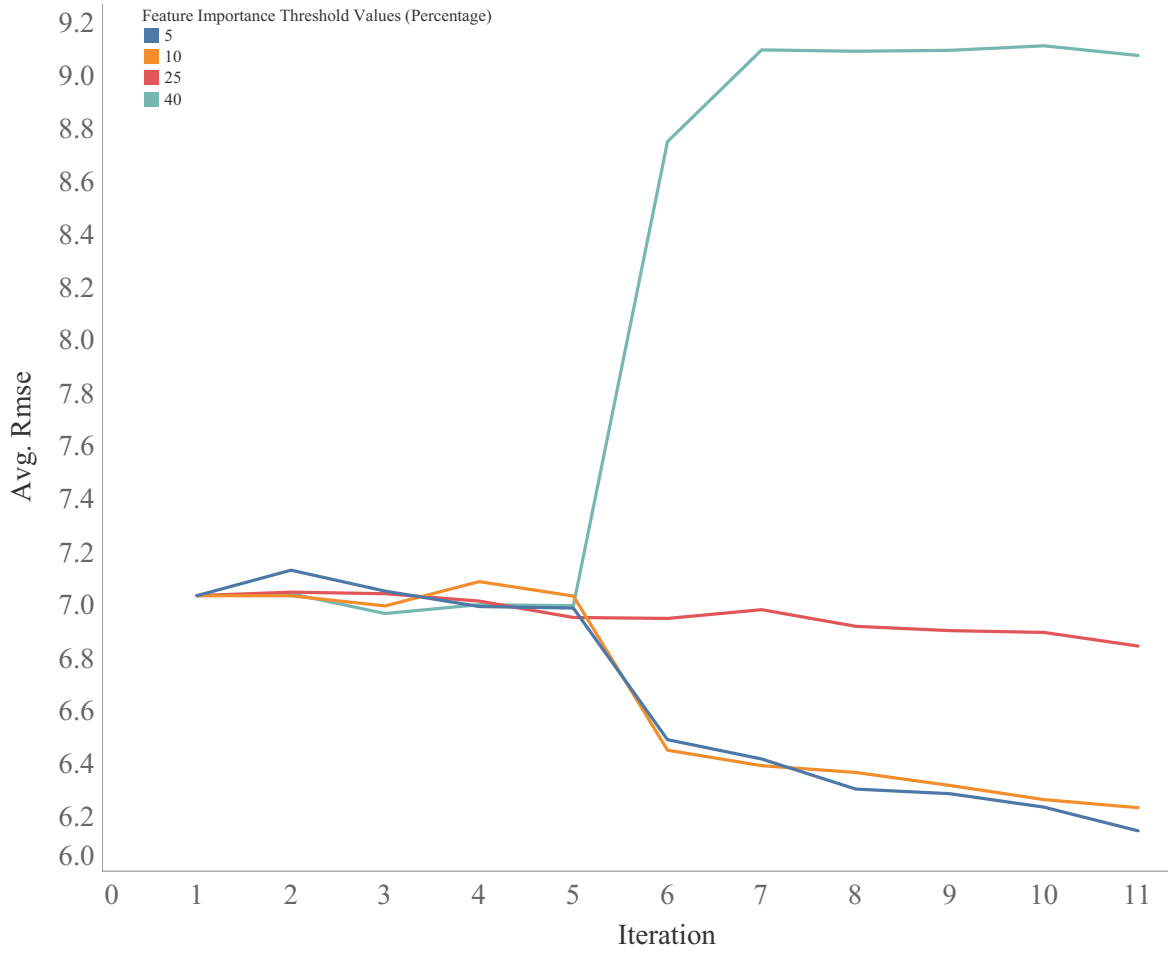

Figure 6.2. Average RMSE scores of meta-models with different *Feature Importance Threshold* values through iterations.

Features of the meta-model (parameters of the agent-based simulation model) are as follows;

- *Max-Links*: The parameter defines the size of each agent's network by determining the number of links that an agent forms with its neighbors.
- *Evidence*: In the case of Reinforcement Learning method activation, that parameter is used to define the number of sampling processes for agents. Reinforcement Learning is not activated in the agent-based simulation model, so that parameter

is used as a dummy parameter in this study.

- *Sc-Bel-Prop*: The parameter is introduced to the agent-based simulation model for theory validation. It is used for analyzing the individual agent sensitivity to believe under conformity motivations.
- *Prop-Likelihood*: The parameter defines the probability of opinion declaration for agents.
- *N-Init-Believers*: The value of the parameter is used as the number of agents that share their beliefs at the beginning of the simulation run. Opinions spread from that specific number of agents to the population.
- *Prior-Mean*: Mean value used for the distribution of prior belief, expertise, and trustworthiness levels.
- *Prior-Sd*: Standard deviation value used for the distribution of prior belief, expertise, and trustworthiness levels.
- *Expertise-Influence*: The parameter determines how the expertise level of the source affects opinion formation on the target's side.

The features that are eliminated from the meta-model are summarized in Table 6.2. The table consists of the features that are eliminated at least once in any of the 10 repetitions of the procedure with respective *Feature Importance Threshold* values. The procedure tends to eliminate more features when higher threshold values are used as it keeps only the features that have higher importance values than the threshold. However, these values result in lower accuracy levels. Besides, the procedure eliminates the same features from the meta-model when 0.05 and 0.1 threshold values are used.

Table 6.2. Eliminated features at different values of *Feature Importance Threshold*.

	Feature Importance Threshold Values			
	0.4	**0.25**	**0.1**	**0.05**
Eliminated Features	Evidence	Evidence	Evidence	Evidence
	Expertise-Influence	Max-Links	Max-Links	Max-Links
	Max-Links	Prior-Mean	Prop-Likelihood	Prop-Likelihood
	N-Init-Believers	Prop-Likelihood	Sc-Bel-Prop	Sc-Bel-Prop
	Prior-Mean	Sc-Bel-Prop		
	Prop-Likelihood			
	Sc-Bel-Prop			

To decide whether to use a 0.05 or 0.1 threshold value, the feature elimination process is investigated in detail. For each of the 10 replications of the procedure, both 0.05 and 0.1 thresholds eliminate the same features at the end of the process. However, the procedure cannot eliminate all of those 4 features immediately when a 0.05 threshold value is used. That is because some features like *Max-Links* have an importance value that is on the borderline. Thus, the procedure works in a more robust way using a 0.1 *Feature Importance Threshold* value with that specific agent-based simulation model. In order to ensure the elimination of these insignificant features from the meta-model, a 0.1 *Feature Importance Threshold* value is used for the rest of this study.

6.4. Validation of Feature Elimination Decisions

The eliminated and kept features of meta-model (parameters of the agent-based model) are also investigated through behavior space experiments of NetLogo. The output of interest is selected as *cl-prop-same*, and it shows the average proportion of agents having the same opinion within linked agents. Experiments are made to see the interaction between that output and agent-based model parameters. In order to

eliminate the number of agent effects, 3 levels of population size are used through experiments as 1000, 5000, and 10000. Moreover, to consider the randomness, 50 simulation runs are conducted with each agent-based parameter combination.

For the *Evidence* parameter of the agent-based model, 750 simulation runs are conducted with 3 levels of population size (1000, 5000, 10000), 5 levels of *Evidence* parameter (0, 25, 50, 75, 100), and 50 repetitions for each combination. Other parameters of the agent-based model kept as same to see the effect of the parameter on the output. 17 runs out of 750 are ended within 4 time steps on average. The duration of simulation runs is 23 steps for 1000 agents, 46 steps for 5000 agents, and steps 62 for 10000 agents. The average run duration is 44 steps for 750 runs. Thus, the 17 runs that are ended within 4 steps on average are marked as outliers and removed from the analysis. Figures 6.3 and 6.4 shows that changing the *Evidence* parameter has no remarkable effect on the output of interest.

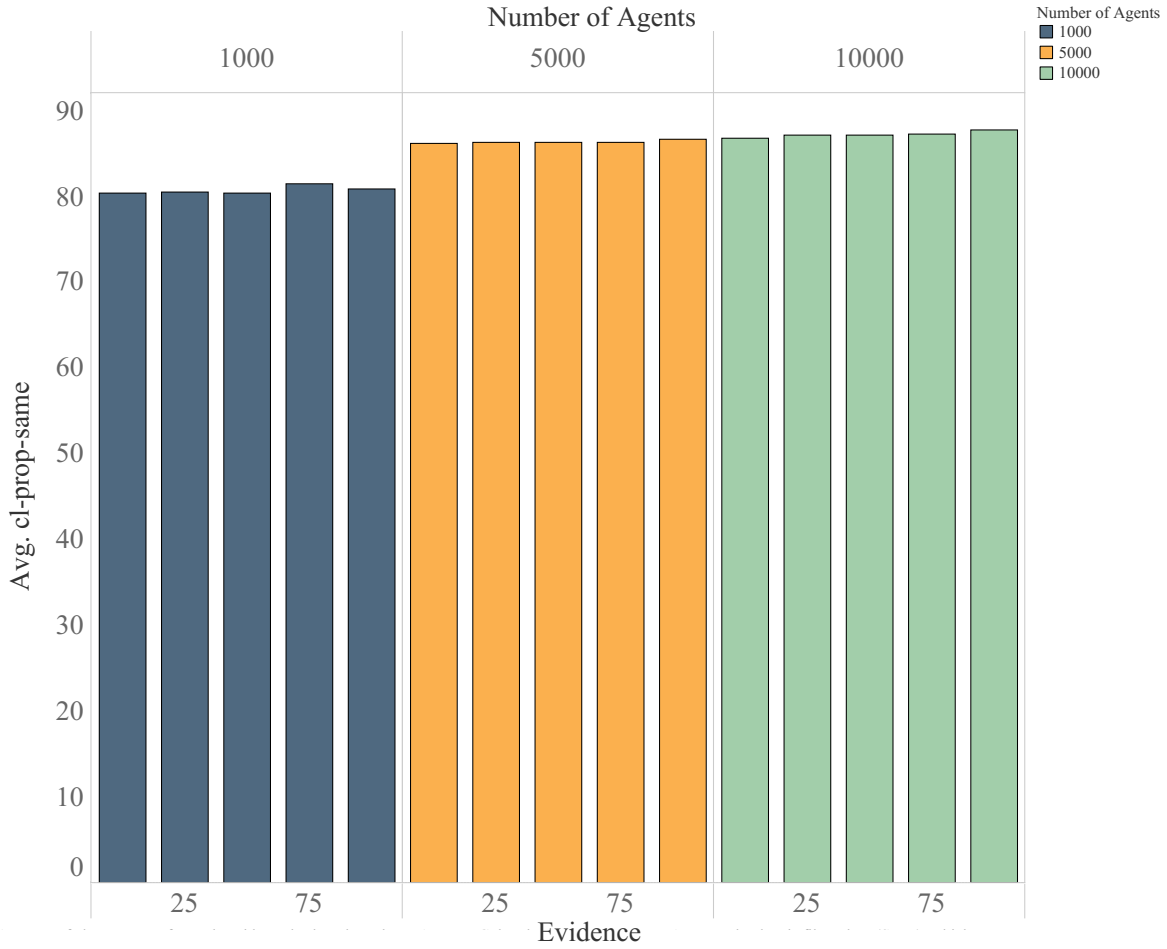

Figure 6.3. Change of average output value at different levels of *Evidence* parameter and changing population sizes (Outlier runs are eliminated.).

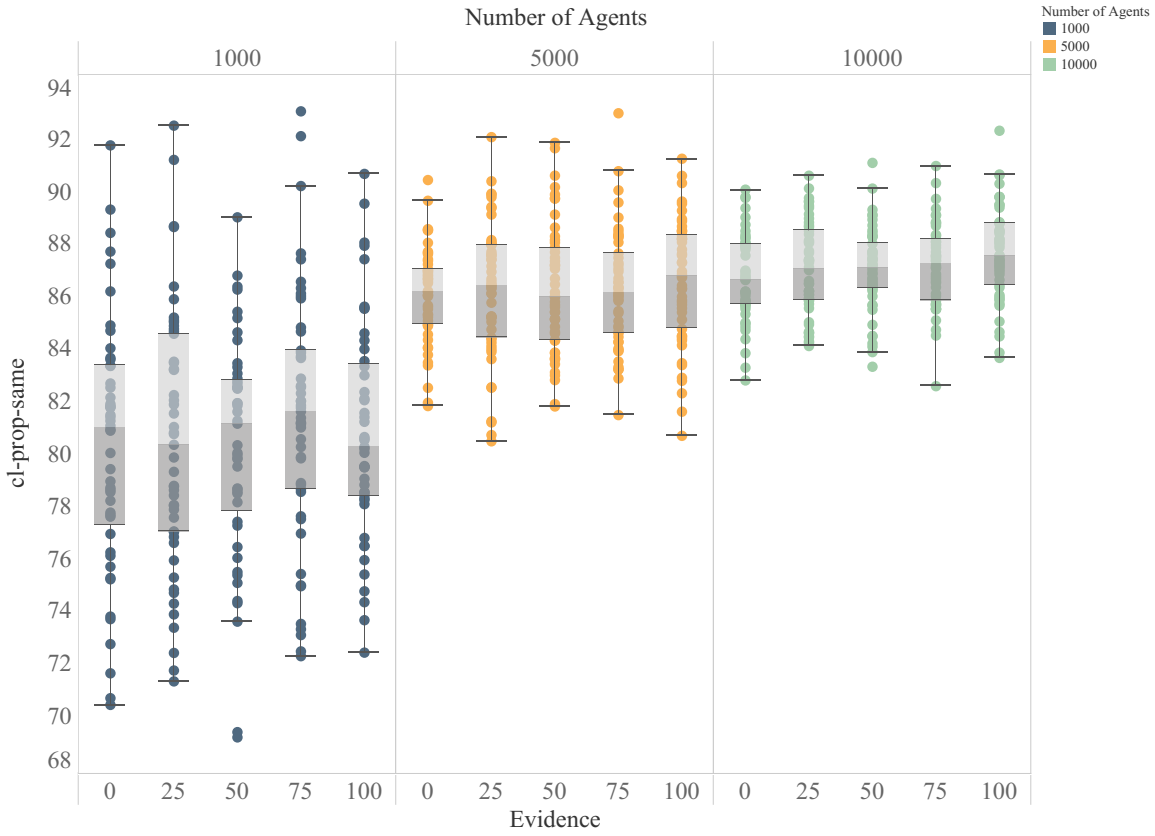

Figure 6.4. Boxplot of output value at different levels of *Evidence* parameter and changing population sizes (Outlier runs are eliminated.).

The effect of the *Max-Links* parameter on the output of interest is also investigated. 50 simulation runs are conducted for 8 levels of *Max-Links*. It is computationally hard to conduct simulation runs with high population size and high values of the parameter as the agent-based model gets too complex. Thus, the experiments are designed with a population of 1000 agents. 50 simulation runs are executed and analysed. The average output of 50 simulation runs of each *Max-Links* value is seen in Figure 6.5. The changing effect is seen with values under 150. The average output value does not change much after that point (the average change is around 1% between the higher values).

However, the procedure eliminates that feature from the meta-model at every *Feature Importance Threshold* value level. To understand the reason behind that decision, the training data reached at the end of the procedure is investigated.

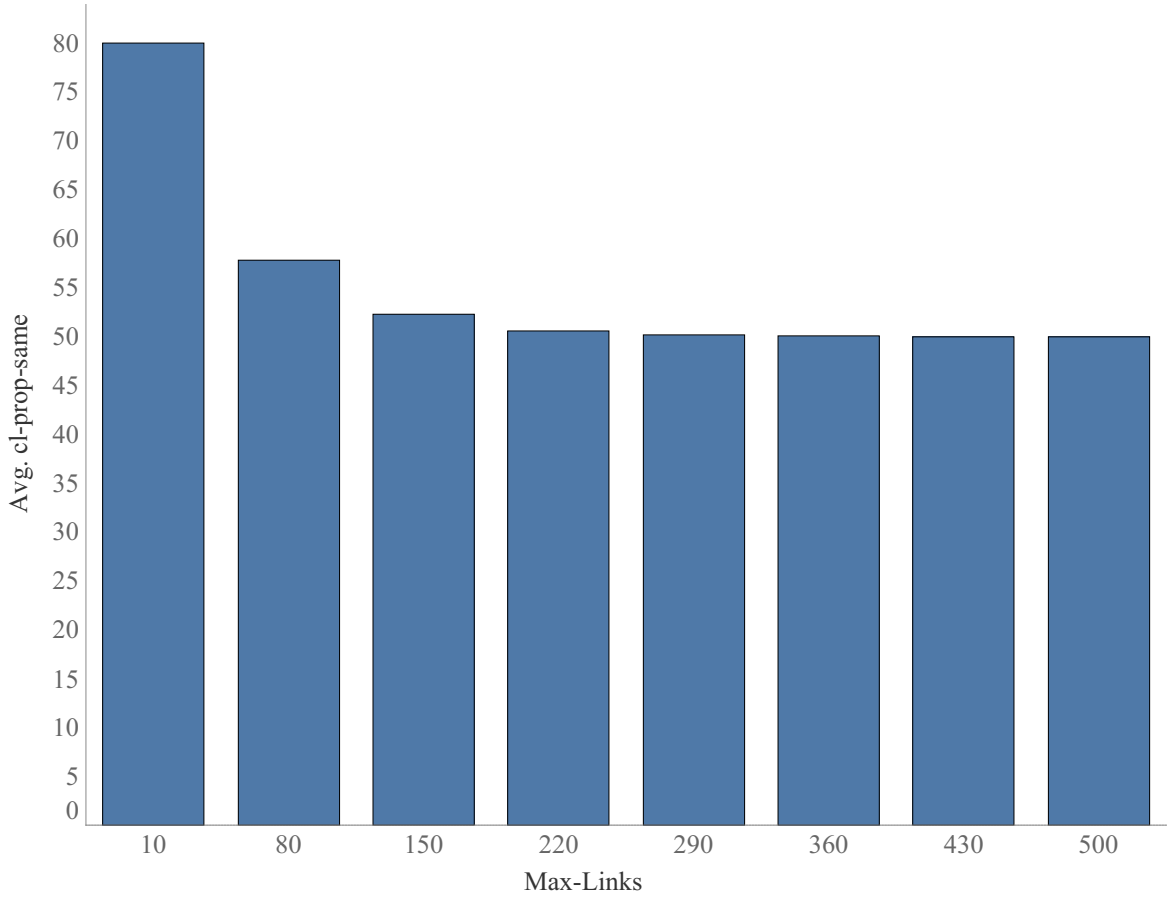

Figure 6.5. Change of average output value at different levels of *Max-Links* parameter (400 runs).

The training data is constructed with 500 initial instances. 5 new instances are added at each iteration of the procedure except the final iteration (for 10 iterations). Thus, at the end of a distinct run of the procedure, the training data size reaches 550 instances. The procedure is replicated 10 times, so the final training data contains 5500 instances. Table 6.3 shows that on average 70% of the training data contains *Max-Links* values over 150 due to the uniform distribution of the feature values over allowable values (from 2 to 500). As the training data contains more instances from the values that do not affect the output much, it is understandable that the procedure selects the feature as insignificant and decides to eliminate it.

Table 6.3. Distribution of *Max-Links* parameter value within the final training data set.

		Max_Links Values (M)			
		M<=80	**80<M<=150**	**150<M<=220**	**220<M**
Replication Number	1	16%	14%	14%	56%
	2	16%	14%	13%	57%
	3	16%	14%	13%	56%
	4	16%	14%	14%	57%
	5	16%	14%	13%	56%
	6	16%	14%	14%	57%
	7	16%	14%	14%	56%
	8	15%	14%	14%	57%
	9	15%	14%	14%	57%
	10	15%	14%	14%	57%

To analyse the effect of the *Prop-Likelihood* parameter of the agent-based model, 900 simulation runs are conducted with 3 levels of population size (1000, 5000, 10000), 6 levels of *Prop-Likelihood* parameter (0, 0.2, 0.4, 0.6, 0.8, 1), and 50 repetitions for each combination. It is understandable that setting the *Prop-Likelihood* parameter to 0 means that agents in the population do not make their opinions public. That makes the opinion sharing and belief updating mechanism stop working.

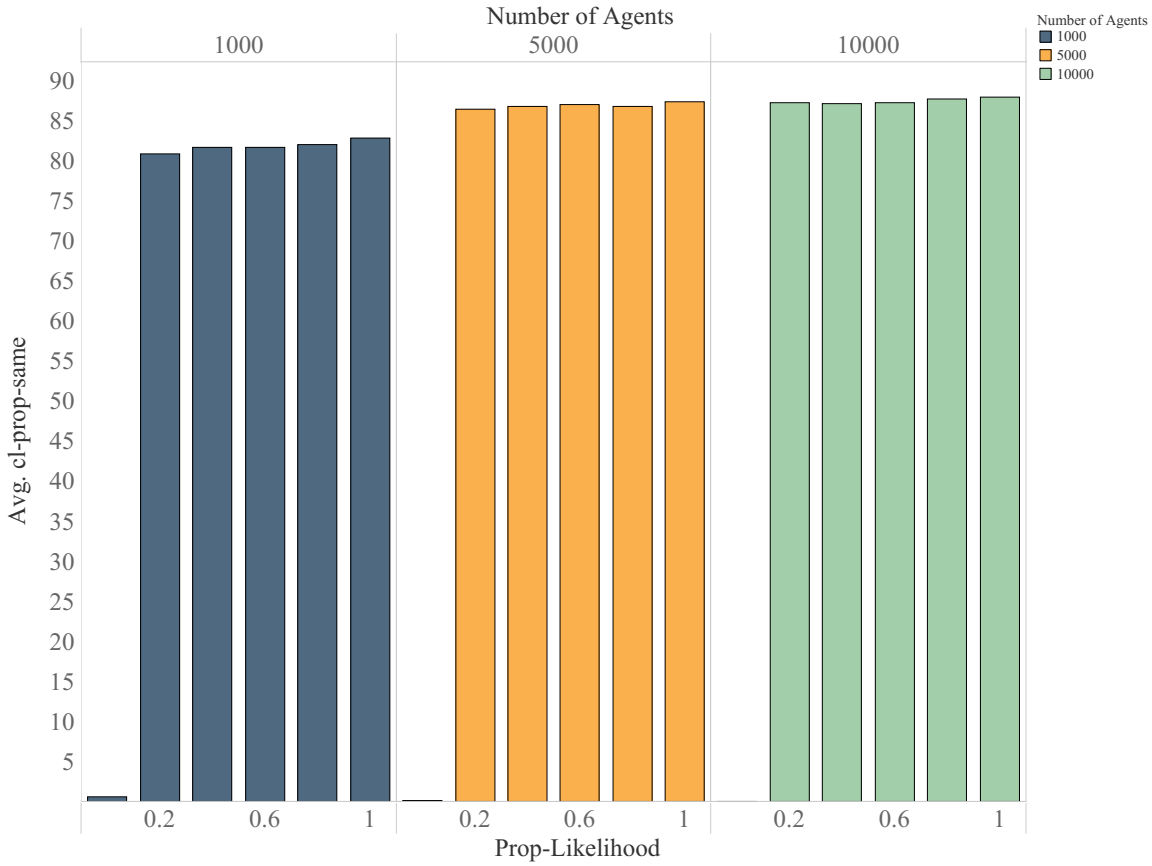

Figure 6.6. Change of average output value at different levels of *Prop-Likelihood* parameter and changing population sizes (900 runs).

That effect is seen in Figure 6.6. Other values of the parameter do not seem to have a remarkable impact on the average output. Similar to the *Max-Links* parameter of the ABM, the instances with 0 *Prop-Likelihood* values constitute a very small portion of uniformly distributed training data and so the procedure finds that parameter as an insignificant one. Figure 6.7 shows the boxplot for changing levels of the *Prop-Likelihood* parameter and population size excluding the 0 value of the parameter. Figure 6.7 also supports the idea that changing the value of the parameter does not affect the output drastically as the distributions and quartiles do not change much within the runs of the same population size. So, the procedure finds that the importance of the related feature does not satisfy the *Feature Importance Threshold* (0.1) and eliminates it from the meta-model.

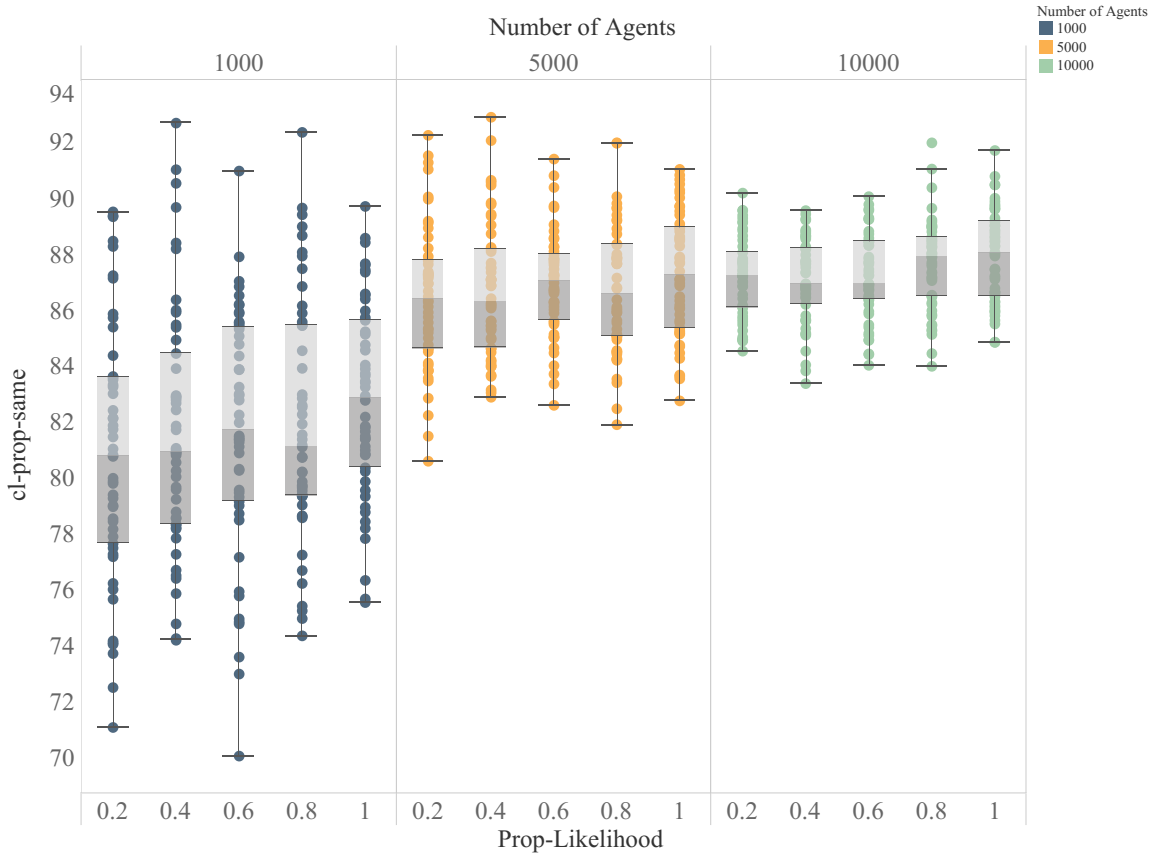

Figure 6.7. Boxplot of output value at different levels of *Prop-Likelihood* parameter and changing population sizes (Simulation runs with 0 value of *Prop-Likelihood* are eliminated.).

In order to evaluate the elimination decisions of the procedure comprehensively, the features that are kept in the meta-model are also investigated. *Prior-Sd* is not eliminated in any meta-model replications of any *Feature Importance Threshold* values. Thus, it is expected to see the changing levels of the *Prior-Sd* parameter have a significant impact on the *cl-prop-same* value of the agent-based simulation runs. To investigate that, 900 simulation runs are conducted with 3 levels of population size (1000, 5000, 10000), 6 levels of *Prior-Sd* parameter (0, 0.2, 0.4, 0.6, 0.8, 1), and 50 repetitions for each combination. 16 of those 900 simulation runs are completed within 4 or 5 time steps. These runs are marked as outliers and removed from the analysis.

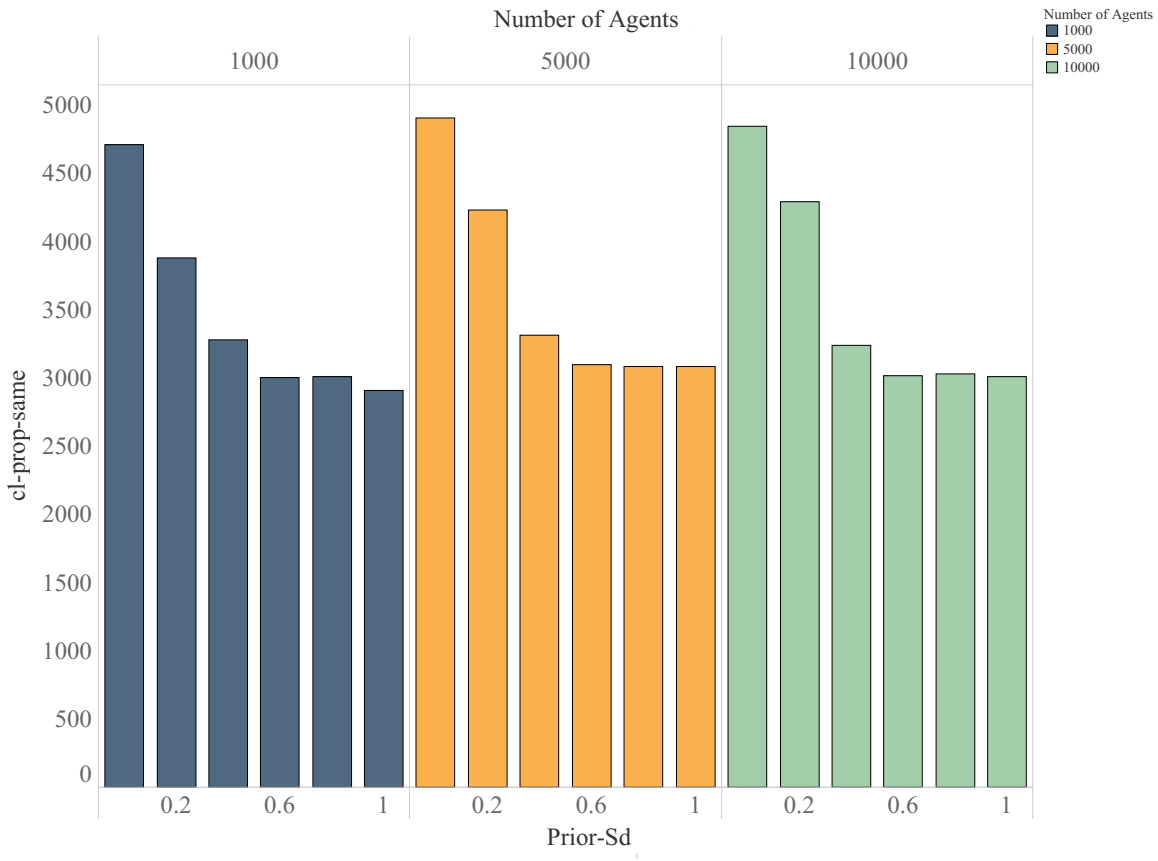

Figure 6.8. Change of average output value at different levels of *Prior-Sd* parameter and changing population sizes (Outlier runs are eliminated.).

Figure 6.8 shows that the changing levels of the *Prior-Sd* parameter affect the average output for each 3 population sizes. The effect is especially remarkable for low levels of the parameter. That is an expected result as the population lost its homogeneity when it is constructed with low standard deviation levels. When the standard deviation increases, the variety of agents in terms of prior belief, trustworthiness, and expertise increases. However, the increase in the variety does not affect the results much after a specific level (0.6 in this case). The boxplot in Figure 6.9 also supports the findings. Observed distributions of the output change dramatically until the 0.6 value and the change slows down after that value.

For that parameter, the value range that has no effect on the output (values above 0.6) has a smaller portion than the value range that has a remarkable impact (contrary to the case of *Max-Links*). Thus, it is understandable that the parameter is found as a significant feature for the meta-model, and the procedure decides to keep

it.

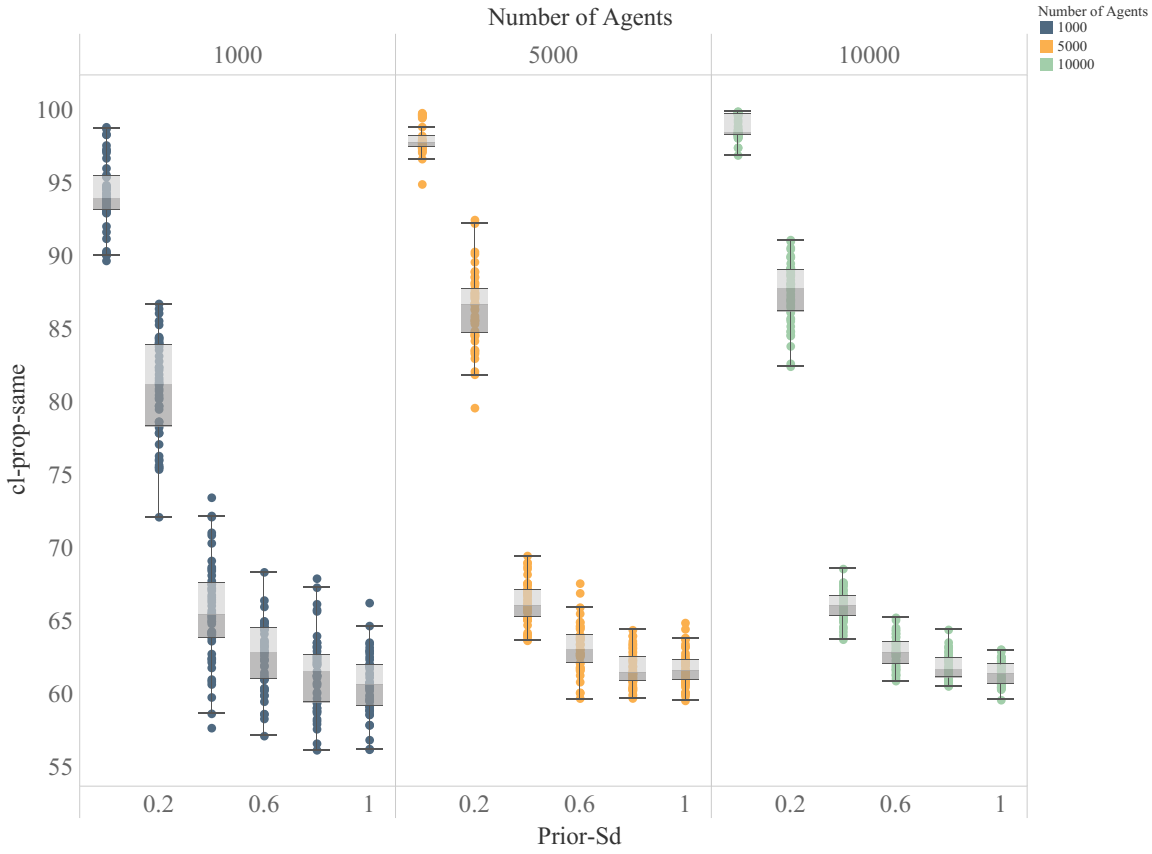

Figure 6.9. Boxplot of output value at different levels of *Prior-Sd* parameter and changing population sizes (Outlier runs are eliminated.).

Elimination decisions of the procedure are assessed through experiments in this chapter. Each feature of the meta-model corresponds to a parameter of the agent-based simulation model. So, experiments on the agent-based simulation model with changing values of model parameters that correspond to eliminated features give insight into the validation of elimination decisions. Besides that, a simulation parameter which is represented by a meta-model feature that is not eliminated in any replications of the procedure is examined to make validate the decisions from the opposite point of view. Experiment sets on eliminated features showed that the elimination decisions of the procedure with a *Feature Importance Threshold* value of 0.1 are valid, and eliminated features are insignificant ones. Moreover, the feature that is not eliminated in any *Feature Importance Threshold* value and any replications is shown to have a remarkable effect on output.

6.5. Comparison of Procedure Versions

As mentioned before, the second set of experiments is designed to compare the improved version and the previously proposed versions of the procedure. To make this comparison, 10 training data sets are generated with a size of 500 instances. Moreover, a test set with a size of 215 instances is generated. Details about data generation and new sample selection (for expanding the training set over iterations) can be found in sections 4.1.2 and 6.1. Each version of the procedure is performed with 10 different training data sets for 10 distinct repetitions. So, a workflow of 11 iterations (the predefined iteration budget) is completed 100 times. The design of experiments is illustrated in Figure C.1 in Appendix C. Throughout the iterations, new instances (agent-based simulation model parameter combinations and observed output values for those combinations) are added to the training data and feature elimination operations are applied to obtain an improved meta-model. Therefore, accuracy comparisons are made on the final meta-models of those procedure runs to evaluate the final meta-models that are generated at the end of the procedure.

6.5.1. Elimination Decisions Comparison

Before comparing the accuracy performances of the two procedure versions, eliminated features are investigated to understand if those decisions are coherent. Elimination decisions of 10 distinct repetitions that are performed with 10 different training data sets are reported for each version of the procedure. That means the elimination decisions are investigated for 100 distinct runs for each version of the procedure. Table 6.4 shows the elimination decisions that are taken in 100 distinct runs that applied the improved procedure. Each row shows the used training data sets. Each column shows the feature names that are eliminated at least once within 100 distinct runs. The number of elimination decisions that are taken within 10 repetitions of the respective training data set is reported within the cells. To be clear, the first cell shows that the *Evidence* feature of the meta-model (*Evidence* parameter of the agent-based simulation model) is eliminated in all of the 10 repetitions of the improved procedure that are trained with the first training data set.

Table 6.4. Elimination decisions of the improved procedure.

	Feature Names			
Training Data Set	*evidence*	max_links	prop-likelihood	sc-bel-prop
1	10	10	10	10
2	10	10	10	10
3	10	10	10	10
4	10	10	10	10
5	10	10	10	10
6	10	10	10	10
7	10	10	10	10
8	10	10	10	10
9	10	10	10	10
10	10	10	10	10
Total Elimination	100	100	100	100

Elimination decisions that are taken in 100 distinct runs that applied the previous procedure are investigated in Table 6.5. Table 6.5 is also constructed with the same logic as 6.4.

Table 6.5. Elimination decisions of the previously proposed procedure.

	Feature Names			
Training Data Set	*evidence*	max_links	prop-likelihood	sc-bel-prop
1	10	10	10	10
2	10	10	10	10
3	10	10	10	10
4	10	10	10	10
5	10	10	10	10
6	10	10	10	10
7	10	10	10	10
8	10	10	10	10
9	10	10	10	10
10	10	8	10	10
Total Elimination	100	98	100	100

Tables 6.4 and 6.5 firstly show that both procedures are consistent in themselves. Features that are selected for elimination once are selected for all repetitions that are trained with 10 different training data sets (except the *Max-Links* feature in the previous version of the procedure). Moreover, the eliminated features are also the same in both versions of the procedure. As a result, both versions are capable and consistent in terms of eliminating the features that have less or no effect on the output of interest.

6.5.2. Accuracy Performance Comparison

RMSE values are calculated for the final meta-models that are generated at the end of the procedure after the feature elimination decisions and training data set expansions. The boxplot in Figure 6.10 shows the average RMSE values for different versions of the procedure. As explained before, 10 distinct repetitions of the procedure

are performed with each training data set. The dots in the boxplot represents those repetitions and average RMSE values are calculated for the final meta-models that are obtained using different training data sets. So, each dot represents an average RMSE value that is calculated for 10 distinct meta-models that are trained with 10 different training data sets.

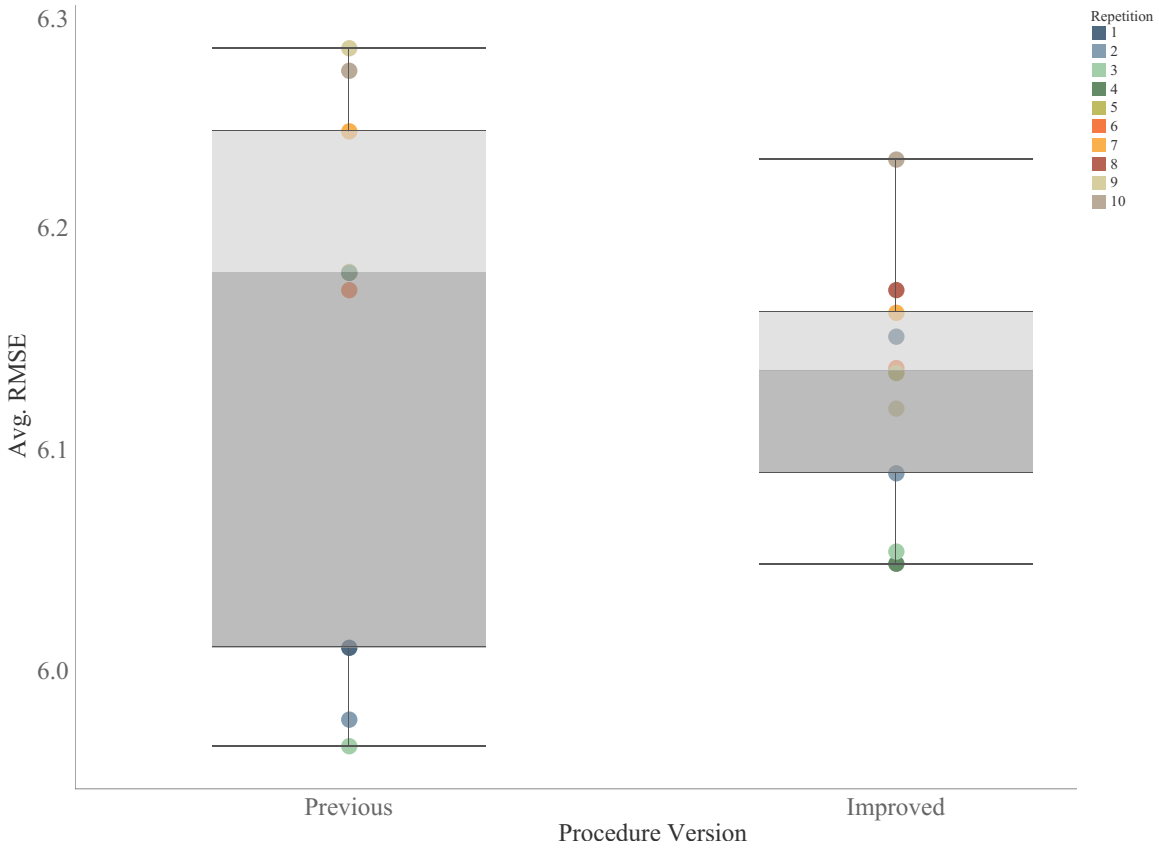

Figure 6.10. Comparison of average RMSE values for different versions of the procedure.

Figure 6.10 shows that the median RMSE for the previous procedure is higher than the value of the upper quartile for the improved procedure. Thus, it can be seen that there is a difference between these two versions in terms of accuracy. The whiskers and box sizes show that the previous version of the procedure has a wider distribution of average RMSE values for repetitions. The average RMSE values are more scattered for the previous version than for the improved version. Despite having a slight difference between median values, the difference between the variability of the two versions seems significant. So, it can be said that randomness has a greater effect on the previous version than on the improved version, and the improved version is more

robust in that manner.

The accuracy performances of distinct procedure repetitions trained with different data sets are also assessed. The aim is to compare two procedure versions considering any possible effects of training sets by comparing only runs that are trained with the same data sets.

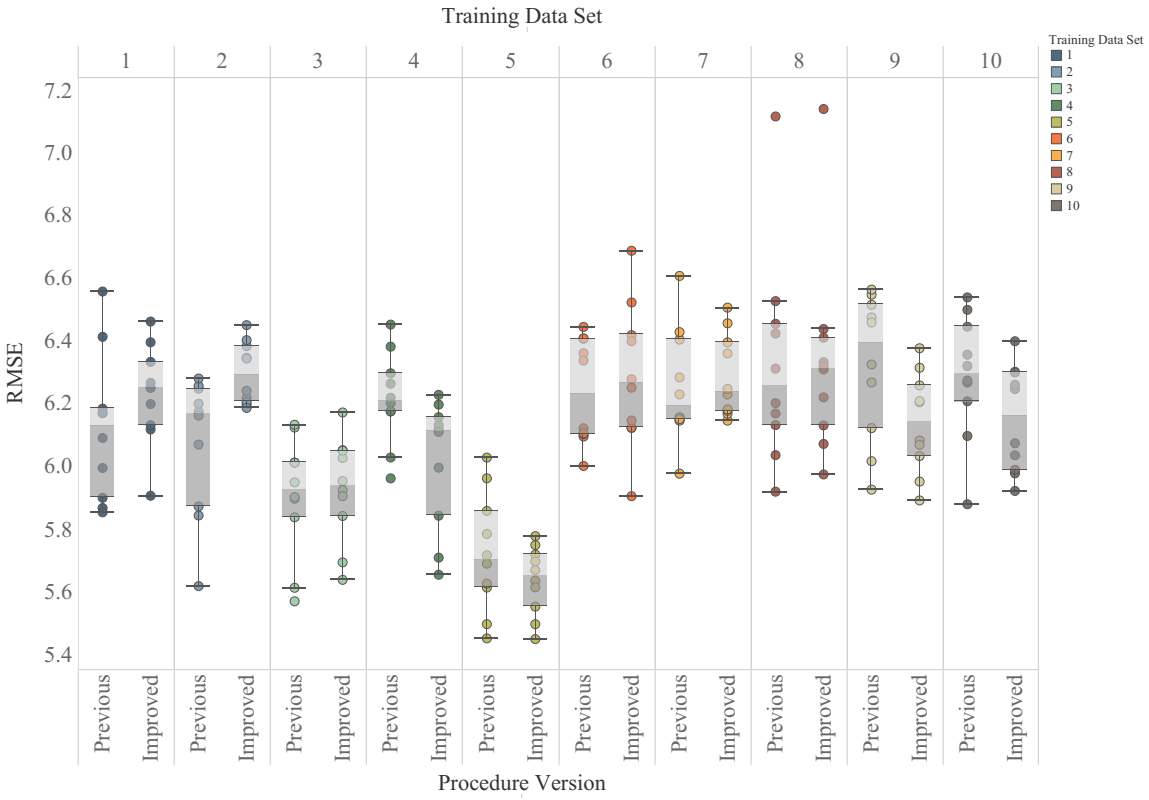

Figure 6.11. Boxplots to compare individual procedure runs over different repetitions and training data sets.

In Figure 6.11, each dot represents the RMSE value of the final meta-model obtained from one procedure run. Colors represent the training sets that are used in those runs. In each pane, two boxplots are drawn. Each boxplot contains 10 distinct procedure runs that used the respective training data set. Comparisons are made between those two boxplots containing 20 procedure runs that are trained with the same data set.

Eight out of ten comparisons support the previous interpretations by either showing lower median values for the improved procedure version or by showing wider dis-

tributions for the previous procedure version. Moreover, some of the comparisons (training data sets 5, 8, 9, 10) support both of those interpretations.

6.5.3. Run Time Comparison

In addition to accuracy and consistency, two procedures are compared in terms of average completion time. Figure 6.12 shows the comparison of the average completion time for two versions of the procedure. Each bar in the figure represents the average completion time of 10 repetitions of the procedure that used the respective training data sets. Each of 10 repetitions consists of meta-model training, feature elimination, and sampling processes throughout 11 iterations. Figure 6.12 shows that there is no evident supremacy of one version over another for the average duration comparison.

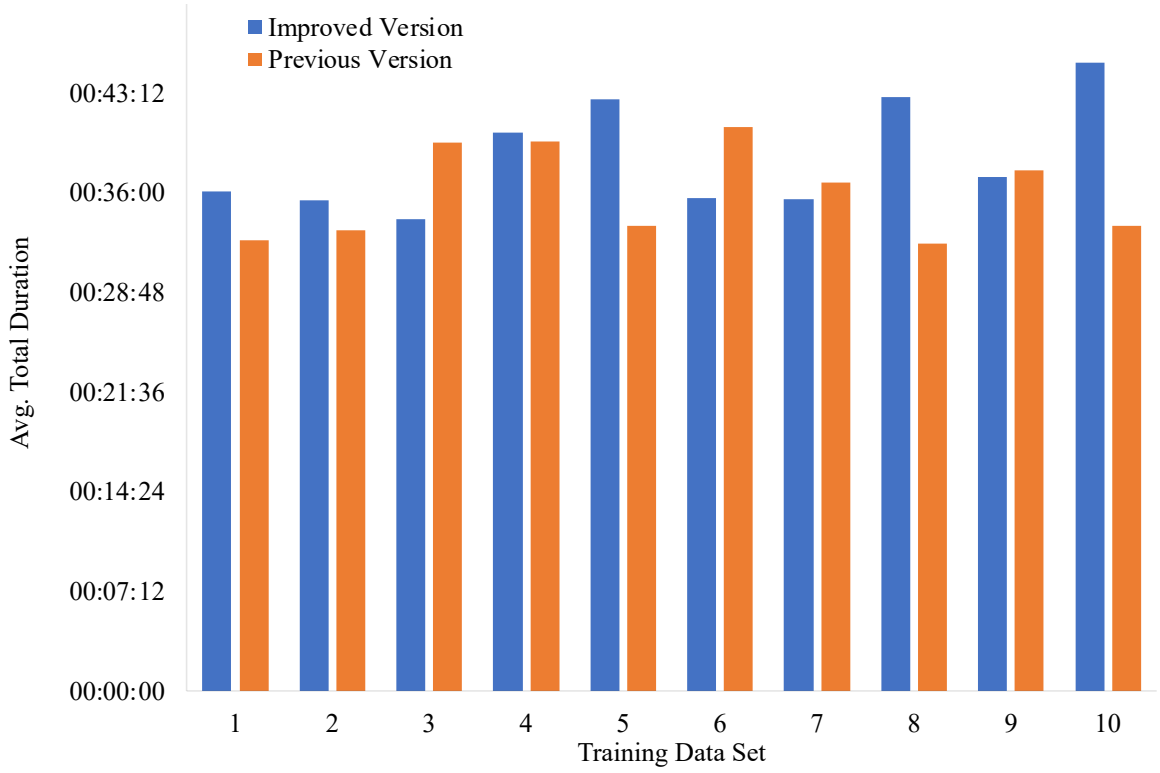

Figure 6.12. Average completion time comparison of procedure versions trained with different training data sets.

To investigate each repetition in detail and see if there are obvious outliers that affect the average value, Figure 6.13 is drawn. That figure also supports that there is no clear evidence to show that the improved version of the procedure has lower

completion time than the previous version has.

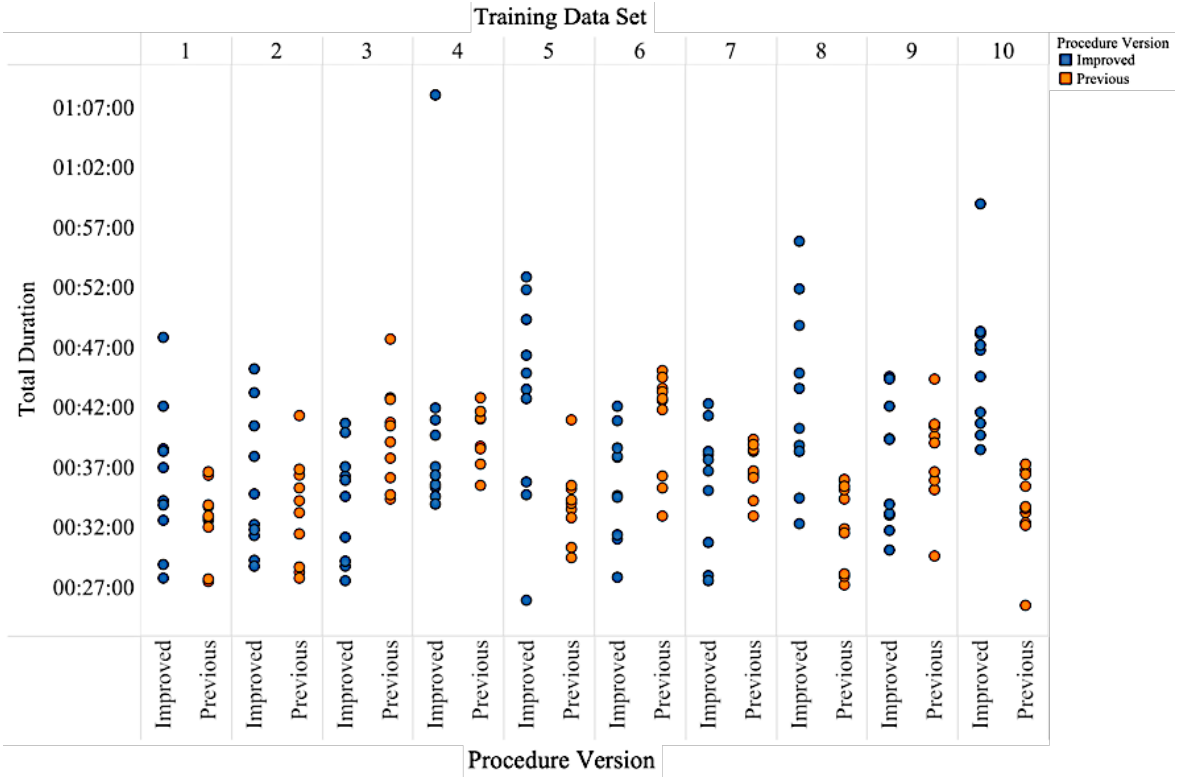

Figure 6.13. Completion times for individual repetitions of improved and previously proposed procedures.

6.5.4. Summary of Comparative Analysis

In the light of the analysis made in this chapter, three main results are drawn. The first result is that; both versions of the procedure are able to eliminate the same features from the meta-model. So, the procedure versions are consistent with each other. Those eliminated features are the ones that have slight or no impact on the output of interest, therefore the elimination decisions are confirmed to be valid. The experiments about the impact of features (parameters of agent-based simulation model) on the output can be found in chapter 6.4.

Moreover, same features are eliminated from the meta-model in all repetitions of procedures (both versions are replicated 10 times). This shows that both procedures are also consistent within themselves, and they are able to eliminate the insignificant features without being affected by randomness.

The second finding is on the accuracy performance comparison of two procedure versions. Comparing RMSE values of the final meta-models reached at the end of the procedures, two versions of the procedure are found to have differences. The main difference is found between the variabilities of accuracy performances. This interpretation shows that the randomness has a greater effect on the previous version than on the improved version.

The last finding is about the completion time comparison. The experiments show that there is no significant improvement in terms of completion time. To conclude, through experiments it is found that the main strength of the improved procedure over the previously proposed one is its accuracy performance and robustness.

7. DISCUSSIONS AND CONCLUSIONS

In this thesis, an advanced procedure that utilizes meta-modeling, adaptive sampling, and feature elimination methods is constructed to investigate agent-based simulation models. It is aimed to replicate the agent-based model with a meta-model by representing the agent-based model parameters with meta-model features and to predict the agent-based simulation model output via the meta-model. That approach also enables researchers to make interpretations of the relationships between agent-based simulation model parameters and outputs through feature importance measurements. In this thesis, previous studies in the literature on the concept of using meta-modeling techniques with agent-based simulation models are reviewed. Those methods found in the literature are applied to simple agent-based simulation models for performance evaluation. After reviewing and analyzing the previous applications, improvement opportunities in the previous works are discussed and practiced in this thesis.

A literature review is conducted to find an agent-based simulation model to apply the previously proposed procedure and evaluate performance. In order to find out improvement opportunities, a complex agent-based model with nontrivial parameters is sought after. Another motivation behind this search is that a complex agent-based model that contains non-linear relationships between its paraments and output gives more insights into the performance of the proposed procedure. For this purpose, agent-based simulation models that are analyzing socio-dynamic systems are reviewed. A recent agent-based simulation model that is constructed to analyse the echo chamber formation in social networks is selected to apply the procedure. The agent-based model is found as a good candidate in terms of complexity as it investigates the probabilistic nature of social media users' belief updating mechanism. Moreover, the agent-based model is constructed with a wide range of parameters to reflect real-life well. Thus, the agent-based model is selected to apply the procedure, conduct experiments, and define the improvement opportunities within the previously proposed procedure.

The previously proposed procedure consists of two sub-procedures. The first is the adaptive sampling procedure that constructs the final training data set ensuring the informative quality of the training instances. The second sub-procedure is feature elimination. That sub-procedure enables meta-model to focus on the significant features and prevent the curse of dimensionality.

The main improvement opportunity is found in the feature elimination part of the procedure as the elimination decisions depend on two procedure parameters that must be defined by the user. User-defined parameters of the feature elimination procedure are ones that need knowledge about the machine learning concepts to be defined correctly. They are not simple parameters to choose from distinct and/or finite alternative values. One of these parameters is *Elimination Proportion* which determines the number of features to be evaluated for elimination in each iteration and decreases that number (of candidate features) exponentially throughout the procedure. The elimination speed of the procedure depends directly on *Elimination Proportion*. An aggressive elimination approach is expected with large values of the *Elimination Proportion*. Another parameter is the *OOB Allowance* value which gives the final decision of feature elimination. At each iteration OOB value of the meta-model is recalculated assuming the features having the least importance values are eliminated (the number of elimination candidates is determined by the *Elimination Proportion*). Elimination is applied if the OOB error increase after the elimination assumption does not exceed the allowed value which is calculated by using the *OOB Allowance* value. So, it is expected from the user to complete a pre-work to learn about the concept and conduct experiments to choose the proper parameter values which fit the agent-based simulation model and purpose of the study. Thus, it is planned to remove the two parameters from the procedure, simplify the feature elimination process and introduce one simple parameter to control feature elimination decisions. In that manner firstly the flow of the procedure is changed by removing the part defining the number of features for elimination. Instead of defining a subset of the features to review for elimination, all features are evaluated with their own importance values. A new and straightforward parameter is introduced to the procedure which is the *Feature Importance Threshold* value to make the decision of elimination. Features having lower importance values than the threshold are

directly eliminated from the meta-model with the new design. Besides removing the *OOB Allowance* parameter from the procedure, that design also prevents the repetitive calculation of the OOB error rate in case of no elimination. For further improvement opportunities, different procedure designs are also assessed. The efficiency and accuracy of the procedure are affected by the overall iteration budget (total length of the procedure) and the iteration number that the feature elimination process starts, which are controlled by user-defined procedure parameters. The experiments are started by keeping the *Iteration Budget* constant and comparing 7 different *Elimination Start Iteration* values in terms of accuracy and procedure completion time. Starting feature elimination at earlier iterations supports efficiency, but the elimination decisions are made with a less saturated model than they can be in further iterations. On the other side, starting feature elimination later causes an opportunity loss of training the meta-model in a lower-dimensional space without insignificant features. So, the selection of a proper *Elimination Start Iteration* is crucial for procedure performance.

After gaining the initial insights from the first set of experiments, a more detailed investigation is designed to compare three levels of *Elimination Start Iteration* values. Three alternative starting points represent starting the elimination at the beginning of the procedure (second iteration), in the middle (fifth iteration), and at the last iterations (eighth iteration) of the procedure keeping the *Iteration Budget* constant. Experiments show consistent results with the initial interpretations that starting the elimination at the earlier and later iterations has different advantages when the *Iteration Budget* is constant.

Moreover, alternative designs are evaluated to see if dynamically determining the *Iteration Budget* will make any evident changes. First, an analysis is made on a scenario that the meta-model with the minimum RMSE is recorded as the final meta-model of the procedure. In that scenario the selected meta-model should be trained until reaching the *Iteration Budget*, thus that change does not simplify the procedure design. The user still needs to determine the proper *Iteration Budget* and all of the iterations need to be completed to select the meta-model with the minimum RMSE value. For this reason, another alternative design is proposed to terminate the

procedure at the first iteration where the RMSE value of the meta-model increases and keep the previous meta-model as the final one.

Three designs (using the final meta-model, using the meta-model having the minimum RMSE within the *Iteration Budget*, and using the meta-model of the previous iteration to the first RMSE increase) are compared in detail. The design of using the final meta-model at the end of the *Iteration Budget* performs better than the design of using the meta-model of the previous iteration of the first RMSE increase in terms of accuracy when starting the feature elimination at the beginning of the procedure (at the second and fifth iterations) as the second design prevents the meta-model to be improved further. Besides that, in the case of starting the elimination at the eighth iteration, the second design performs slightly better. The missed opportunity for meta-model improvement is also evaluated by comparing the second design with the one using the meta-model having the minimum RMSE within the *Iteration Budget*. It is seen that, especially for the case of starting the elimination earlier, there is a missed opportunity for accuracy improvement when using the meta-model of the previous iteration to the first RMSE increase. As a result of these evaluations, it is decided to continue with the first design, to use the final meta-model that is reached at the end of a predefined *Iteration Budget* and start the feature elimination process in the middle of the procedure. In this way, it is aimed to make the best use of the advantages of the feature elimination process and not miss the opportunity to develop the meta-model.

After designing the improved procedure, experiments are conducted to select the proper value of the feature importance value. The selected value is important for the performance of the improved procedure as it determines the procedure's tendency to make elimination decisions. 4 levels of alternative feature importance values are compared on 10 distinct procedure repetitions. Those repetitions are trained with a data set that contains 500 parameter value combinations for the agent-based simulation model. The *Feature Importance Threshold* value which gives the most accurate and robust meta-models for the agent-based simulation model that is used in this study is selected (the selected value is 0.1).

Validation of elimination decisions that are made by the improved procedure working with the selected *Feature Importance Threshold* is made through behavior space experiments of NetLogo. Experiments are repeated for 3 levels of population sizes, which are 1000, 5000, and 10000, to eliminate the effects of the population size. To take the randomness into consideration, simulation runs are repeated 50 times for each agent-based parameter combination. Agent-based parameters corresponding to the meta-model features which are eliminated by the procedure are assessed in terms of their effects on the output. Moreover, the effect of an agent-based model parameter which is corresponding to a meta-model feature that is considered significant (and not eliminated) is investigated. At the end of the detailed analysis, elimination decisions of the procedure are proven to be valid.

After the validation phase of the procedure, comparative analysis experiments are conducted to evaluate the performance of the improved procedure over the previously proposed one. 10 distinct repetitions of each of the two procedure versions are performed with 10 different training data sets throughout the experiments. A complete procedure is designed with 11 iterations, and each procedure is performed 100 times (10 repetitions for 10 training data sets). So, for each version of the procedures, the meta-model is trained 1100 times and the comparisons are made using the final meta-models that are built at the end of the procedure after adaptive sampling and feature elimination processes. As the first checkpoint, the eliminated features are compared within and between each procedure version. Both versions of the procedure are found to be consistent within themselves, they are able to eliminate the same features at all repetitions. Moreover, the same features are eliminated in each procedure version, so both versions are found to be consistent to detect the same features as insignificant. After checking the consistency, the accuracy performances of the procedures are compared. Analysis showed that the improved procedure is able to reach meta-models with lower RMSE values at the final iteration compared to the previously proposed version. Moreover, the comparison of average RMSE values of the final meta-models trained with different training data sets shows that improved procedure is affected less by the randomness. Thus, the improved version is found more robust than the previously proposed procedure. After the consistency investigation and accuracy comparison, the

runtimes of the procedure versions are evaluated. It is observed that there is no evident result indicating that one version of the procedure has a shorter run time than the other.

REFERENCES

1. Polhill, J. G., J. Ge, M. P. Hare, K. B. Matthews, A. Gimona, D. Salt and J. Yeluripati, "Crossing the Chasm: A 'Tube-Map' for Agent-Based Social Simulation of Policy Scenarios in Spatially-Distributed Systems", *GeoInformatica*, Vol. 23, pp. 169-199, 2019.

2. Walbert, H. J., J. L. Caton and J. R. Norgaard, "Countries as Agents in a Global Scale Computational Model", *Journal of Artificial Societies and Social Simulation*, Vol. 21, 2018.

3. Lempert, R., "Agent-Based Modeling as Organizational and Public Policy Simulators", *Proceedings of the National Academy of Sciences*, Vol. 99, pp. 7195–7196, 2002.

4. Yurdadön, P., *Adaptive Sampling with Feature Elimination for Agent-Based Models*, M.S. Thesis, Boğaziçi University, 2020.

5. Lamperti, F., A. Roventini and A. Sani, "Agent-Based Model Calibration Using Machine Learning Surrogates", *Journal of Economic Dynamics and Control*, Vol. 90, pp. 366-389, 2018.

6. Edali, M., *Analysis of Agent-Based Simulation Models Through Metamodeling*, Ph.D. Thesis, Boğaziçi University, 2019.

7. Edali, M. and G. Yucel, "Exploring the Behavior Space of Agent-Based Simulation Models Using Random Forest Metamodels and Sequential Sampling", *Simulation Modelling Practice and Theory*, Vol. 92, pp. 62-81, 2019.

8. Cai, J., J. Luo, S. Wang and S. Yang, "Feature Selection in Machine Learning: A New Perspective", *Neurocomputing*, Vol. 300, pp. 70-79, 2018.

9. Lee, J.-S., T. Filatova, A. Ligmann-Zielinska, B. Hassani-Mahmooei, F. Stonedahl, I. Lorscheid, A. Voinov, G. Polhill, Z. Sun and D. C. Parker, "The Complexities of Agent-Based Modeling Output Analysis", *Journal of Artificial Societies and Social Simulation*, Vol. 18, 2015.

10. Zhao, Z., F. Morstatter, S. Sharma, S. Alelyani, A. Anand and H. Liu, "Advancing Feature Selection Research", *ASU Feature Selection Repository Arizona State University*, pp. 1-28, 2010.

11. Vicario, M. D., A. Bessi, F. Zollo, F. Petroni, A. Scala, G. Caldarelli, H. E. Stanley and W. Quattrociocchi, "The Spreading of Misinformation Online", *Proceedings of the National Academy of Sciences*, Vol. 113, pp. 554-559, 2016.

12. Vicario, M. D., W. Quattrociocchi, A. Scala and F. Zollo, "Polarization and Fake News: Early Warning of Potential Misinformation Targets", *ACM Transactions on the Web*, Vol. 13, pp. 1-22, 2019.

13. Fränken, J. P. and T. Pilditch, "Cascades Across Networks are Sufficient for the Formation of Echo Chambers: An Agent-Based Model", *Journal of Artificial Societies and Social Simulation*, Vol. 24, 2021.

14. Galán, J. M., L. R. Izquierdo, S. S. Izquierdo, J. I. Santos, R. del Olmo, A. López Paredes and B. Edmond, "Errors and Artefacts in Agent-Based Modelling", *Errors and Artefacts in Agent-Based Modelling*, Vol. 12, pp. 1-1, 2009.

15. Russell, S. and P. Norvig, *Artificial Intelligence : A Modern Approach*, Pearson Education Inc, New Jersey, 2021.

16. Janiesch, C., P. Zschech and K. Heinrich, "Machine Learning and Deep Learning", *Electronic Markets*, Vol. 31, pp. 685-695, 2021.

17. Bishop, C. M., *Pattern Recognition and Machine Learning (Information Science and Statistics)*, Springer Science+Business Media LLC, New York, 2006.

18. Aria, M., C. Cuccurullo and A. Gnasso, "A Comparison Among Interpretative Proposals for Random Forests", *Machine Learning with Applications*, Vol. 6, p. 100094, 2021.

19. Breiman, L., "Random Forests", *Machine Learning*, Vol. 45, pp. 5-32, 2001.

20. Burrage, K., P. Burrage, D. Donovan and B. Thompson, "Populations of Models, Experimental Designs and Coverage of Parameter Space by Latin Hypercube and Orthogonal Sampling", *Procedia Computer Science*, Vol. 51, pp. 1762-1771, 2015.

21. McKay, M. D., R. J. Beckman and W. J. Conover, "Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code", *Technometrics*, Vol. 21 pp. 239-245, 1979.

22. Schelling, T. C., *Micromotives and Macrobehavior*, Norton, New York, 1978.

23. Merdes, C., M. von Sydow and U. Hahn, "Formal Models of Source Reliability", *Synthese*, Vol. 198, pp. 5573-5801, 2021.

APPENDIX A: OBSERVED RMSE VALUES

		Iterations												
Elimination Start Iteration	Repetitions	1	2	3	4	5	6	7	8	9	10	11	Min rmse	Min rmse iter
2	1	7.04	7.05	6.61	6.56	6.74	6.42	6.42	6.38	6.56	6.31	6.25	6.25	11
2	2	7.04	7.30	6.72	6.57	6.50	6.59	6.58	6.47	6.57	6.63	6.09	6.09	11
2	3	7.04	7.04	6.35	5.94	5.99	6.09	6.00	5.88	6.18	5.78	5.91	5.78	10
2	4	7.04	6.75	6.25	6.49	6.40	6.33	6.50	6.47	6.01	6.26	6.14	6.01	9
2	5	7.04	7.21	6.51	6.78	6.86	7.10	6.87	6.56	6.34	6.28	6.42	6.28	10
2	6	7.04	7.00	6.66	6.63	6.69	6.49	6.24	6.38	6.50	6.38	6.27	6.24	7
2	7	7.04	6.96	6.63	6.39	6.50	6.40	6.32	6.24	6.16	6.07	6.15	6.07	10
2	8	7.04	7.30	6.51	6.71	6.44	6.17	6.21	6.30	6.22	6.26	5.93	5.93	11
2	9	7.04	6.99	6.49	6.26	6.32	6.24	6.31	6.25	6.28	6.16	6.21	6.16	10
2	10	7.04	7.21	6.54	6.66	6.58	6.71	6.59	6.47	6.40	6.53	6.41	6.40	9
2	11	7.04	7.00	6.65	6.48	6.62	6.40	6.40	6.68	6.50	6.37	6.49	6.37	10
2	12	7.04	7.04	6.44	6.22	6.48	6.45	6.53	6.35	6.23	6.37	6.30	6.22	4
2	13	7.04	6.86	6.50	6.54	6.45	6.34	6.58	6.42	6.37	6.22	6.18	6.18	11
2	14	7.04	7.04	6.50	6.29	6.29	6.10	6.16	6.11	6.04	5.96	5.82	5.82	11
2	15	7.04	6.79	6.29	6.18	6.38	6.05	6.23	6.01	5.97	5.92	5.94	5.92	10
2	16	7.04	7.24	6.81	7.03	6.87	6.86	6.75	6.40	6.61	6.48	6.70	6.40	8
2	17	7.04	7.36	6.74	7.02	6.72	6.60	6.69	6.62	6.70	6.64	6.86	6.60	6
2	18	7.04	7.07	6.61	6.80	6.52	6.41	6.51	6.41	6.56	6.73	6.66	6.41	8
2	19	7.04	7.11	6.86	6.61	6.62	6.54	6.48	6.53	6.67	6.54	6.37	6.37	11
2	20	7.04	7.13	6.53	6.56	6.39	6.46	6.66	6.64	6.40	6.54	6.52	6.39	5
2	21	7.04	7.19	6.28	6.18	6.38	6.41	6.40	6.18	6.44	6.29	6.45	6.18	4
2	22	7.04	6.63	6.29	6.28	6.37	6.19	6.38	6.30	6.30	6.16	6.13	6.13	11
2	23	7.04	7.03	6.47	6.53	6.73	6.70	6.64	6.76	6.47	6.25	6.34	6.25	10
2	24	7.04	7.04	6.53	6.46	6.29	6.52	6.43	6.42	6.31	6.18	6.26	6.18	10
2	25	7.04	6.75	6.67	6.61	6.44	6.40	6.54	6.34	6.41	6.52	6.33	6.33	11
2	26	7.04	7.10	6.66	6.69	6.62	6.61	6.59	6.59	6.56	6.69	6.62	6.56	9
2	27	7.04	7.06	6.54	6.42	6.54	6.41	6.29	6.50	6.42	6.53	6.27	6.27	11
2	28	7.04	7.17	6.66	6.57	6.37	6.38	6.51	6.72	6.63	6.43	6.59	6.37	5
2	29	7.04	7.09	6.81	6.73	6.53	6.38	6.53	6.60	6.45	6.35	6.00	6.00	11
2	30	7.04	7.00	6.60	6.71	6.58	6.57	6.77	6.46	6.47	5.92	5.84	5.84	11
5	1	7.04	7.17	7.13	7.04	6.76	6.52	6.21	6.27	6.10	6.34	6.22	6.10	9
5	2	7.04	7.15	7.16	6.83	6.95	6.40	6.30	6.39	6.39	6.44	6.26	6.26	11
5	3	7.04	6.86	6.50	6.71	6.81	6.29	6.28	6.19	6.36	6.09	6.14	6.09	10
5	4	7.04	6.68	6.67	6.87	6.83	6.23	6.42	6.44	6.24	6.32	6.15	6.15	11
5	5	7.04	7.20	7.15	7.15	7.40	6.71	6.71	6.33	6.38	6.24	6.22	6.22	11
5	6	7.04	6.98	7.03	7.33	7.33	6.70	6.31	6.35	6.28	6.31	6.10	6.10	11
5	7	7.04	6.99	6.92	7.11	7.10	6.51	6.48	6.48	6.37	6.35	6.28	6.28	11
5	8	7.04	7.01	7.14	7.08	7.04	6.40	6.34	6.40	6.45	6.38	6.35	6.34	7
5	9	7.04	6.88	6.95	7.13	6.87	6.41	6.41	6.27	6.33	6.45	6.28	6.27	8
5	10	7.04	7.02	7.17	7.11	6.85	6.47	6.49	6.46	6.48	6.48	6.53	6.46	8
5	11	7.04	7.06	7.11	6.93	6.94	6.14	6.34	6.11	6.23	6.03	6.26	6.03	10
5	12	7.04	6.88	7.03	6.92	6.67	6.31	6.14	6.12	6.14	6.21	5.92	5.92	11
5	13	7.04	6.96	6.93	6.60	6.73	6.40	6.42	6.46	6.46	6.25	6.31	6.25	10
5	14	7.04	7.00	6.81	6.86	6.96	6.54	6.60	6.51	6.35	6.14	6.21	6.14	10
5	15	7.04	6.74	6.74	6.84	6.87	6.40	6.53	6.30	6.09	6.25	6.21	6.09	9
5	16	7.04	7.27	7.28	7.23	7.01	6.84	6.78	6.55	6.48	6.54	6.74	6.48	9
5	17	7.04	7.21	7.02	7.30	7.23	6.89	6.63	6.49	6.64	6.67	6.61	6.49	8
5	18	7.04	7.18	7.14	7.02	7.18	6.48	6.38	6.50	6.35	6.46	6.47	6.35	9
5	19	7.04	7.13	7.03	7.06	6.79	6.34	6.32	6.31	6.36	6.18	6.25	6.18	10
5	20	7.04	7.09	7.05	6.65	6.68	6.32	6.38	6.40	6.21	6.25	6.41	6.21	9
5	21	7.04	7.31	6.71	6.60	6.65	6.27	6.23	6.26	6.04	6.09	5.96	5.96	11
5	22	7.04	6.81	6.77	6.56	6.49	6.20	6.27	6.21	6.09	6.30	6.16	6.09	9
5	23	7.04	6.91	6.91	6.62	6.85	6.36	6.49	6.47	6.44	6.46	6.29	6.29	11
5	24	7.04	6.98	6.83	7.01	6.84	6.64	6.67	6.72	6.55	6.59	6.38	6.38	11
5	25	7.04	6.89	6.93	6.88	6.82	6.71	6.62	6.51	6.50	6.38	6.50	6.38	10
5	26	7.04	7.22	7.14	7.11	7.15	6.92	6.54	6.89	6.82	6.90	6.86	6.54	7
5	27	7.04	6.81	7.05	7.14	6.85	6.64	6.77	6.66	6.73	6.72	6.70	6.64	6
5	28	7.04	7.04	7.01	7.06	7.11	6.51	6.60	6.63	6.95	6.48	6.42	6.42	11
5	29	7.04	7.01	7.08	7.06	6.96	6.66	6.61	6.69	6.48	6.32	6.03	6.03	11
5	30	7.04	7.11	7.03	7.22	6.99	6.68	6.80	6.63	6.56	6.03	5.99	5.99	11
8	1	7.04	6.90	7.12	7.19	6.81	6.76	6.92	6.54	6.17	6.23	6.27	6.17	9
8	2	7.04	7.11	7.07	6.86	6.76	7.04	6.97	7.08	6.18	6.24	6.19	6.18	9
8	3	7.04	6.89	6.52	6.76	6.60	6.88	6.79	6.78	6.25	6.13	6.10	6.10	11
8	4	7.04	6.69	6.93	6.96	6.75	6.81	6.96	6.83	6.21	6.30	6.21	6.21	11
8	5	7.04	7.18	7.35	7.14	7.33	7.32	7.26	7.23	6.38	6.43	6.46	6.38	9
8	6	7.04	7.06	7.13	7.25	7.25	7.08	7.16	6.88	6.39	6.37	6.45	6.37	10
8	7	7.04	7.22	7.09	7.02	7.31	7.23	6.82	6.89	6.36	6.41	6.46	6.36	9
8	8	7.04	7.25	6.89	7.14	7.20	7.17	6.83	6.79	6.06	5.87	5.88	5.87	10
8	9	7.04	7.01	7.00	6.95	6.90	6.86	6.77	6.52	6.06	5.92	5.96	5.92	10
8	10	7.04	7.18	7.17	6.94	7.01	6.95	6.75	6.84	6.34	6.34	6.27	6.27	11
8	11	7.04	6.93	6.88	6.72	6.91	6.64	6.70	6.82	6.26	6.29	6.38	6.26	9
8	12	7.04	6.91	7.03	6.90	6.74	6.81	6.68	6.76	6.33	6.34	6.04	6.04	11
8	13	7.04	6.88	6.95	6.74	6.91	6.90	6.89	6.74	6.40	6.24	6.16	6.16	11
8	14	7.04	7.09	6.77	6.97	6.90	6.84	6.89	6.95	6.43	6.42	6.25	6.25	11
8	15	7.04	6.71	7.01	6.86	6.97	7.07	6.96	6.91	6.34	6.46	6.37	6.34	9
8	16	7.04	7.08	7.19	7.11	7.27	7.24	7.24	7.04	6.44	6.40	6.57	6.40	10
8	17	7.04	7.08	7.37	7.32	7.28	7.36	6.86	6.75	6.59	6.39	6.46	6.39	10
8	18	7.04	6.90	7.15	7.03	7.07	6.83	6.46	6.52	6.14	6.12	6.00	6.00	11
8	19	7.04	7.16	7.24	6.99	6.65	6.46	6.53	6.48	6.37	6.23	6.06	6.06	11
8	20	7.04	6.96	6.99	6.83	6.72	6.67	6.58	6.60	6.19	6.35	6.08	6.08	11
8	21	7.04	6.93	6.70	6.66	6.63	6.47	6.54	6.43	6.26	6.20	6.04	6.04	11
8	22	7.04	6.74	6.53	6.46	6.34	6.46	6.39	6.45	6.34	6.38	6.22	6.22	11
8	23	7.04	6.93	6.89	6.89	6.75	6.77	6.89	6.95	6.66	6.47	6.47	6.47	10
8	24	7.04	6.90	6.87	6.89	6.85	6.74	7.00	6.98	6.82	6.85	6.87	6.74	6
8	25	7.04	6.93	6.88	6.92	6.86	6.99	6.98	7.03	6.59	6.66	6.73	6.59	9
8	26	7.04	7.08	6.99	6.95	7.22	7.16	7.09	6.80	6.70	6.74	6.74	6.70	9
8	27	7.04	7.07	7.09	7.06	6.96	7.17	6.95	6.84	6.53	6.48	6.39	6.39	11
8	28	7.04	7.01	7.26	7.31	7.08	7.33	7.20	7.21	6.84	6.37	6.56	6.37	10
8	29	7.04	7.08	7.03	7.02	7.11	6.97	7.14	6.89	6.60	6.46	6.18	6.18	11
8	30	7.04	6.99	7.14	7.12	6.80	6.94	7.09	6.78	6.47	6.15	6.13	6.13	11

Figure A.1. Raw data of minimum RMSE values and iterations where they were observed.

APPENDIX B: DESIGN COMPARISONS THROUGH RMSE VALUES

		Iterations									
Elimination Start Iteration	Repetition	2	3	4	5	6	7	8	9	10	11
2	1	0.24%	-6.22%	-0.84%	2.73%	-4.69%	0.04%	-0.77%	2.82%	-3.76%	-0.94%
2	2	3.80%	-8.02%	-2.21%	-1.12%	1.44%	-0.21%	-1.62%	1.52%	1.02%	-8.20%
2	3	-0.01%	-9.71%	-6.56%	0.88%	1.67%	-1.35%	-2.13%	5.20%	-6.59%	2.31%
2	4	-4.06%	-7.36%	3.84%	-1.45%	-1.02%	2.70%	-0.47%	-7.14%	4.07%	-1.91%
2	5	2.53%	-9.79%	4.22%	1.15%	3.44%	-3.13%	-4.55%	-3.41%	-0.83%	2.12%
2	6	-0.45%	-4.96%	-0.37%	0.86%	-2.93%	-3.94%	2.26%	1.90%	-1.88%	-1.67%
2	7	-1.07%	-4.73%	-3.72%	1.84%	-1.56%	-1.33%	-1.23%	-1.21%	-1.57%	1.40%
2	8	3.81%	-10.85%	2.99%	-3.91%	-4.19%	0.51%	1.50%	-1.22%	0.66%	-5.38%
2	9	-0.60%	-7.18%	-3.59%	1.02%	-1.38%	1.20%	-0.97%	0.48%	-1.89%	0.75%
2	10	2.48%	-9.36%	1.87%	-1.15%	2.02%	-1.83%	-1.90%	-1.01%	2.10%	-1.94%
2	11	-0.54%	-4.91%	-2.57%	2.12%	-3.31%	0.04%	4.24%	-2.60%	-1.98%	1.88%
2	12	0.09%	-8.54%	-3.44%	4.20%	-0.40%	1.18%	-2.70%	-1.88%	2.19%	-1.15%
2	13	-2.56%	-5.23%	0.69%	-1.34%	-1.79%	3.84%	-2.41%	-0.90%	-2.31%	-0.55%
2	14	0.00%	-7.68%	-3.10%	-0.06%	-3.04%	0.95%	-0.80%	-1.04%	-1.39%	-2.28%
2	15	-3.49%	-7.40%	-1.70%	3.26%	-5.17%	2.98%	-3.51%	-0.78%	-0.80%	0.43%
2	16	2.84%	-5.95%	3.27%	-2.18%	-0.22%	-1.54%	-5.20%	3.25%	-1.90%	3.29%
2	17	4.63%	-8.49%	4.15%	-4.18%	-1.89%	1.39%	-0.94%	1.13%	-0.88%	3.35%
2	18	0.48%	-6.45%	2.86%	-4.13%	-1.73%	1.64%	-1.68%	2.44%	2.65%	-1.09%
2	19	1.01%	-3.45%	-3.69%	0.19%	-1.17%	-1.03%	0.89%	2.14%	-2.08%	-2.60%
2	20	1.32%	-8.38%	0.37%	-2.46%	0.99%	3.06%	-0.20%	-3.72%	2.22%	-0.24%
2	21	2.17%	-12.64%	-1.64%	3.35%	0.46%	-0.27%	-3.37%	4.28%	-2.42%	2.50%
2	22	-5.70%	-5.24%	-0.14%	1.45%	-2.87%	3.05%	-1.25%	0.13%	-2.22%	-0.57%
2	23	-0.12%	-7.89%	0.92%	3.05%	-0.52%	-0.80%	1.82%	-4.38%	-3.38%	1.46%
2	24	0.08%	-7.27%	-1.03%	-2.65%	3.67%	-1.42%	-0.10%	-1.78%	-2.04%	1.32%
2	25	-4.01%	-1.20%	-0.89%	-2.69%	-0.55%	2.21%	-3.08%	1.09%	1.76%	-3.00%
2	26	0.93%	-6.17%	0.45%	-1.03%	-0.15%	-0.37%	0.00%	-0.51%	2.05%	-1.08%
2	27	0.28%	-7.35%	-1.82%	1.82%	-1.95%	-1.79%	3.26%	-1.22%	1.69%	-3.88%
2	28	1.92%	-7.11%	-1.35%	-3.08%	0.17%	2.02%	3.31%	-1.39%	-2.97%	2.42%
2	29	0.81%	-4.06%	-1.04%	-3.09%	-2.17%	2.20%	1.11%	-2.30%	-1.54%	-5.47%
2	30	-0.48%	-5.74%	1.61%	-1.86%	-0.18%	2.98%	-4.58%	0.18%	-8.40%	-1.50%
5	1	1.90%	-0.54%	-1.29%	-3.94%	-3.55%	-4.79%	1.00%	-2.67%	3.87%	-1.95%
5	2	1.58%	0.21%	-4.66%	1.76%	-7.85%	-1.63%	1.47%	-0.07%	0.77%	-2.76%
5	3	-2.52%	-5.19%	3.11%	1.50%	-7.57%	-0.18%	-1.36%	2.72%	-4.23%	0.85%
5	4	-5.00%	-0.26%	2.99%	-0.55%	-8.73%	2.94%	0.44%	-3.25%	1.31%	-2.69%
5	5	2.33%	-0.70%	0.07%	3.44%	-9.30%	0.03%	-5.66%	0.66%	-2.14%	-0.31%
5	6	-0.77%	0.66%	4.32%	0.04%	-8.72%	-5.74%	0.59%	-1.15%	0.58%	-3.36%
5	7	-0.70%	-0.99%	2.74%	-0.15%	-8.25%	-0.44%	-0.10%	-1.65%	-0.22%	-1.13%
5	8	-0.36%	1.90%	-0.88%	-0.61%	-9.12%	-0.87%	0.95%	0.78%	-1.09%	-0.55%
5	9	-2.28%	1.07%	2.66%	-3.74%	-6.66%	0.03%	-2.25%	1.03%	1.81%	-2.66%
5	10	-0.22%	2.11%	-0.87%	-3.55%	-5.55%	0.32%	-0.48%	0.29%	-0.01%	0.77%
5	11	0.39%	0.65%	-2.46%	0.03%	-11.51%	3.36%	-3.63%	1.82%	-3.19%	3.81%
5	12	-2.28%	2.25%	-1.64%	-3.59%	-5.41%	-2.67%	-0.36%	0.39%	1.10%	-4.59%
5	13	-1.03%	-0.50%	-4.75%	1.93%	-4.94%	0.33%	0.63%	0.12%	-3.33%	1.00%
5	14	-0.58%	-2.60%	0.74%	1.34%	-6.02%	1.02%	-1.40%	-2.48%	-3.24%	1.12%
5	15	-4.17%	-0.01%	1.40%	0.51%	-6.91%	2.09%	-3.56%	-3.26%	2.55%	-0.61%
5	16	3.26%	0.20%	-0.73%	-3.01%	-2.43%	-0.90%	-3.37%	-1.09%	1.00%	3.04%
5	17	2.43%	-2.65%	4.07%	-1.01%	-4.73%	-3.77%	-2.09%	2.33%	0.49%	-0.95%
5	18	2.02%	-0.49%	-1.73%	2.35%	-9.86%	-1.53%	1.90%	-2.33%	1.70%	0.16%
5	19	1.40%	-1.46%	0.41%	-3.76%	-6.67%	-0.39%	-0.08%	0.73%	-2.76%	1.13%
5	20	0.73%	-0.56%	-5.69%	0.43%	-5.29%	0.94%	0.31%	-2.99%	0.68%	2.50%
5	21	3.84%	-8.18%	-1.67%	0.82%	-5.76%	-0.66%	0.53%	-3.52%	0.83%	-2.09%
5	22	-3.25%	-0.53%	-3.15%	-1.10%	-4.42%	1.17%	-1.05%	-1.80%	3.37%	-2.23%
5	23	-1.73%	-0.01%	-4.27%	3.47%	-7.05%	1.93%	-0.34%	-0.41%	0.34%	-2.64%
5	24	-0.84%	-2.15%	2.65%	-2.36%	-2.93%	0.49%	0.62%	-2.48%	0.68%	-3.20%
5	25	-2.14%	0.58%	-0.64%	-0.95%	-1.53%	-1.34%	-1.69%	-0.21%	-1.80%	1.86%
5	26	2.67%	-1.12%	-0.41%	0.48%	-3.15%	-5.59%	5.37%	-0.92%	1.12%	-0.62%
5	27	-3.20%	3.58%	1.25%	-4.13%	-3.02%	2.00%	-1.66%	1.00%	-0.17%	-0.32%
5	28	0.02%	-0.44%	0.79%	0.64%	-8.42%	1.34%	0.57%	4.71%	-6.70%	-1.00%
5	29	-0.32%	0.97%	-0.31%	-1.43%	-4.29%	-0.66%	1.12%	-3.16%	-2.46%	-4.61%
5	30	1.04%	-1.11%	2.70%	-3.13%	-4.54%	1.83%	-2.42%	-1.06%	-8.20%	-0.65%
8	1	-1.90%	3.21%	0.92%	-5.25%	-0.74%	2.29%	-5.45%	-5.63%	0.91%	0.63%
8	2	1.01%	-0.54%	-3.03%	-1.42%	4.11%	-0.93%	1.51%	-12.70%	0.99%	-0.74%
8	3	-2.03%	-5.37%	3.69%	-2.36%	4.16%	-1.32%	-0.11%	-7.76%	-2.04%	-0.43%
8	4	-4.94%	3.67%	0.37%	-3.06%	0.99%	2.21%	-1.93%	-9.02%	1.41%	-1.42%
8	5	2.05%	2.40%	-2.94%	2.69%	-0.16%	-0.81%	-0.45%	-11.66%	0.75%	0.42%
8	6	0.37%	0.94%	1.72%	4.07%	-2.22%	1.07%	-3.37%	-7.18%	-0.35%	1.29%
8	7	2.61%	-1.77%	-0.96%	4.08%	-1.05%	-5.76%	1.09%	-7.76%	0.75%	0.86%
8	8	2.97%	-4.84%	3.60%	0.79%	-0.38%	-4.70%	-0.59%	-10.81%	-3.17%	0.21%
8	9	-0.36%	-0.21%	-0.62%	-0.76%	-0.62%	-1.25%	-3.68%	-7.14%	-2.27%	0.77%
8	10	2.05%	-0.13%	-3.28%	1.09%	-0.84%	-2.91%	1.30%	-7.32%	0.09%	-1.23%
8	11	-1.44%	-0.77%	-2.38%	2.94%	-3.98%	0.91%	1.73%	-8.13%	0.47%	1.41%
8	12	-1.85%	1.82%	-1.81%	-2.44%	1.13%	-2.00%	1.32%	-6.46%	0.17%	-4.64%
8	13	-2.18%	1.02%	-3.11%	2.65%	-0.22%	-0.12%	-2.22%	-5.08%	-2.39%	-1.26%
8	14	0.75%	-4.47%	2.96%	-1.02%	-0.88%	0.70%	0.93%	-7.46%	-0.22%	-2.62%
8	15	-4.57%	4.37%	-2.12%	1.62%	1.43%	-1.50%	-0.85%	-8.24%	2.01%	-1.49%
8	16	0.59%	1.66%	-1.20%	2.33%	-0.41%	-0.03%	-2.82%	-8.54%	-0.55%	2.71%
8	17	0.69%	4.09%	-0.73%	0.57%	1.12%	-6.84%	-1.51%	-2.41%	-3.11%	1.17%
8	18	-1.96%	3.71%	-1.68%	0.55%	-3.42%	-5.35%	0.90%	-5.82%	-0.31%	-2.09%
8	19	1.75%	1.09%	-3.46%	-4.79%	-2.86%	1.04%	-0.69%	-1.75%	-2.14%	-2.83%
8	20	-1.05%	0.46%	-2.34%	-1.56%	-0.78%	-1.40%	0.31%	-6.14%	2.58%	-4.28%
8	21	-1.52%	-3.24%	-0.63%	-0.52%	-2.38%	1.05%	-1.58%	-2.65%	-1.02%	-2.61%
8	22	-4.15%	-3.22%	-1.02%	-1.86%	1.86%	-0.99%	0.90%	-1.79%	0.64%	-2.48%
8	23	-1.52%	-0.52%	-0.12%	-1.91%	0.28%	1.67%	0.93%	-4.22%	-2.80%	0.07%
8	24	-1.96%	-0.46%	0.31%	-0.50%	-1.59%	3.85%	-0.41%	-2.27%	0.52%	0.22%
8	25	-1.48%	-0.82%	0.58%	-0.76%	1.80%	-0.12%	0.77%	-6.33%	1.05%	1.17%
8	26	0.68%	-1.32%	-0.59%	3.85%	-0.72%	-1.06%	-4.10%	-1.49%	0.63%	0.03%
8	27	0.42%	0.28%	-0.31%	-1.50%	3.05%	-3.00%	-1.58%	-4.62%	-0.75%	-1.44%
8	28	-0.41%	3.63%	0.62%	-3.14%	3.53%	-1.66%	0.10%	-5.19%	-6.86%	2.93%
8	29	0.66%	-0.78%	-0.04%	1.28%	-2.02%	2.40%	-3.44%	-4.23%	-2.15%	-4.39%
8	30	-0.67%	2.15%	-0.32%	-4.45%	2.08%	2.13%	-4.29%	-4.59%	-4.91%	-0.34%

Chosen Iteration	Chosen RMSE	Min RMSE Iter	Min RMSE	Last RMSE	Last vs. Chosen	Chosen vs. Min
4	6.56	11	6.25	6.249	4.96%	-4.72%
5	6.50	11	6.09	6.090	6.65%	-6.24%
4	5.94	10	5.78	5.909	0.44%	-2.68%
3	6.25	9	6.01	6.137	1.90%	-3.86%
3	6.51	10	6.28	6.417	1.42%	-3.44%
4	6.63	7	6.24	6.271	5.76%	-5.95%
4	6.39	10	6.07	6.151	3.80%	-4.99%
3	6.51	11	5.93	5.926	9.88%	-8.99%
4	6.26	10	6.16	6.207	0.84%	-1.57%
3	6.54	9	6.40	6.408	1.99%	-2.07%
4	6.48	10	6.37	6.493	-0.14%	-1.70%
4	6.22	4	6.22	6.299	-1.26%	0.00%
3	6.50	11	6.18	6.184	5.06%	-4.81%
6	6.10	11	5.82	5.824	4.72%	-4.51%
4	6.18	10	5.92	5.944	3.97%	-4.23%
3	6.81	8	6.40	6.698	1.59%	-5.92%
3	6.74	6	6.60	6.863	-1.84%	-2.10%
3	6.61	8	6.41	6.662	-0.72%	-3.16%
4	6.61	11	6.37	6.366	3.82%	-3.68%
3	6.53	5	6.39	6.521	0.16%	-2.10%
4	6.18	4	6.18	6.446	-4.18%	0.00%
4	6.28	11	6.13	6.129	2.44%	-2.38%
3	6.47	10	6.25	6.341	2.08%	-3.45%
5	6.29	10	6.18	6.261	0.47%	-1.77%
6	6.40	11	6.33	6.327	1.17%	-1.16%
3	6.66	9	6.56	6.617	0.69%	-1.62%
4	6.42	11	6.27	6.275	2.29%	-2.24%
5	6.37	5	6.37	6.589	-3.34%	0.00%
6	6.38	11	6.00	6.000	6.42%	-6.03%
3	6.60	11	5.84	5.836	13.11%	-11.59%
7	6.21	9	6.10	6.216	-0.11%	-1.70%
7	6.30	11	6.26	6.259	0.64%	-0.64%
8	6.19	10	6.09	6.145	0.80%	-1.62%
6	6.23	11	6.15	6.147	1.39%	-1.37%
6	6.71	11	6.22	6.220	7.91%	-7.33%
7	6.31	11	6.10	6.100	3.47%	-3.35%
11	6.28	11	6.28	6.282	0.00%	0.00%
7	6.34	7	6.34	6.345	-0.07%	0.00%
6	6.41	8	6.27	6.275	2.14%	-2.22%
6	6.47	8	6.46	6.532	-0.89%	-0.16%
6	6.14	10	6.03	6.256	-1.89%	-1.82%
8	6.12	11	5.92	5.922	3.27%	-3.17%
6	6.40	10	6.25	6.311	1.34%	-2.29%
6	6.54	10	6.14	6.212	5.22%	-6.01%
6	6.40	9	6.09	6.208	3.02%	-4.76%
9	6.48	9	6.48	6.742	-3.91%	0.00%
8	6.49	8	6.49	6.608	-1.82%	0.00%
7	6.38	9	6.35	6.465	-1.36%	-0.47%
8	6.31	10	6.18	6.251	0.95%	-2.05%
6	6.32	9	6.21	6.409	-1.34%	-1.78%
7	6.23	11	5.96	5.962	4.44%	-4.25%
6	6.20	9	6.09	6.159	0.65%	-1.69%
6	6.36	11	6.29	6.290	1.19%	-1.17%
6	6.64	11	6.38	6.383	4.05%	-3.90%
10	6.38	10	6.38	6.497	-1.83%	0.00%
7	6.54	7	6.54	6.856	-4.67%	0.00%
6	6.64	6	6.64	6.695	-0.81%	0.00%
6	6.51	11	6.42	6.416	1.45%	-1.43%
7	6.61	11	6.03	6.027	9.75%	-8.88%
6	6.68	11	5.99	5.986	11.53%	-10.34%
9	6.17	9	6.17	6.267	-1.52%	0.00%
9	6.18	9	6.18	6.192	-0.23%	0.00%
11	6.10	11	6.10	6.100	0.00%	0.00%
9	6.21	11	6.21	6.212	0.02%	-0.02%
9	6.38	9	6.38	6.457	-1.16%	0.00%
10	6.37	10	6.37	6.449	-1.27%	0.00%
9	6.36	9	6.36	6.460	-1.59%	0.00%
10	5.87	10	5.87	5.880	-0.21%	0.00%
10	5.92	10	5.92	5.965	-0.77%	0.00%
9	6.34	11	6.27	6.265	1.15%	-1.14%
9	6.26	9	6.26	6.380	-1.86%	0.00%
9	6.33	11	6.04	6.042	4.70%	-4.49%
11	6.16	11	6.16	6.164	0.00%	0.00%
11	6.25	11	6.25	6.251	0.00%	0.00%
9	6.34	9	6.34	6.368	-0.48%	0.00%
10	6.40	10	6.40	6.575	-2.64%	0.00%
10	6.39	10	6.39	6.460	-1.15%	0.00%
11	6.00	11	6.00	5.995	0.00%	0.00%
11	6.06	11	6.06	6.058	0.00%	0.00%
9	6.19	11	6.08	6.081	1.84%	-1.81%
11	6.04	11	6.04	6.039	0.00%	0.00%
9	6.34	11	6.22	6.218	1.89%	-1.85%
10	6.47	10	6.47	6.474	-0.07%	0.00%
9	6.82	6	6.74	6.867	-0.73%	-1.07%
9	6.59	9	6.59	6.735	-2.19%	0.00%
9	6.70	9	6.70	6.741	-0.66%	0.00%
11	6.39	11	6.39	6.385	0.00%	0.00%
10	6.37	10	6.37	6.557	-2.87%	0.00%
11	6.18	11	6.18	6.175	0.00%	0.00%
11	6.13	11	6.13	6.134	0.00%	0.00%

Figure B.1. RMSE changes between iterations and the selected iteration number before the RMSE increase with the RMSE comparisons between alternative designs.

APPENDIX C: DESIGN OF EXPERIMENTS FOR PROCEDURE VERSION COMPARISONS

Figure C.1. The design of experiments for comparison of procedure versions.