Effects of Missing Data on Tree Based Ensemble Multitask Learning Method

A study submitted in partial fulfilment
of the requirements for the degree of
Master of Science in Data Science

at

THE UNIVERSITY OF SHEFFIELD

by

Hasan Can KARAPINAR

September 2018

Name: Hasan Can Karapınar

Registration Number : 170133340

Word Count: 10026

ABSTRACT

Background: The missing data issue is a common problem in the world of data mining and machine learning. The effect of the missing data is known, yet there are not much knowledge explains the magnitude of the performance loss.

Aims: This study aimed to investigate the performance loss occurred in tree based ensembled multitask method by comparing the imputation methods.

Methods: Tree based ensemble multitask learning models were developed and trained with the data that consists of the percent inhibition values between biological targets and molecules. The variables in the training data were deleted iteratively for representing a data set with missed values. Afterwards, these missing points are imputed with general mean, column mean, row mean, and predicted values from decision tree.

Results: Performance loss due to missing data is valid and correlated with the ratio of the missing data. Among the imputation methods, decision tree based imputation technique was distinguished by its success for contributing and overcoming the missing data issue. Column and row mean imputation methods showed a similar pattern where general mean imputation was the most different of all since it showed almost an exponential increment in evaluation metrics.

Conclusion: This study showed the effects of missing data to the multitask model and it can give approximately the ratio of missing data required to train a machine learning algorihm.

Table of Contents

TABLE OF FIGURES

1. INTRODUCTION

Chemoinformatics consists of different fields of interests: computational chemistry, statistics, mathematics, and computer science. It is an important field for chemistry, since there is vast amount of data generated by physical experiments and by using these data sets researchers can predict the effect of drugs, reduce the number of vivisections or can predict side-effects of a drug as well as molecules toxicity and its destruction to the nature. To put these studies into practice, researches need solid models trained with relevant and adequate data sets. There are still many molecules that have not been tested and assayed yet. The aim of this study is to demonstrate how to deal with missing data to support chemoinformatics' applications and the performance loss due to general mean, column mean, row mean and decision tree imputation methods, using multitask learning on PKIS data.

In public available chemical data sets, researchers commonly encounter with incomplete data in terms of relation between a compound and biological target. Thus, models that are based on these data sets are expected to show weaker performance than the complete data sets. The aim of the study is to investigate how well multitask learning algorithms can process with missing data imputation using publicly available chemical data set. A complete PKIS data set will be iteratively deleted to represent a data set with missing values. Then, a random forest model will be constructed for prediction on chemical data. In the end, the outcome of each experiments based on imputation methods, will be evaluated and compared. According to results, the effect of missing data and the imputation methods to random forest model will be demonstrated.

This project provided an important opportunity to advance the understanding of the performance loss caused by missing data issue and the imputation techniques applied to a multitask problem. Due to time and computing power constraints, this paper cannot provide a comprehensive comparison of the response of different multitask learning algorithms to data imputation techniques. The essay has been organised in the following way.

The overall structure of the study takes the form of five chapters, including this introductory chapter. The second chapter begins by laying out the chemical descriptor types which are essential for digitalizing the chemical compounds in order to using them as a training input matrix in machine learning algorithms. In the following section of second chapter, the most known and frequently used machine learning techniques in chemoinformatics environment is presented. Then, the researches for machine learning imputation methods are

indicated. In the third chapter, the data set, multitask model, data deletion and imputation procedures, and evaluation methods are introduced. The fourth chapter presents the change in imputation values and evaluation metrics with comparing the imputation methods. Finally, the conclusion gives a brief summary and critique of findings with areas for further research. Throughout this dissertation the term R^2 will refer to the coefficient of determination, the abbreviation MAE will be used to refer to mean absolute error and the abbreviation RSMD will be used as root mean square deviation.

2. LITERATURE REVIEW

2.1. Chemical Descriptors

To enable using, manipulating, and analysing chemical structural information with computer aided systems, molecular descriptors (also known as chemical descriptors) has been created. These descriptors are numerical variables that can profile, characterise and represent molecules and their features. There are variety of molecular descriptors that have been introduced in academy and industry depending on several different purposes. This variation based on the computational requirements and the complexity of the information such as molecular weight which is easy to compute yet does not represent features in-depth, by contrast there are some descriptors based on quantum mechanics which can demonstrate more conceptual features of the molecule but requires more time for computation (Leach & Gillet, 2007). Chemical descriptors can be classified into two classes as descriptors based on 2D structure and descriptors based on 3D structure. In terms of 2D structure representation, there are descriptors such as molar refractivity, topological indices, 2D fingerprints and some descriptors that are required for 3D representing structures. 2D descriptors are easy to compute within large data sets.

Leach and Gillet (2007) stated that the simple counts of features represent the simplest way of chemical descriptors which are related to the count of “hydrogen bond donors, hydrogen bond acceptors, ring systems, rotatable bonds and molecular weight”. This method makes these features be found at present, if not, enables an easy calculation from 2D structure of the molecule since these features are substructures or molecular fragments. Nevertheless, these

descriptors are not powerful parameters that can highly contribute to classification, which is why they usually processed with other descriptors in applications.

Hydrophobicity and lipophilicity are important factors for determining biological activity and disposition of the drugs. They are widely calculated by the partition coefficient *P* or the algorithm log *P.* Mannhold and van der Waterbeemd (2001) divided the methods for log *P* into two classes: substructure approaches and whole molecule approaches.

Topological indices are numerical values that are calculated based on 2D structural representation of molecules (Randi, 2001). Primary topological indices were consist of only the information of atom connectivity which is basically counting the every bonds connecting each pair of atoms, but they were able to represent many properties of hydrocarbons effectively, nonetheless, nowadays they include not only the atom connectivity but also "the nature of atoms and the bond multiplicity" (Devillers & Balaban, 1999).

Molar refractivity is calculated by the division of molecular weight and the density, which yields molar volume, multiplied by some operations with refractive index (Livingstone, 2000).

The kappa shape indices are proposed by Hall and Kier (2007) to derive the molecular shape into data. "It is based on the count of 2-bond fragments in a graph relative to the maximum number possible and the minimum number in the isomeric linear graph"(Dearden, 2017,p.73).

The electrotopological state (E-state) indices (Kier & Hall, 1990) are the descriptors that contain the combination of both electronic and topological characteristics of a molecule. In 1995, Kier and Hall (1995) published another paper about electrotopological state in atomic level which they called "atom type E-state indices".

2D Fingerprints are one of the most common descriptors used in chemoinformatics. Initially, they were invented to boost the substructure searching algorithms instead of intention of creating novel descriptor (Leach & Gillet, 2007).

Atom pair descriptors encode all pairs of atoms in a molecule together with the length of the shortest bond-by-bond path between them (Smith, Carhart, & Venkataraghavan, 1985). Each atom is described by its elemental type together with the number of non-hydrogen atoms which it is bonded. Other 2D descriptors are Extended Connectivity Fingerprints (ECPFs) (Rogers & Hahn, 2010), which is similar to Functional Connectivity Fingerprints (FCFs) (Hert

et al., 2004), and BCUT descriptors which are “designed to encode atomic properties relevant to intermolecular interactions” (Leach & Gillet, 2007, p.64).

3D Fragment Screens were introduced for substructure searching, yet they are also used as descriptors which encode “spatial relationship between different features of a molecule” (Leach & Gillet, 2007, p.65)**.** Pharmacore Keys are 3D descriptors based on pharmacophoric functions and initially designed for 3D pharmacophore searching.

2.2. Machine Learning Approach in Chemistry

Support vector machines (also known as support vector networks) are supervised learning algorithms which is used for classification, ranking and regression analysis. It is a binary classifier algorithm with a non-probabilistic approach. Support vector machine is one the most convenient algorithm for chemoinformatics due to its classifier ability which can classify objects into two classes based on their functions (Jorissen & Gilson, 2005). One of the applications of SVM in chemistry is generated to test if the algorithm can recognize and classify the drug and non-drug compounds (Byvatov, Fechner, Sadowski, & Schneider, 2003). Results yield that SVM have scored 82% of correct predictions with high overall prediction accuracy. In another example, SVM is used for classifying molecules, preferable novel compounds, according to the activity similarity of molecules in training data (Jorissen & Gilson, 2005). Besides SVM’s nature of classification, it has also been used in ranking problems achieved by the compounds’ diverse probability of activeness. Ranking has been calculated by the distance between the desired compound and hyperplane (Agarwal, Dugar, & Sengupta, 2010). Although support vector machine is a good classifier with high accuracy and low level of overfitting, it only can yield a binary classification and requires huge computational resources with huge amount of data.

Decision trees are tree shaped decision support methods that grows with recursive binary splitting. In chemoinformatics, decision trees have been widely used to predict ‘drug likeness’, biological activities of compounds, and forming profiles of compounds. For example, privileged substructures were identified according to the relationship of the activity of compound and substructure presence for enrichment of “chemical libraries with potential drug molecules” (Klekota & Roth, 2008). Another approach was to analyse how convenient is decision trees on classifying chemical compounds with respect to their drug likeness (Schneider, Jäckels, Andres, & Hutter, 2008). It is easy to comprehend and interpret decision

tree models, nevertheless, predictions can be misdirected due to high variance, which regularly can be observed within data sets, especially small ones, that an insignificant change may yield erratic split points (Lavecchia, 2015).

k-nearest neighbours algorithm is a simple and intuitive non-parametric method commonly used for classification and regression problems. k-NN is usually defined as the simplest machine learning algorithm. Istkowitz and Tropsha (2005) modified the traditional k-NN by examining the contribution of k-NN parameters, thus weighting function is derived due to performance maximization. The modified model is applied to 5 different data sets (Artificial Toy Set, Toxicity Data Set, Anticonvulsant Data Set, Dopamine D1 Antagonists Data Set, Protein-Ligand Complexes) and demonstrated a considerable advancement. Briem and Günther (2005) applied a k-nearest neighbour algorithm, which linked to genetic algorithm for optimizing k number and weights, to "predict kinase inhibitor-likeness of compounds". k-NN method is also used to classify the cannabinoid compounds into psychoactive and psychoinactive classes and eight compounds (test data) are classified by k-NN correctly (Honório & da Silva, 2005) . Despite the fact that k-NN method is simple and intuitive, it also has some limitations. The accuracy of the classification of new compound depends on the k neighbours, therefore a noisy data may deviate the computation. To be more precise, if a training datum is misclassified, it leads the prediction on new molecule to estimate inaccurately, correspondingly, "irrelevant descriptors" will reduce predictions' accuracy, furthermore "predicted value can never be lower or greater than the minimum and maximum activity in the training set" (Lavecchia, 2015, p.325).

Naïve Bayesian classifier method is based on Bayesian theorem and "assumes that the effect of an attribute value on a given class is independent of the values of the other attributes" (Leung, 2007,p.3). Naïve Bayesian classifier applications are used in wide range of areas in chemoinformatics such as "virtual screening, the prediction of the toxicity of the compound, phospholipidosis mechanism, and protein target and bioactivity classification for drug-like molecules" (Lavecchia, 2015, p.323). In-silico target prediction of bioactive molecules has been held by Naïve Bayesian classifier with 894 human protein and more than 155000 ligand proteins (Koutsoukas et al., 2013). In the same study Bayesian algorithm yielded 80% accuracy on predictions. In another study, an alternative Bayesian network model proposed and compared its performance with the traditional Bayesian network algorithm. This study's aim was to tackle the problem of traditional Bayesian networks poor performance on similarity-based virtual screening of molecules with "structurally heterogeneous sets of actives" (Abdo, Chen, Mueller,

Salim, & Willett, 2010). Nevertheless, there was no significant divergence among performance results of algorithms which are tested by 3 different data sets. Naïve Bayesian classifiers are easy to train, works solid even if there are unnecessary features in data. However, it can only work with variables that has no dependency among each other.

Artificial neural networks are most common machine learning algorithm that can be encountered in literature of soft computing. It has been widely used in image and voice recognition, pattern recognition, prediction, detection, classification and clustering in many different fields. Nowadays, it is becoming more and more popular in chemoinformatics' applications such as pharmaceutical research, drug delivery and pharmacy curriculum (Sutariya, Groshev, & Pathak, 2013). Genetic neural network is built on molecular similarity indices and worked on steroid data set of corticosteroid binding globulin (CBG) (So & Karplus, 1997). In another study, neural networks were used to predict the toxicity of chemicals with a four level of hierarchical method which contains topostructural, topochemical, geometrical, and quantum chemical indices (Basak, Grunwald, Gute, Balasubramanian, & Opitz, 2000). They found out that topochemical indices play an important role in boosting the prediction power of the algorithm. No matter how popular the neural networks are, they also have some drawbacks. Artificial neural networks' parameters are hard to understand, and they are needed to be optimized by using other algorithms or by trial and error. It is a kind of black box method that researchers cannot manipulate or evaluate the outbound of the iterates. The most important disadvantage of the neural networks is that it encounters with the risk of overfitting.

Until last decade, machine learning algorithms can only be used for a single task at a time. To deal with this limitation, multitask algorithms were developed and implemented in many different fields of computing. Most common multitask learning methods are random forests and deep neural networks. Random forests are an ensemble method that build upon decision trees. Deep neural networks are very similar to artificial neural networks with algorithmic procedure yet differ with its multiplex hidden layers. A novel tree based ensemble multitask learning method, which is built on "Extremely Randomized Trees", was proposed to minimize the negative transfer among binary decision trees for classification and prediction (Simm, Magrans, Abril, & Sugiyama, 2014). They implemented the new method on two different well-known multitask benchmark datasets and results showed that the model can interpret better results than state-of-art methods. Deep neural networks and random forests methods were compared using large diverse QSAR datasets (Ma, Sheridan, Liaw, Dahl, & Svetnik, 2015). Study showed that deep neural network algorithm has lots of parameter to be

adjusted, yet there is no need to optimize all the parameters for all individual sets of data. Thus, they fix up on the parameters and used them in all datasets, so that deep neural network model outperformed random forests in most of the data sets. Recently, a deep neural network algorithm tested and implemented in DeepChem for publicly use. The aim of the study is to overcome both the complexity of implementing deep neural networks into software and mis anticipate the solid performance of multitask deep networks (Ramsundar et al., 2017). In terms of toxicity prediction, a deep learning based model, which is called "DeepTox", was developed to enable multitasking learning for computing all the toxic effects in one neural network (Mayr, Klambauer, Unterthiner, & Hochreiter, 2016). "DeepTox" method, then, compared with other complementary methods (support vector machines, random forests and elastic net) and achieve the best performance among all. Despite all benefits and contributions, multitask learning methods have some limitations. Deep neural networks require exorbitant computational power and usually operated on powerful costly graphical processing units (GPUs).

2.3. Imputation with machine learning

In academy, there are several imputation methods that have been introduced. Most known and used methods are mean, median or random value substitution. However, recently the interest and the number of researches in imputation with machine learning has increased. This growing interest can be related to the general attention to the field of machine learning. One of the first approach to the imputation with machine learning was a research that used two machine learning algorithms for imputation and compared the performances (Lakshminarayan, Harp, Goldman, & Samad, 1996). For this comparison, they built supervised and unsupervised model which are decision tree-based classification and a model that built upon Bayesian classification theory, respectively. At the same year with the research mentioned above, a novel machine learning imputation method and framework was introduced (Nordbotten, 1996). Nordbotten used artificial neural networks for imputation but for each variable, he created a new model which is time consuming and out of sync with the nature of machine learning. Nevertheless, a new framework for imputing with using artificial neural networks was introduced (Silva-Ramírez, Pino-Mejías, López-Coello, & Cubiles-de-la-Vega, 2011). In this study, for building models to predict each missing variable, they built models for each attribute. In other words, artificial neural network was processing on the matrix horizontally. They have split the training data set into two subsets; training and test set. The training subset contained the observed values and test subset contained missing values. A similar approach to this framework is used in this thesis. Jerez et al. (2010), published a research that compares

statistical and machine learning imputation methods in a real breast cancer problem. The imputation techniques, that were compared, are mean, hot-deck, and multiple imputation for statistical methods whereas the machine learning methods are multi-layer perceptron, Self-organising maps and K-nearest neighbours. The machine learning models outperformed the statistical methods in the research and the best results yielded from k-nearest neighbours. Rahman and Islam (2013), proposed a hybrid technique for imputing data with machine learning. They have combined decision tree method and expectation maximization (EM) algorithm. Then applied their model to two different publicly available data sets and evaluated the performance of the imputation technique. Another decision tree based imputation method was developed and proposed again by Rahman & Islam (2013). This time they used decision trees in horizontal segments since they implied in that research "records belonging to a segment have higher similarity and attribute correlations" (Rahman & Islam, 2013, p.51). According to the similarity and correlations, they have calculated the imputation values and applied their framework to 9 publicly available data sets. Folguera, Zupan, Cicerone, and Magallanes (2015, p.146), generated self-organizing map based imputation technique in order to "predict physicochemical parameters of water samples". They have compared the SOM prediction results with professional criteria and found out that SOM is considerably effective and time saving technique for imputation in experimental data sets.

3. METHODOLOGY

In this study, a complete matrix of chemical data will be deleted progressively to represent an incomplete data set. Random forest models will be constructed to work with the incomplete data sets and predict inhibition value based on the model. Afterwards, each imputation technique's results will be examined by to explore the effects of missing data.

For this study, it is suitable to use PKIS data set because all the inhibition values between compounds and biological targets are present in the data set, in other words there is no missing data and a fully loaded data matrix can be generated with using PKIS data set. Thus, it can provide a better practice and insight to the missing data problem for multitask learning method in chemoinformatics.

Deep learning and random forest algorithms outperformed single task methods in chemoinformatics' researches (Xu, Ma, Liaw, Sheridan, & Svetnik, 2017). That's why, the number of studies conducted with these two algorithms showed a significant growth. For that

reason, it is important to investigate the performance loss of multitask algorithms with missing data. Since deep neural networks require strong software skills and deep understanding, random forest will be used in this study due to its computational efficiency and robustness (Ramsundar et al., 2017).

3.1. Data Set

The data set that will be used in this project is publicly available PKIS data set which can be found at https://www.ebi.ac.uk/chembldb/extra/PKIS/. The PKIS data set contains 367 compounds with 460 kinase assays with the inhibition activity value as percent. There are some publications about the importance of this open-source data set (Dranchak et al., 2013), and the motivation behind sharing this data publicly (Knapp et al., 2013). As a chemical descriptor, Extended-connectivity fingerprints (ECFPs) is used to digitalize the molecules. They are "developed fingerprint methodology explicitly designed to capture molecular features relevant to molecular activity" (Rogers & Hahn, 2010, p. 742). There are 3 stages to find the ECFP of the molecule. In the first stage, the atoms in the molecule are given an integer identifier. In the second stage, atoms collect the identifiers of its atom neighbours and creates substructures. In the last stage, duplicate identifiers are removed and remained data set forms the ECFP fingerprint. ECFP fingerprints are easy to generate and in terms of drug activity prediction, they can make a huge contribution to the models (Rogers & Hahn, 2010). Afterwards, the data is processed to be eligible for multitask learning. Data is split into training and test sets with 3:1 ratio.

3.2. Random Forests

Random forest is a supervised machine learning algorithm that first introduced by Tin Kam Ho (1995). It is an ensemble learning method with corporation of decision trees. After Ho's approach Breiman (1996) implemented his "bagging" approach to random selection features and published a novel model (Breiman, 2001). The aim of the bagging method is to boost the performance of the algorithms by combining the learning models. Random forests can be used in both classification and prediction applications like most of the other machine learning algorithms. The main difference between decision trees and random forests is that random forests select the split points and variables randomly to build several decision trees. Random forests are easy to develop since the parameters are easy to understand and default parameters are usually yielding a good performance. Although they do not require a long-term

development phase and parameter optimization, they often need a huge computational power which still depends on the number of decision trees it built.

3.3.Particle Swarm Optimization Algorithm

Particle swarm optimization (PSO) is first introduced by Kennedy and Eberhart (1995). PSO is a heuristic search method that aims to optimize an issue or a function by using data points called as swarm which are roaming on the continuous function iteratively (Shi & Eberhart, 1999). Lu, Zhou, He, and Liu proposed a framework for optimizing the parameters of Support Vector Machine using "short term load forecasting of electric power system" (2009, p. 285). Their framework inspired me to conduct an analysis in this project. In this study, particle swarm optimization is given hyperparameters of the random forest algorithm to optimize it based on the corresponding mean absolute error value. Since the mean absolute error value should be lower for better model, PSO was established to minimize MAE score. The parameters for PSO was chosen as in swarm there are 20 particles, maximum iterations are set to 50 and the minimum step size was given as 1. Particle swarm optimization is applied to the model by using Python with pyswarm library.

3.4. Data Deletion

The main objective of this thesis is to investigate the impact of missing data to the performance of the ensemble tree based multitask learning method. Likewise, performance loss is absolute by virtue of missing data. However, there are limited number of researches, focuses on the question "how much". Therefore, the aim is to demonstrate the performance loss caused by increasing rates of missing data. Since the PKIS data matrix is a complete data set, a data deletion process had been developed to assign 'NaN' data type (most of the programming languages and tools read missing cells as 'NaN' type) to the observed values in order to represent missing data points. Data deletion process is only applied, when the initial data set was split into training, test along with input and output (target) matrices. With reference to the interest of the study, data in training target set was deleted using the data deletion function where it consists of the percent inhibition values between assay and biological target which can be blank because of not being tested. Test set of output remained complete and no process was applied to it.

Deletion function is generated manually using Python, basically the function takes a data matrix to delete data and returns the matrix with missing values. Therefore, data points in the activity matrix is randomly chosen and deleted according to the deletion rate. The deletion rate starts from 0% and continue until it reaches to %95 of the total data points in activity matrix. After 95% of the matrix is deleted, one or more rows and columns remained empty which disables some of the imputation techniques examined in this paper. That's why, iterations stop at 95%. Furthermore, iteration step size is 1% which means deletion function iterates 95 times. Selection and deletion of data points are not dependent with previous iteration. In all iterations data is selected and deleted starting with a full activity matrix.

3.5. Data Imputation

In this study, several imputation techniques were used and compared. From most basic and well-known techniques to more advanced imputation methods were implemented and tested the contribution level to the multitask ensemble tree learning method. These methods are mean substitution, imputing with regression, and similarity weighted imputing. In terms of mean substitution, empty data spots are imputed with the mean of whole activity table, mean of the row and mean of the column.

3.5.1. Imputing with General Mean Substitution

First imputation technique to investigate, is the substitution of the missing data points with the general mean of activity training matrix. After assigning 'NaN' values to the cells, imputation algorithm searches and detects these values in the matrix. The function that is built manually, come over each cell within activity matrix to both save the locations of 'NaN' values to an array and collects and cumulates the observed values. After cumulating all the data points in the matrix, according to count of non-missing values, it divides the cumulative value to the count in order to obtain general mean. Thereafter, using location array of 'NaN' values, function assigns the general mean to each cell with 'NaN' value.

3.5.2. Imputing with Mean of Assay Substitution

Second imputation method in terms of mean substitution for sparse data is mean of assay substitution. In this method, Imputer function under sklearn preprocessing library is used. Imputer is fast, easy to use and reliable function commonly referred by data scientist and analysts in python pre-processing. Imputer function can only work on data vectors but can

impute both vertically and horizontally. In other words, the function either can impute the mean in accordance with the row or the column. In the present case, the imputer function is called to work on finding and assigning values by calculating the mean of the row where cell lays within.

3.5.3. Imputing with Mean of Target Substitution

Third method is assigning mean of the columns to blank data spaces which are the mean inhibition values of per targets against every compound. This method is almost identical to the mean of assay substitution with a single difference. The difference is the direction of the mean computation. 454 targets distributed among the columns and each of them forms a 1x367 vector, consists of inhibition values.

3.5.4. Imputing with Machine Learning

The most advanced method for imputation used in this research is machine learning imputer. A decision tree algorithm is formed for predicting missing inhibition values in training activity matrix. In literature, decision tree imputation (DMI) method has been proposed and discussed in several studies. However, these techniques are built upon data sets with multiple attributes and creating regression or classification models within the training input matrix. That's why the introduced DMI methods are not applicable for this study since the input matrix consists of the fingerprints of the assays, which fingerprints can be obtained via tools and libraries and as a body they are unique for each assay. For this reason, the attributes of the fingerprint matrix cannot be an interest of imputation. Aim is to create a full matrix of inhibition levels between each target and assay. Owing to the fact that the training matrix can be separated with respect to target type and transformed into a single task learning output vector, a machine learning algorithm may be easily established for imputation. For each target vector, a novel decision tree model was created. Since the number of missing values for each column is random, gathering training and testing target sets and learning process of the methods will differentiate. Hence, the models for each target must train and predict the imputation values separately; specifically, they work column by column.

The function has three inputs: the sparse data set, training set of fingerprints and the array that the location of the assigned 'NaN' values are saved. First, the function finds and sorts the unique elements of the location array's attribute for column index and assign the unique values to a new one-dimensional array for building a loop. The loop is created for roaming and stopping by every column that holds one or more 'NaN' values. Then, the row index of the missing values in that specific column are saved to an array. This procedure helps splitting the activity

and the fingerprint matrix into training and target data sets. Moreover, it facilitates allocating the predicted values to the sparse data matrix. After obtaining the row index array, cells without 'NaN' values form the training output set where the corresponding rows of the fingerprint matrix was split into training and test sets. Ultimately, the decision tree model was trained and predicted the missing values and assigned the predicted values to the basis spot of the activity matrix.

3.6.Evaluation Methods

For evaluating multitask ensemble tree learning method, three evaluation methods are used to calculate the performance. Since the ensemble tree learning is conducted for prediction, the evaluation methods are chosen accordingly. The coefficient of determination (R^2), the mean absolute error (*MAE*) and the root mean square deviation (*RMSD*) were calculated for evaluating the performance. The coefficient of determination predicates the goodness of fit comparing the predictions and observed values. The mean absolute error evaluates the model with calculating the mean distance of the predicted value with its corresponding observed value. Root mean square deviation evaluates the density of predicted values around the best fit line of exact values. All evaluation metrics used in this research are implemented to the algorithm from the scikit learn library. For plotting the evaluation methods, matplotlib library is imported and for distribution figures seaborn library is used. Both libraries are available for Python.

3.7.Ethical Aspects

In this study, I will use PKIS data set which is publicly available online at CheMBL. Data does not contain any private information belongs to an individual. It only consists of information of chemical compounds, their id's, structure representations and SMILEs with their effect to biological target and the information about biological target itself. Therefore, this study carries "No Risks" with respect to ethical concerns.

4. RESULTS AND DISCUSSION

In this chapter of the thesis, the results obtained from the multitask method are demonstrated, compared and discussed in accordance with the imputation methods.

4.1. Parameter Search with Particle Swarm Optimization

In this section of the thesis, the results of parameter search conducted by particle swarm optimization on random forest is demonstrated. Particle swarm algorithm was processed on the random forest algorithm in 4 runs with same parameters for PSO. Swarm size was set to 20, maximum iterations was set to 50 and min-step was 1. PSO search on the random forest model with respect to the mean absolute error score. The table below demonstrates the best results were obtained and in which iteration the best results were obtained for each run.

Table 1. Particle Swarm Optimization

Best Result (MAE)	Iteration No	Number of Estimators	Maximum Depth	Minimum Number of Samples for Split	Minimum Number of Samples for Leaf	Minimum Weighted Fraction
5,264	5th iteration	3929	2964	2	1	0,46
5, 221	14th iteration	5506	7656	2	1	0,31
5,223	5th iteration	1143	4369	2	1	0,46
5,132	12th iteration	1336	511	2	1	0,47

In the first run, best results are obtained when the number of estimators was 3929 and maximum depth was 2964 at the same time the minimum weighted fraction found as 0,46. PSO found the best result at the 5th iteration and until the end swarm leader was not changed. The best score in the first iteration was 6,89 as MAE, yet just one iteration after a new best result is found which was 5,56. The second run yielded the second-best result among five runs. It calculated MAE as 5,221 at 14th iteration and stayed same till the end of the process. The number of estimators was 5506, where maximum depth and minimum weighted fraction were assigned 7656 and 0,31 respectively. In second run, the starting point was 7,23 until the 3rd iteration where the mae score was calculated as 5,56 and at 7th iteration the mae score was 5,224. Third run found the best result just in 5 iterations where number of estimators was set to

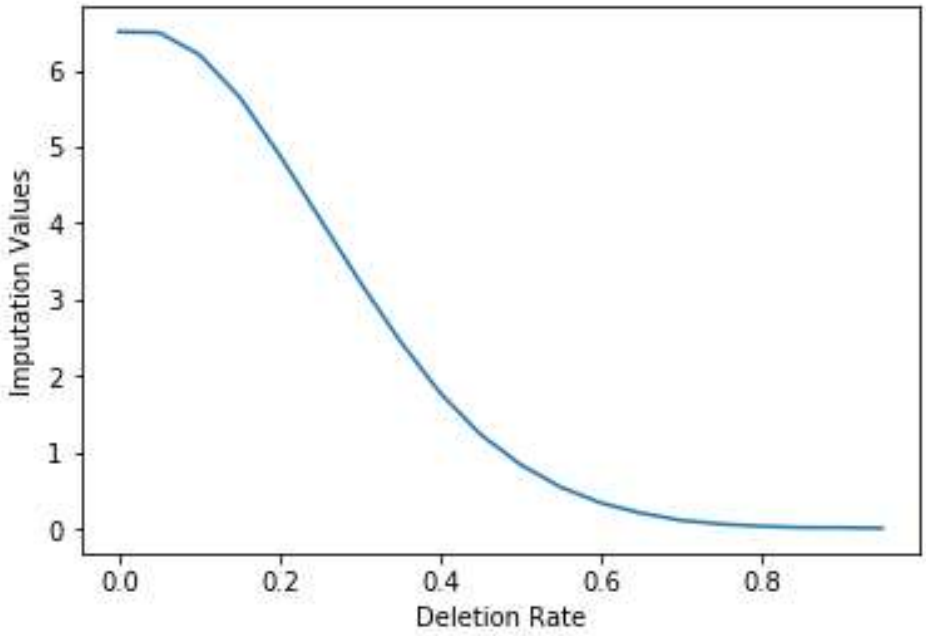
1143 with maximum depth as 4369. The minimum weighted fraction was 0,46 and the mae score was 5,223. The best in the first iteration was 7,14 and changed in the second iteration to 5,53. Between the 5th and 2nd iterations, the swarm leader was the same. As can be seen in the table above, last run showed a significant success and yielded the best result in all runs. It calculated the optimized MAE score as 5,132 with hyperparameters of 1336 number of estimators, 511 maximum depth and 0,47 minimum weighted fraction. In the first iteration the mae score was 7,15 and in second iteration the mae score was 5,28. Swarm leader had changed in 5 iterations until it found the best of all iterations. In 3rd, 6th, 8th, 9th and 11th iterations, new bests were found, and mae scores varied from 5,253 to 5,139.

There is no significant change between the best results founded in each run. However, the hyperparameters used in the random forest did not yielded a stable behaviour where the best results are yielded. The most surprising aspect of the PSO is that there is no significant correlation between the performance of the algorithm vs. the number of estimators and maximum depth. Since, the second-best run required highest number of estimators and maximum depth while the best result had the lowest parameters. In terms of efficiency, more estimators and deeper forests requires more computational power. The random forest experiments with default parameters (number of estimators is 10, and the max depth is calculated along the model) yielded mae score around 5,45. In that case, the imputation experiments were computed with default parameters because multitask models with optimized parameters requires long-time to compute even a single model where in this thesis there were 400 random forest models were processed in each experiment run. Overall, random forest algorithm can be considered as a robust technique to its changing hyperparameters.

4.2. General Mean Imputation

In this section, the imputation values of general mean and the evaluation of the effect of imputation to multitask method is demonstrated.

Figure 1 General Mean Imputation Values



After the data deletion process was finished, the general mean was calculated the remaining data points in each iteration. In other words, the calculation is unique for every experiment and every iteration since in this thesis the aim is to simulate missing data with random. The figure above shows the change in general mean value per iterations. The first iteration represents the complete training activity data matrix in which the general mean is around 6,50 (depends on the split, the value is average). After the 10% of data deleted, general mean values started to decrease rapidly. Around 30% of missing data ratio, general mean value dropped to half to 3,23. At 47^{th} iteration, general mean fell below 1. Around %55 of deletion rate, the slope of decreasing line dramatically declined where general mean was 0,54. In the end at 95% of data deleted, the general mean of the remaining inhibition values was 0,12.

Figure 2 Coefficient of Determination Scores of General Mean Imputation

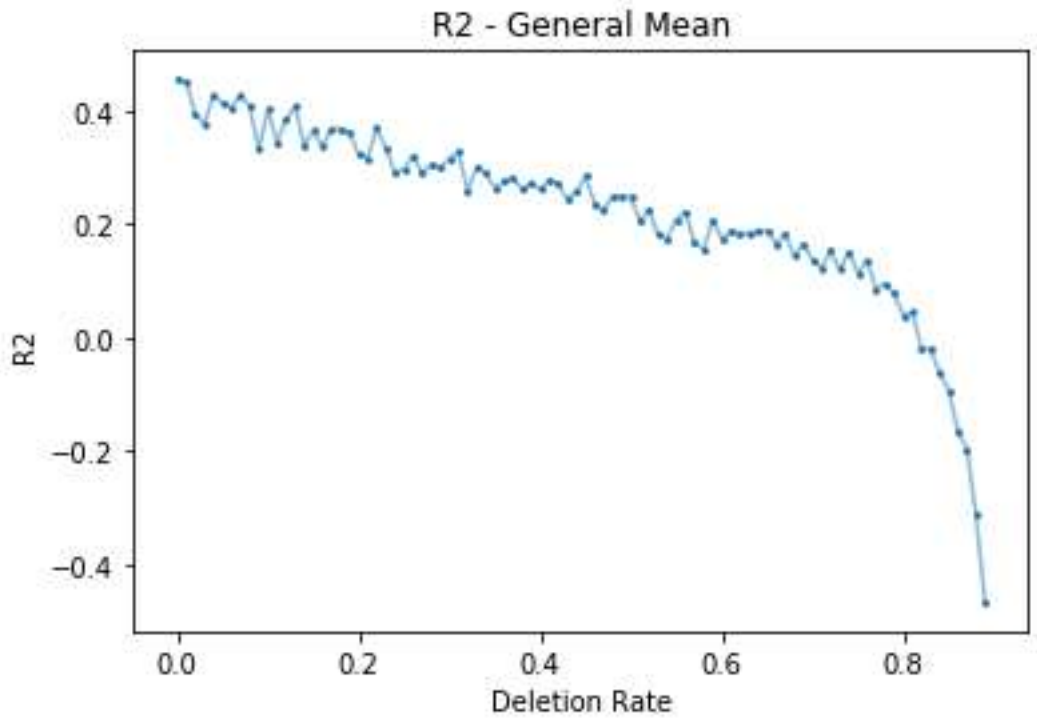
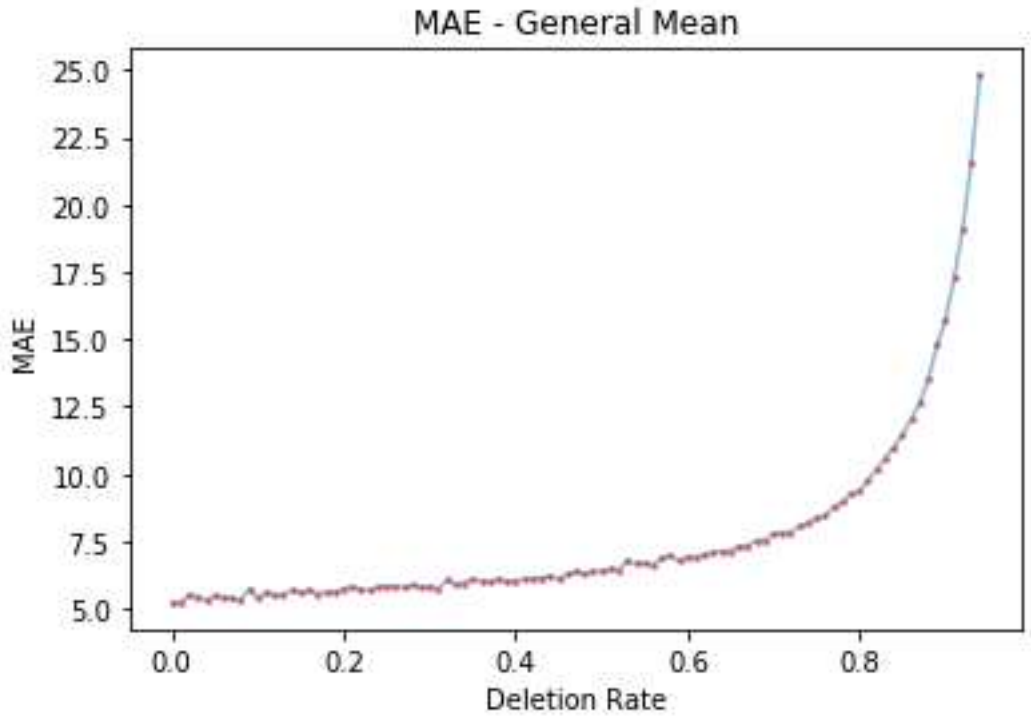


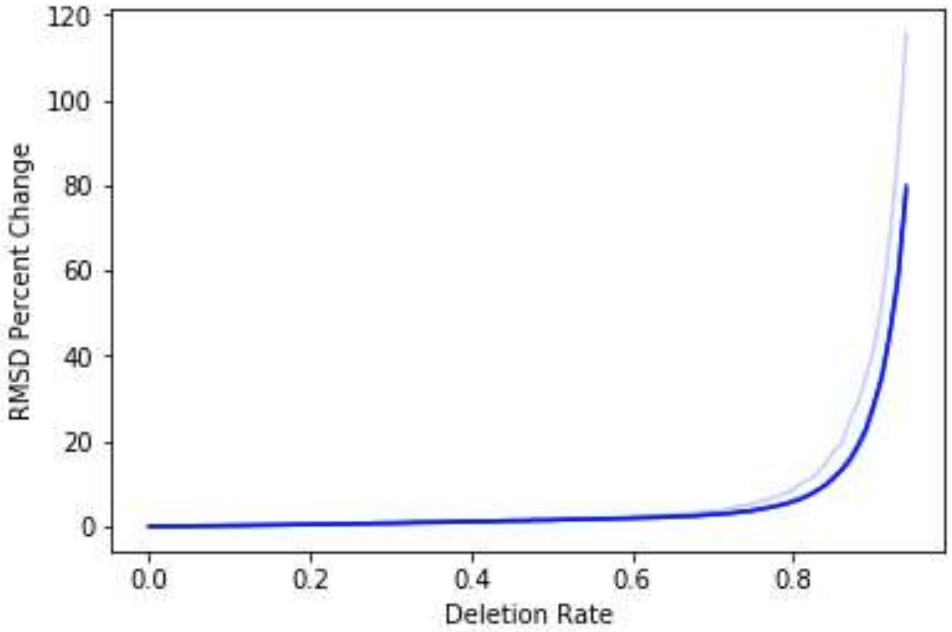
Figure 2 shows the change in coefficient of determination based on the random forest models with default parameters, after general mean imputation. Mean of R^2 scores obtained in experiments starts with 0,45 which model is trained with the complete activity matrix. As the size of missing data increased, the coefficient of determination value decreases like an exponential function. After 20% of the data deleted and imputed with the general mean, change in R^2 was observed as 28%. Until the deletion rate reaches %60, the decline of the values was nearly linear. However, when the ratio of missing data reached %60, the R^2 of the models drop figure showed an exponential decrease pattern. Overall, the coefficient of determination come to 0, when the deletion rate reached %82.

Figure 3 MAE scores of General Mean Imputation



The effect of the general imputation to the mean absolute error (MAE) is demonstrated in Figure 3. Comparing the fluctuation pattern and curve in both R^2 and MAE, the imaginary fit line of the MAE seems to be a proper exponential curve where R^2 values fluctuated on the fit line. With default parameters and complete data, MAE is recorded as 5,24. Just like coefficient of determination, mae values increased almost linearly until 60% of the data is deleted and after %60 MAE increased rapidly up to 6,94 with 32% of change. MAE value, when R^2 hit 0 at 80% deletion rate, is 10.

Figure 4 RMSD Percent Change of General Mean Imputation



The average scores of root mean squared deviation are illustrated in Figure 4. RMSD results showed a similar attitude comparing with R^2 and MAE. However, the percent change in RMSD was quite different from the other evaluation methods. The percentage change was slightly increased until 80% of the data was imputed by general mean. As can be seen in the Figure 4 after %80 deletion rate, the upward trend of RMSD percent changed rise to the occasion. In summary, RMSD score changed %1 only when the deletion rate went up to 38%, and at 62% of deletion happened the percent change in RMSD was 2%. When 80% of data substituted, RMSD had changed 5.9%. In the end, at %95 of deletion, RMSD value was more than the double of the initial score, the change percentage was 107%.

4.3. Column Mean Imputation

In this section, the results gathered from column mean imputation were illustrated. First, imputation values calculated by column mean was shown with some statistics. Afterwards, the performance loss of the random forest algorithm was displayed by three evaluation methods.

Figure 5 Column Mean Imputation Values

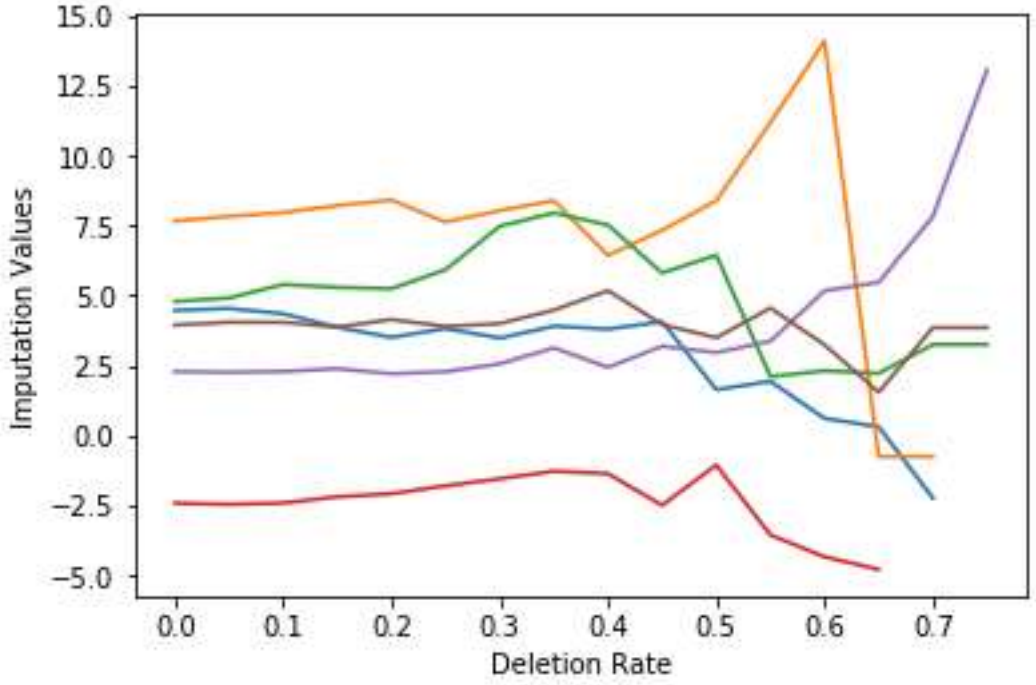
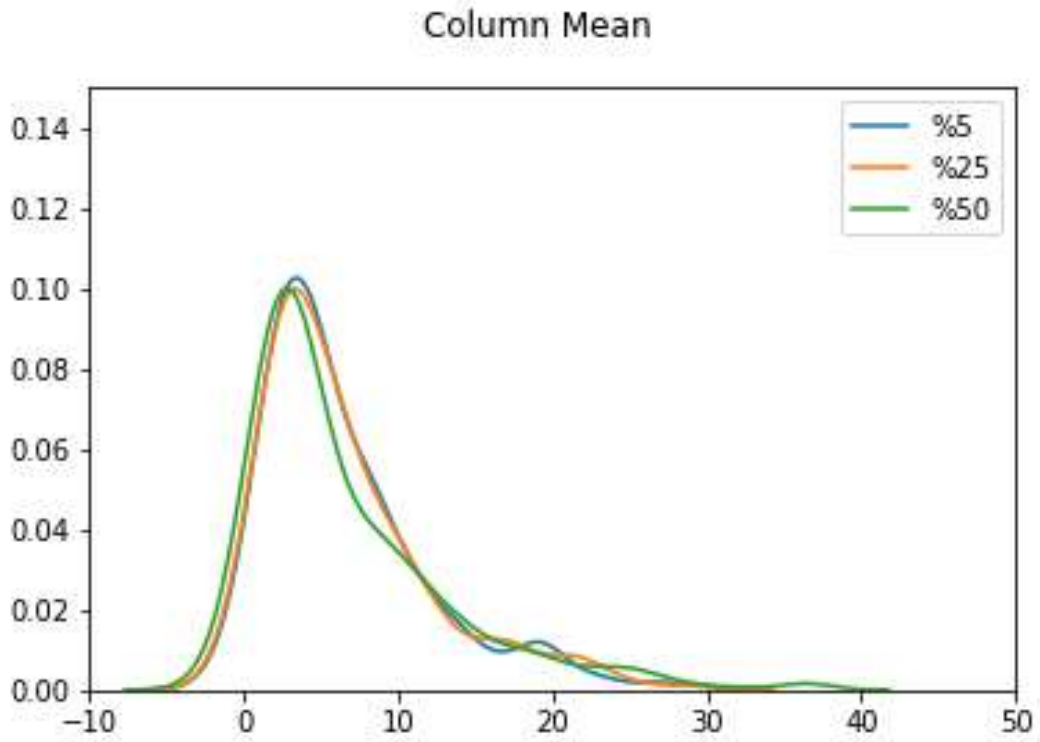


Figure 5 shows five randomly chosen columns and the change of mean of these columns along the iterations. As can be seen in the figure there's not a noticeable trend in the change of the column means. In fact, the means are changing independently since the number of missing data in all the columns are randomized. The most important aspect observed in the figure is that not all the columns are being imputed until the end of iterations. Thus, after searching among matrices and imputation values, one or more columns are being filled with 'NaN' values after deletion rate reached to %60. Therefore, these columns were being deleted and in the remaining iterations random forest models were trained with those matrices.

Figure 6 Column Mean Imputation Value Distribution



Since, there are 454 columns in the activity training matrix, it is impossible to plot every column in one figure. Thus, a distribution of the column means for 6th, 26th and 51st iterations are plotted individually and all together in the Figure 6 for a better observation in the change of means. The difference among these table is that while the iterations proceeds, the population density of inhibition values between 0 and 10 is decreasing slightly and the number of negative values is growing. Furthermore, the maximum values are also increasing with progression of deletion rate. At 5% of deletion, the maximum was between 25 and 30, at 25% of deletion, it was between 30-35 and in the end at 50% of deletion rate observed, the maximum value was between 40 and 45.

Figure 7 R2 Scores - Column Mean

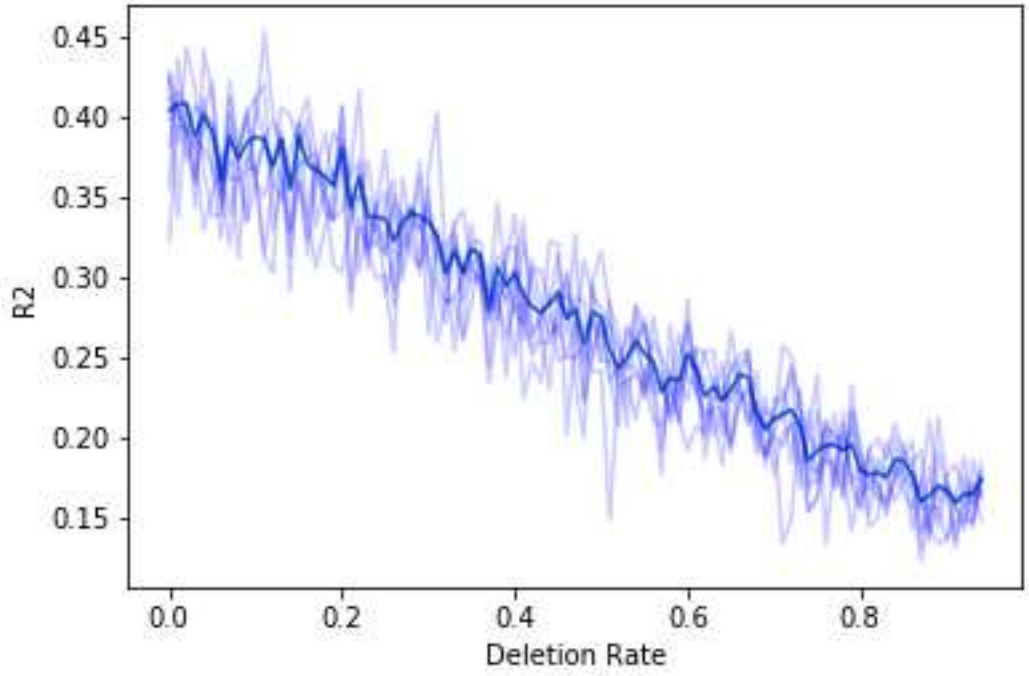
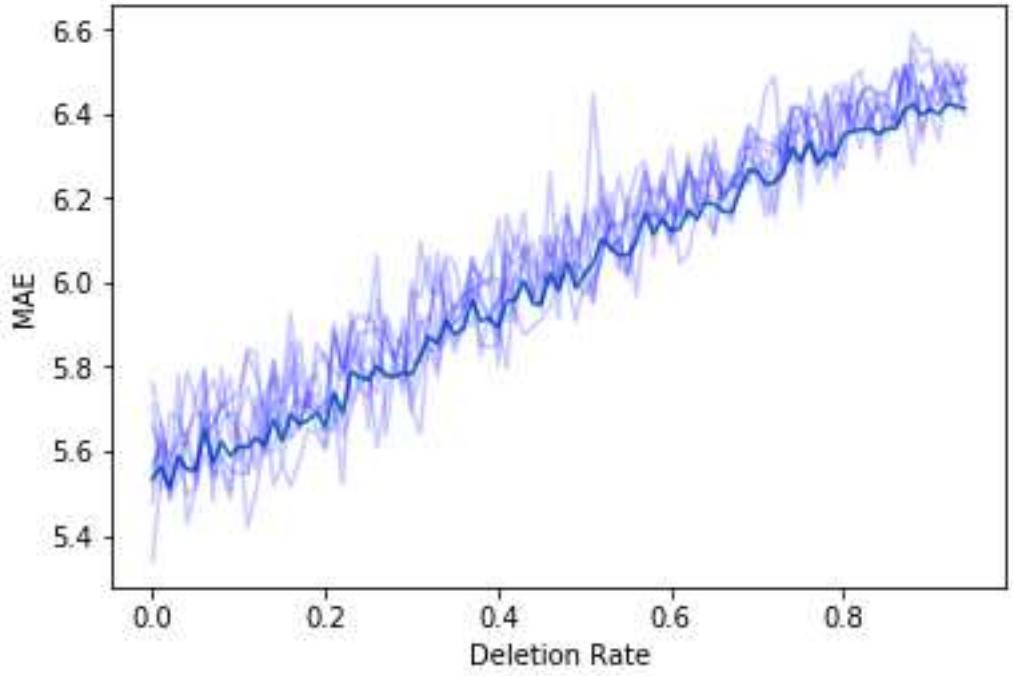


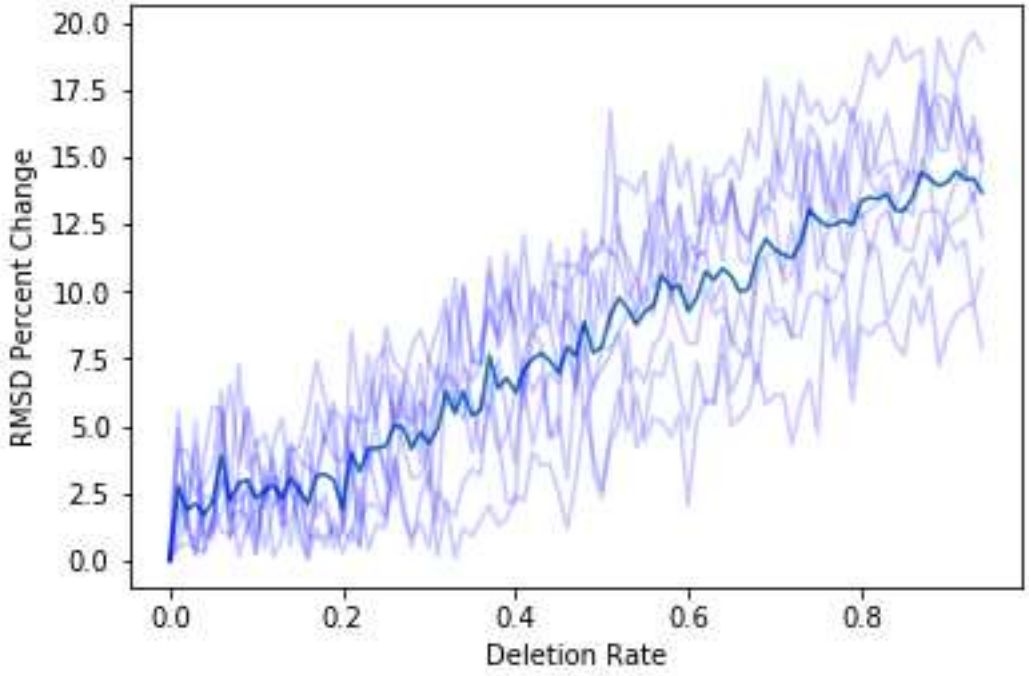
Figure 7 represents the change in R^2 where the imputation is in accordance with the column mean substitution. Initial R^2 value computed using full data set is the same with general mean. When %65 of the data is removed and imputed R^2 value drops down to 20%. Around %85 of the data deletion the coefficient of determination hits %15 and does not change significantly to the end of iterations. There is not a certain trend or pattern in the decline of the R^2 values. However, the distance between the exact error values and fit line is so short that we can say the decline happening on a linear line.

Figure 8 MAE Scores - Column Mean



The mean absolute error scores for random forest models trained with an activity matrix that is imputed with column mean based approach is illustrated in Figure 8. MAE score started with 5,24 and reached 5.8 when 40% of the data missing. At 60% of deletion MAE value raised up to 6. After 80% of the data deleted, MAE value was 6.2 and went up to 6.6 when 95% of the data deleted. Just like the imputation effect to R^2 with column mean, MAE yielded a fluctuant change along with the iterations. In spite of the up and downs among the results, there is no significant distances between consecutive MAE scores.

Figure 9 RMSD Change - Column Mean

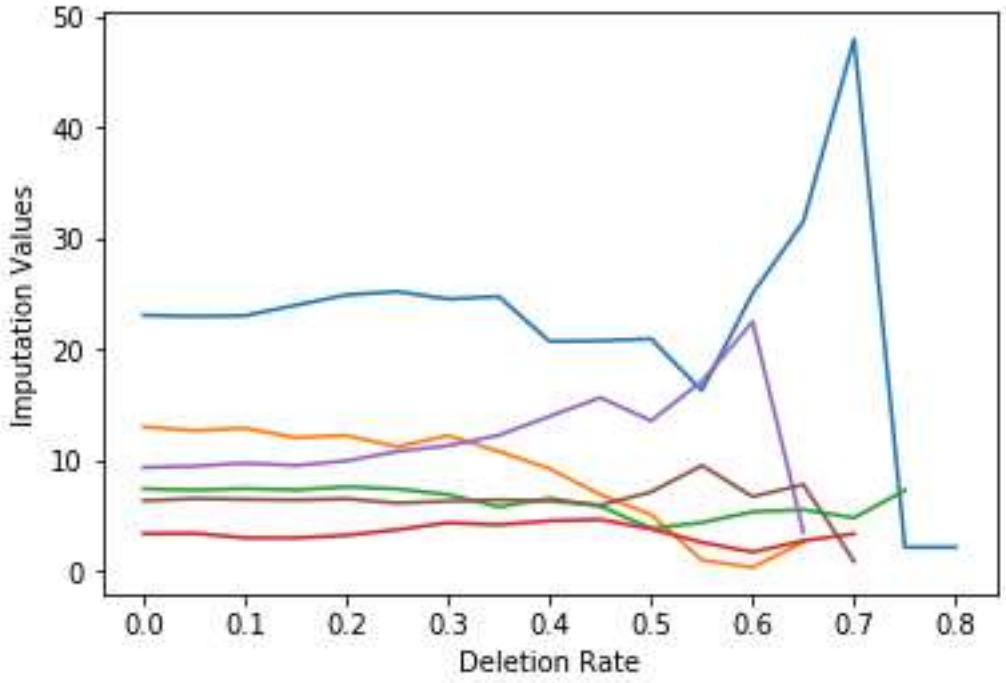


The figure above illustrates the percentage change in RMSD scores obtained from the average of all experiments for training data imputed with column-based mean. From this figure, we can see that RMSE reacted significantly to the imputation. Thus, just in the second iteration where only %1 of the data was imputed, change in RMSD was observed as 2%. The percentage change showed a fluctuant pattern until the 20th iteration where the percentage change was 2,1%. Between the first and 20th iterations, the highest percentage change observed was 3,87% at 7th iteration and the lowest was %1,7 at 4th iteration. From 20th iteration, the pattern turned into increasing format from fluctuant and stable. At the rate of 40% missing data, performance loss in RMSD was 6,78%. RMSD change reached to %10 when the data deletion rate was 57%. RMSD percent change reached the peak in the last iteration with 95% of data deletion rate where the change percentage was 15%.

4.4. Row Mean Imputation

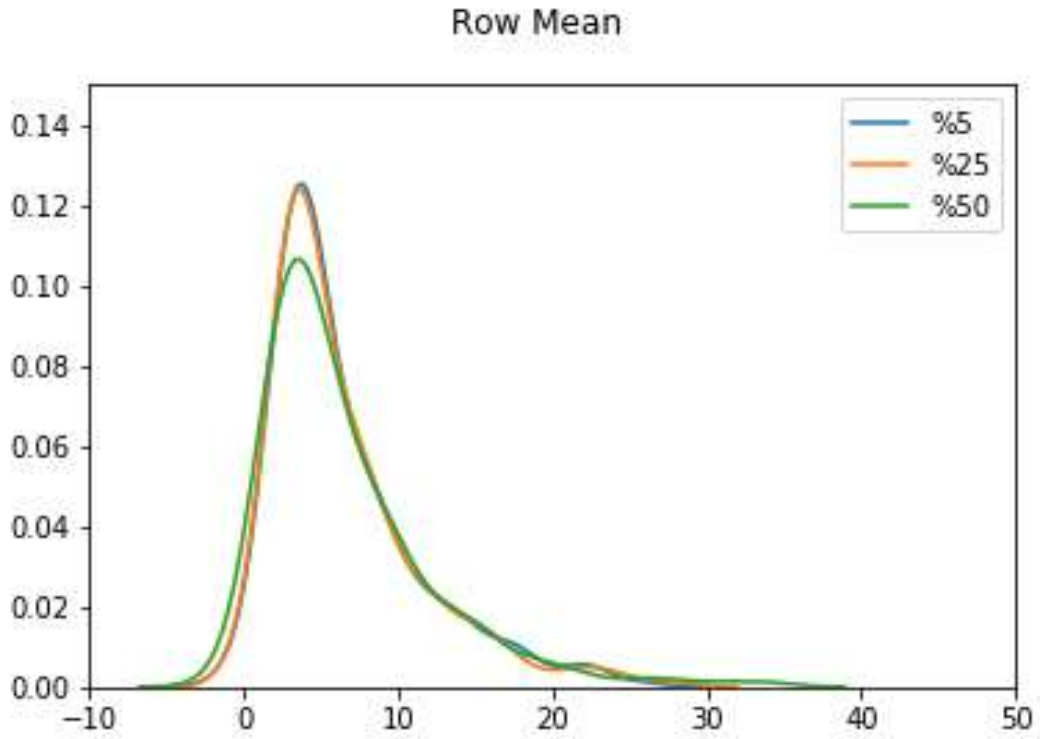
In this section, the imputed values calculated in row mean imputation methods are introduced. Then, the effect of the row mean imputation to the performance of the random forest models are investigated by plotting the evaluation scores for each iteration.

Figure 10 Row Mean Imputation Values



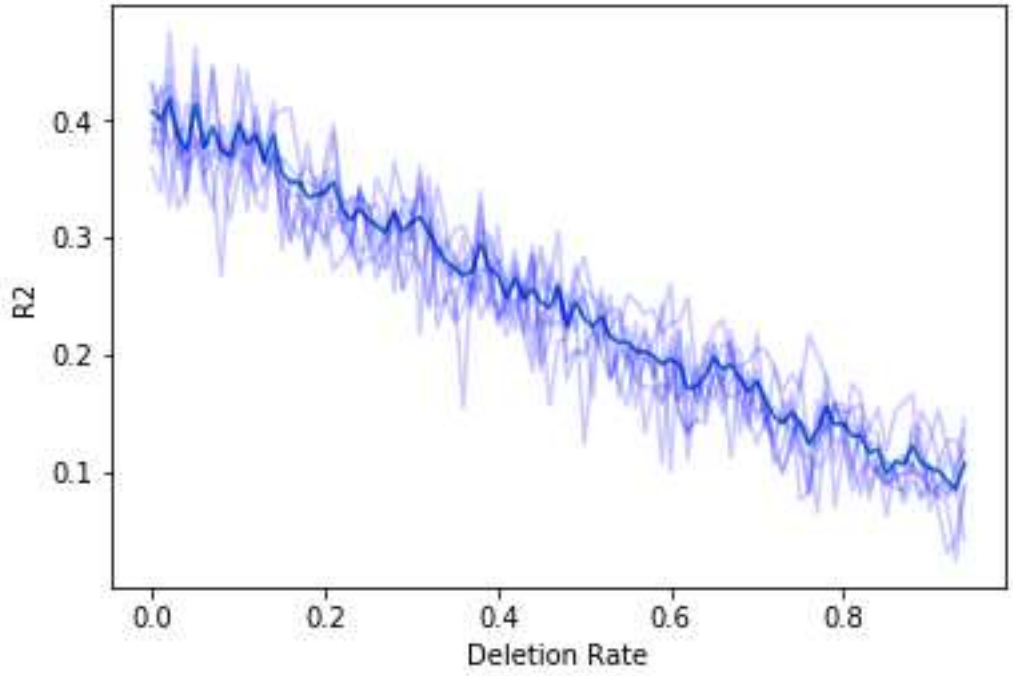
In Figure 10, the mean values for six different rows along the iterations are shown. As it can be seen clearly, there is not a significant correlation among change in means. For example, the row mean represented with red line had slightly changed where the mean of blue line showed a huge gap between iterations. Just like in the column mean imputation, empty arrays started to occur around 60% of deletion rate in row mean deletion. It is apparent from Figure that the orange and purple lines stopped around 65%, red and brown lines stopped around 70% and blue and green lines stopped around %80.

Figure 11 Row Mean Imputation Distribution



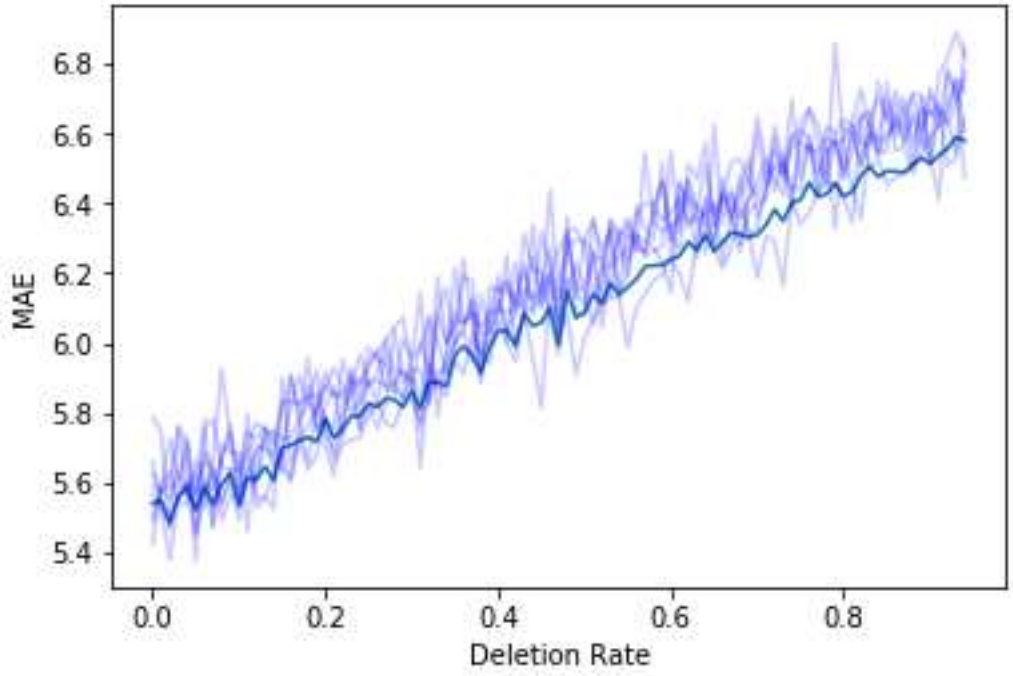
In order to show the mean variation among iterations, the distribution of the means in 6th, 25th and 50th iterations are demonstrated in Figure 11. The distribution of means in 5th and 25th iterations are considerably similar, yet when half of the data is missing, the population of row means between 0 and 10 inhibition percentage diminishes. The maximum and minimum row mean values were growing away from the median of the results as the deletion rate was increasing.

Figure 12 R2 Scores - Row Mean



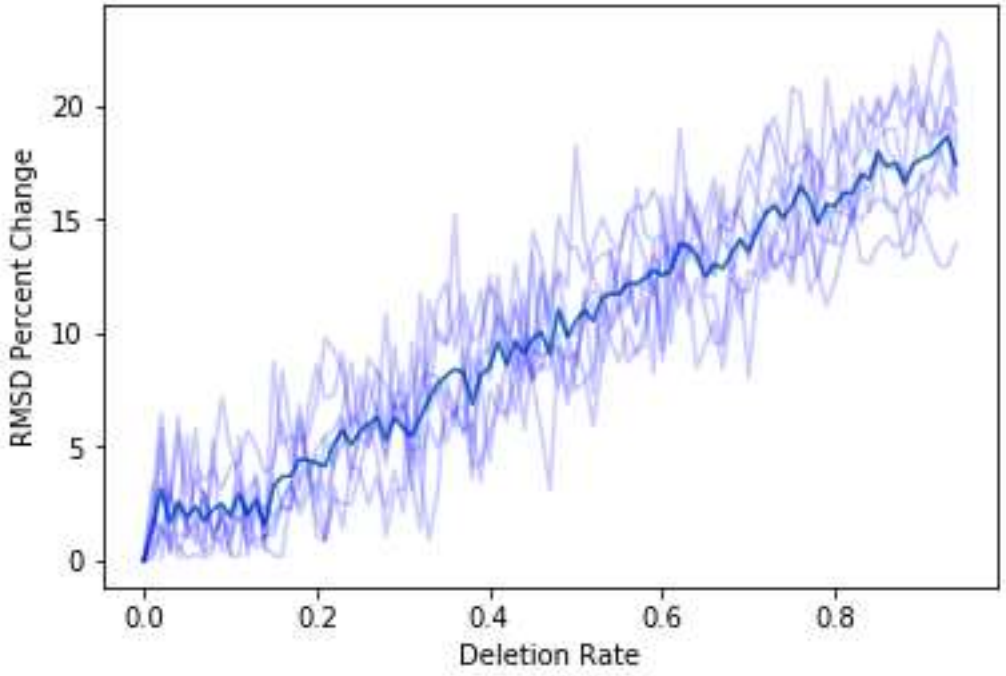
The performance loss, that is calculated based on imputation with row mean, showed a similar pattern with column mean based method in terms of R^2. Among all performance measurement figures, row mean imputation method's error values seems to be fluctuated widely. However, the percentage difference between two following data points had never exceeded %10 of a change. R^2 values of all experiments, conducted with row mean imputation, are demonstrated in the Figure 12. R2 lost %7 of the value when 20% of the data eliminated. At 40% of deletion rate, R2 was 0.28. When 60% of data deleted, the coefficient of determination dropped to 0,20. In the end, where 85% of data assigned with 'NaN', R2 value drop below 0,10 line and remained stable in the residual iterations.

Figure 13 MAE Scores - Row Mean



Row mean imputation provides a stable growth in mean absolute error. As can be seen from Figure 13, the error values changed slightly along the iterations. In response to 20% of data deletion, the mean of MAE values only increased %4,5 to 5,78 comparing to the initial model with complete data set. MAE surged to 6,0 with 40% of deletion rate, 6,5 with %60 deletion rate and 6,5 when the ratio of missing data was reached 83%. The slope of the fit line calculated from the mean MAE value of all experiments is 1,17.

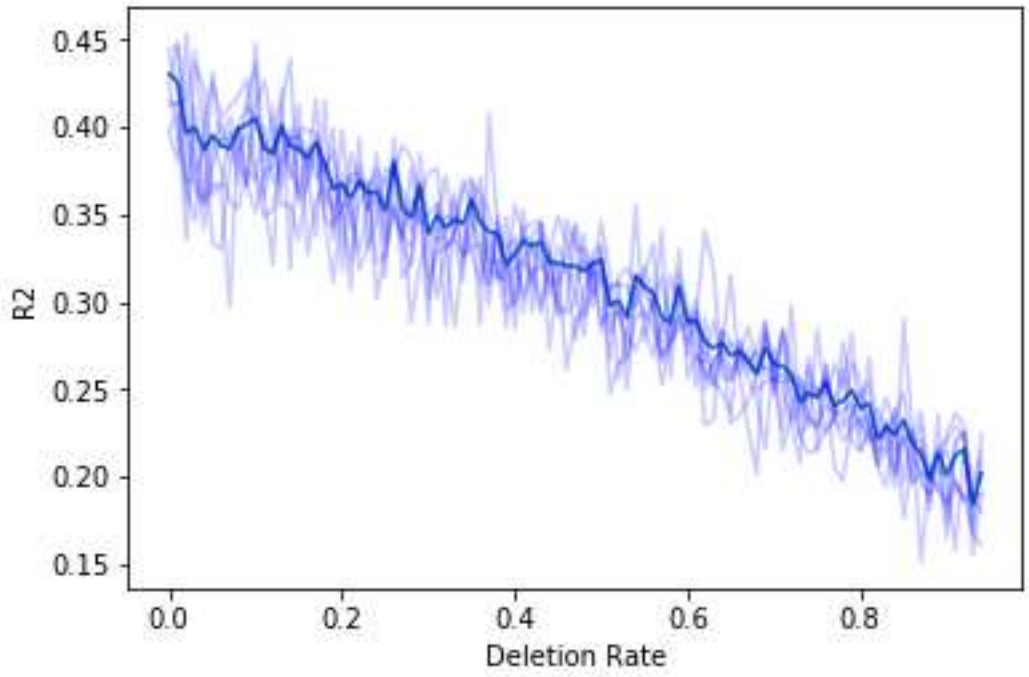
Figure 14 RMSD Change - Row Mean



RMSD percent change depends on row mean substitution, was calculated and plotted in the Figure 14. The experiment plots (light blue) showed an excursive tendency; therefore, the unstable changes affected the average line of the RMSD collection. Nevertheless, from the Figure above, it can be seen that the mean line of RMSD showing more stable pattern than experiments. In the beginning, the percentage change varied, and a significant increment was not observed. Until the 15th iteration where 14% of data deleted, the results kept the unstable behaviour. RMSD change percentage values in those 14 iterations vary from 1,44% to 3,1%. Interestingly, the minimum and maximum values within first 14 iterations was consecutive iterations which are 2nd and 3rd. In oncoming iterations, an increasing pattern was noticed from the figure. Percentage change rose to 4,26% at 21st iteration, 8,45% at 41st iteration and 12,5% at 61st iteration. When the ratio of missing data grew to %80, percentage change was 15,58%. In the last iteration, the percent change in RMSD score was recorded as 18,11%. However, the highest record was observed when the deletion rate was %93 with a change of 18,61% in RMSD value.

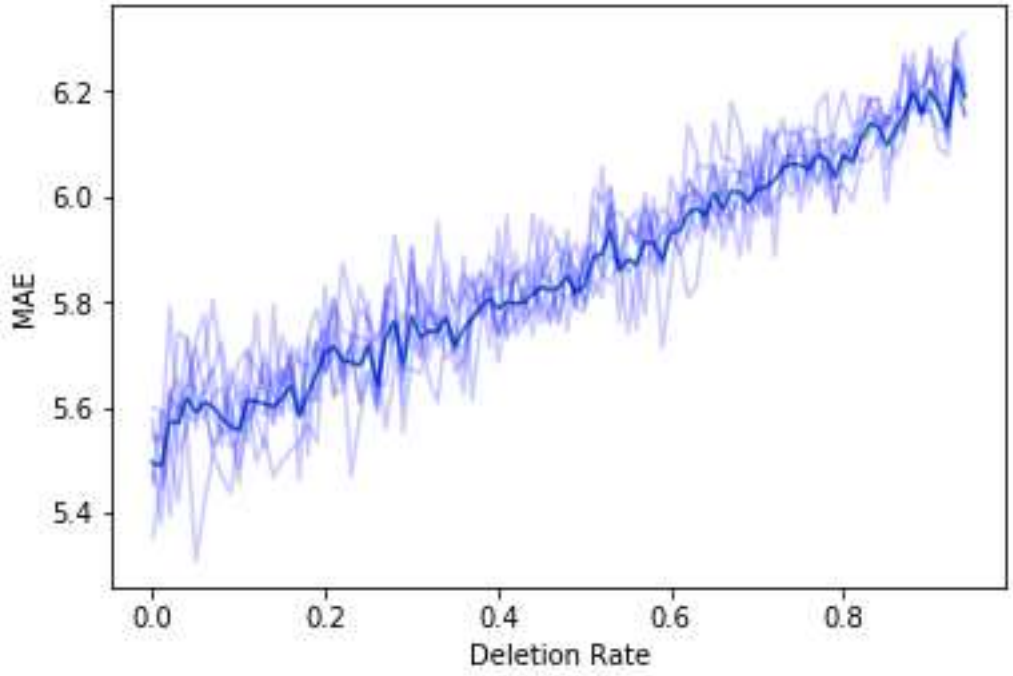
4.5. Decision Tree Imputation

Figure 15 R2 Scores - DT



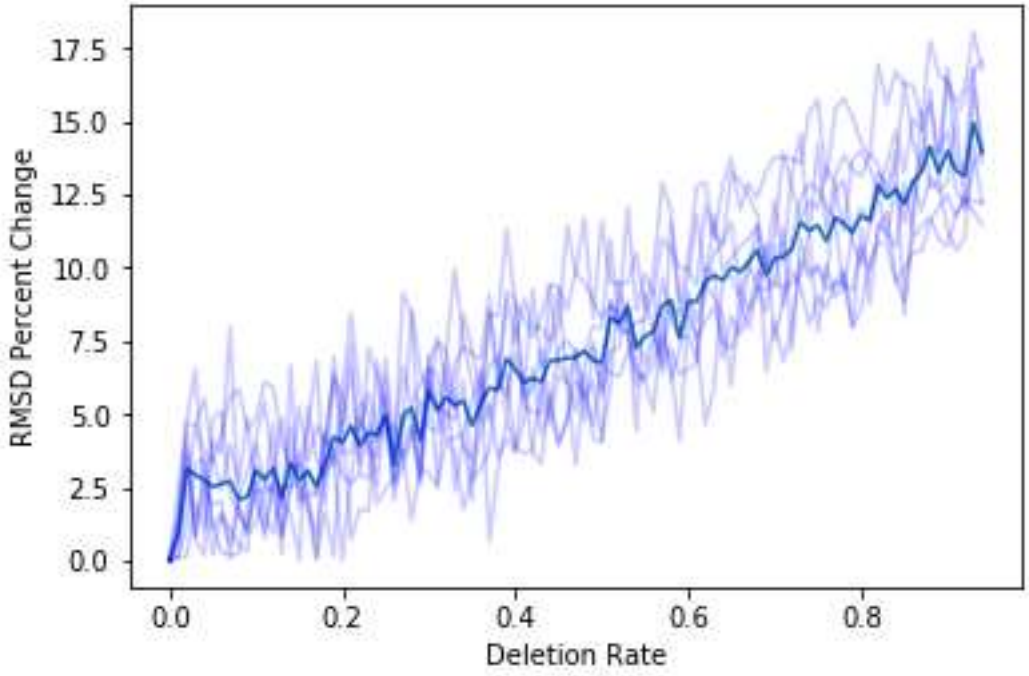
The R^2 values obtained from the imputing missing data with decision tree models are set out in Figure 15. It is apparent from this figure that decision tree models can contribute significantly to get over the issue of missing data in multitask ensemble learning method till half of the complete data set is gone missing. After 20% of the data deleted and filled with the predicted values from decision tree model, multitask model only lost 6% of performance in terms of coefficient of determination which is 0,37 at that rate. Following the addition to the deletion rate, R^2 values flatten out steadily. At 40% deletion it was 0,32, at 60% of deletion R^2 was 0,29 and at 80% ratio of missing data of complete data set is reached, R^2 was 0,24. The R^2 value hit the lowest point of 0,20 in the end of iteration where 95% of data was missing.

Figure 16 MAE Scores - DT



The mean absolute error values, that are obtained from each multitask experiments with decision tree-based imputation, are presented in the Figure 16. Initial error of the mean of absolute difference is 5,494. MAE increased by %3,8 to 5,704 after 20th iteration was completed. Thereafter, when 40% data deleted, the percent change between complete set and sparse data set was %5,3 with the value of 5,787. Finally, the change percentage and the exact values for MAE with %60 and %80 of sparse data were 5,931 with %8 change and 6,079 with %10 change respectively. The highest error rate occurred at the level of 93 percentage of deletion which yielded 6,241 MAE score.

Figure 17 RMSD Change - DT

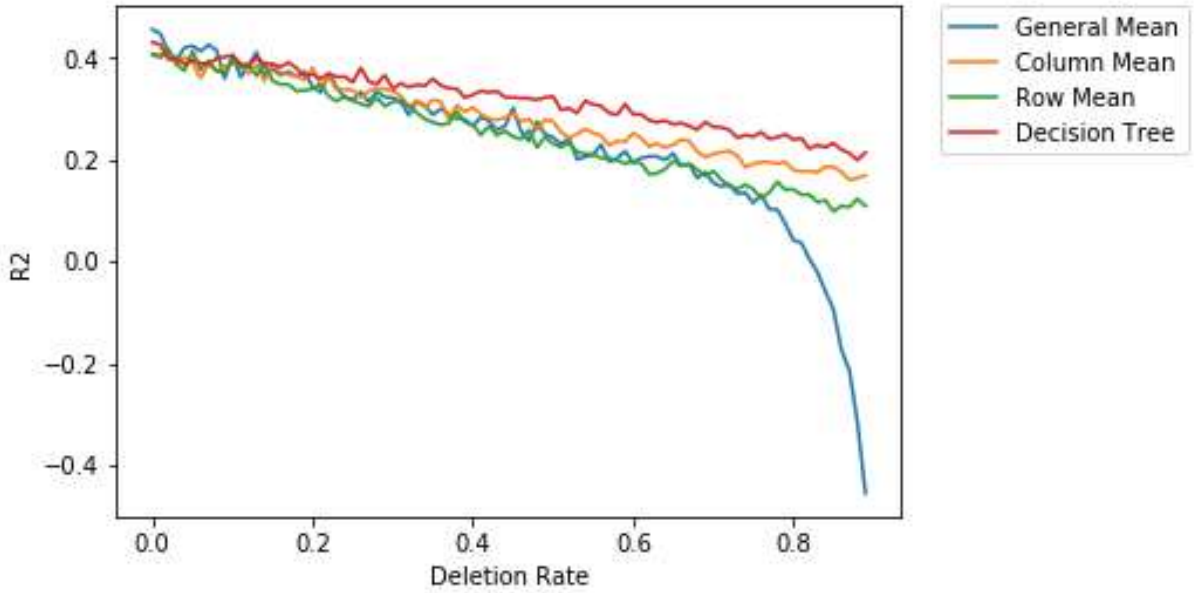


The last evaluation technique used for investigating the performance loss in the multitask ensemble learning method trained with decision based imputed matrix is RMSD percent change. The percentage change in RMSD is shown in the Figure 17. The most noticeable observation is; for same deletion rate, distance between percent change in RMSD values obtained in experiments are fluctuant and can be considered as high. On the mean line, in the second iteration there is a jump from 0,9% of change to 3,13% change. Until, 19th iteration RMSD percent change values roamed around 2,07 to 3,31. When %19 of data imputed, the percentage change rose to 4,21 from 3,29 and showed an increasing pattern. However, in 23rd, 27th, and 30th iterations, a significant drop was observed. RMSD percent change increased to 6 when the iteration count was 40. After 50% of data deleted, change in RMSD was 8,29%. RMSD lost %10 of value at 64th iteration with deletion rate of %65. In 81st and 91st iterations the values were %11,77 and %13,98 respectively. The highest percent change observed in 94th iteration with the change of 14.95%.

4.6. Comparing the Imputation Techniques

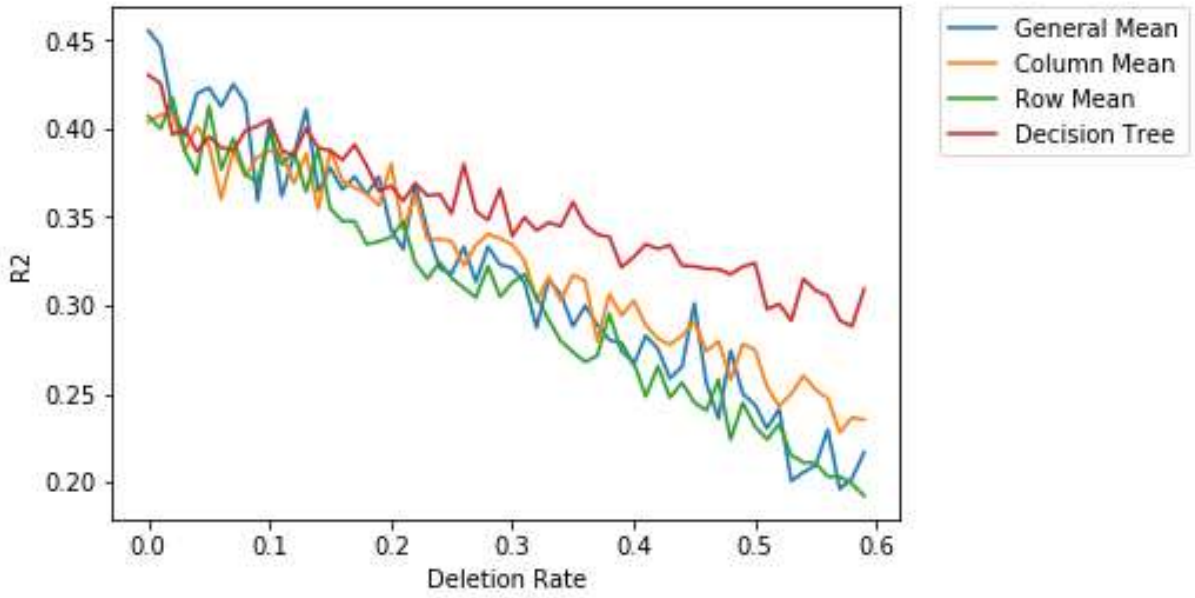
In this section of the thesis, the evaluation scores of the imputation methods were compared investigated and discussed.

Figure 18 R2 Comparison



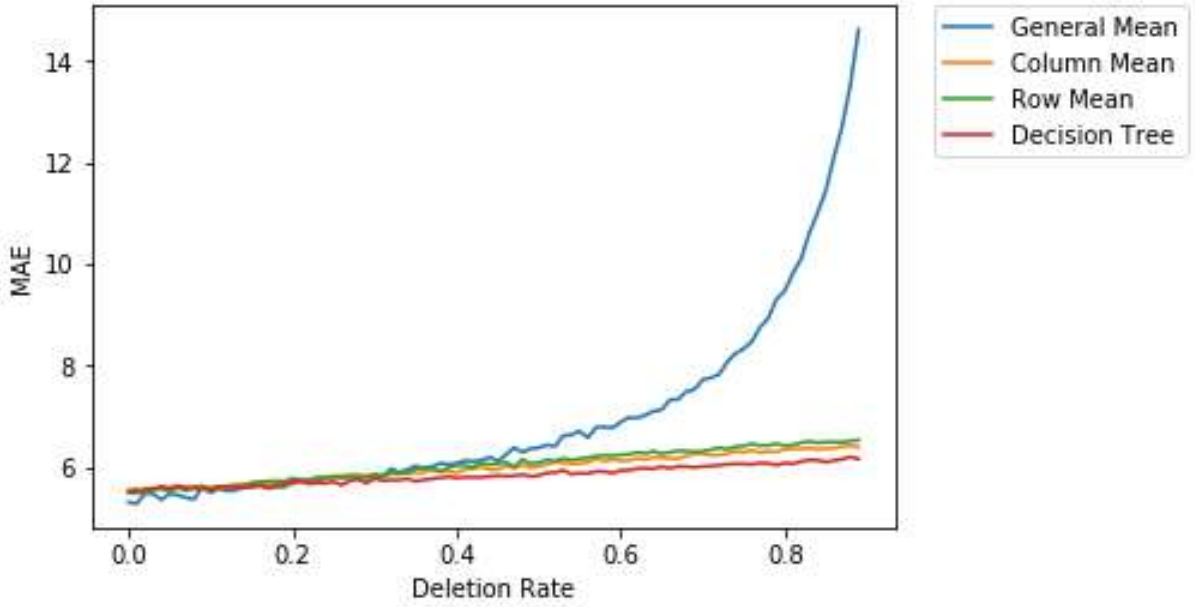
The mean of the R^2 scores of each experiment for different methods are shown in the Figure 18. According to this figure, general mean and row mean methods showed a similar trend until 60% of data is gone missing. After 60% of deletion rate, only the R^2 scores of general mean imputation method change its descending trend from slightly linear to exponential. At the same time, only general mean imputation method went under the y-0 level. In spite of all negative behaviour of the general mean, it is the only one technique that can keep hold of all the column and rows. Therefore, for a reasonable comparison, a new plot was formed and limited the deletion rate to 60%.

Figure 19 R2 Comparison to 60%



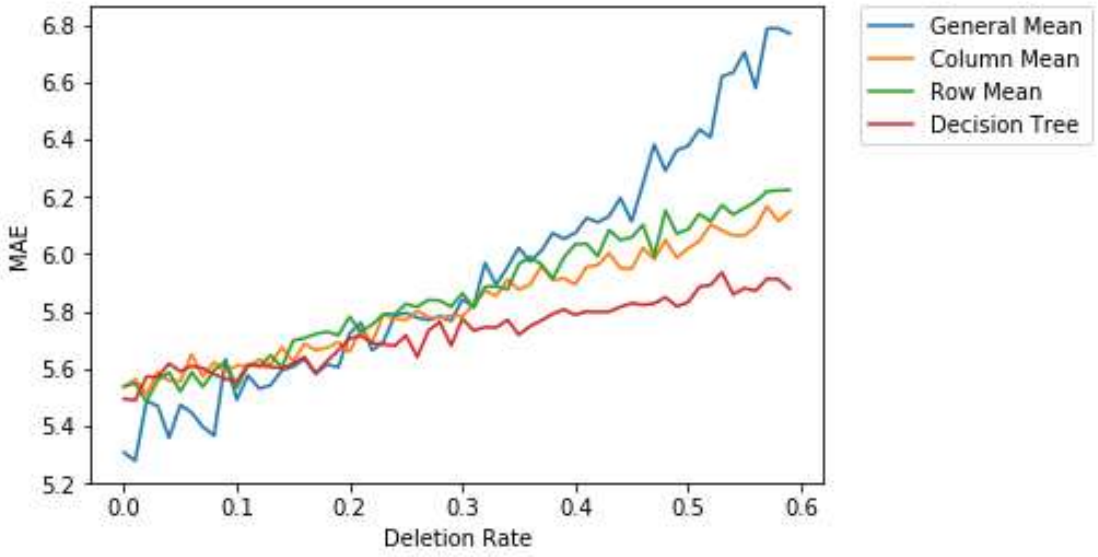
As it can be seen in the Figure 19, when the data imputation is done by segmenting the data into rows or columns, they have a similar performance with general mean in the beginning. At the same time, after a certain amount of data is gone missing, the contribution of these methods to the model increases and surpasses with the general mean imputation approach, even overperforms it with decision tree model. Among the segmentation based imputation methods, decision tree based imputation showed a better performance than row and column mean based techniques. The column, row and general mean based methods showed a similar pattern until the 60% of deletion rate.

Figure 20 MAE Comparison



In figure 20 above, mean absolute values of the imputation methods is shown. Just like in the R^2 comparison, in the beginning the performance loss of imputation methods were similar. After %40 of deletion rate, general mean method started to increase rapidly and opened the gap with the segmentation methods. From the table it can be said that the best results are yielded by decision forest regressor based imputation, followed by column mean imputation. None the less, for a better comparison among the imputation methods, another MAE figure limited to 60% of deletion rate is illustrated below.

Figure 21 MAE Comparison to 60%



According to the Figure 21, the general mean based imputation method was the best performer among all the imputation techniques until the iteration count hit 10. Generally, general mean imputation method's MAE score's tendency of increment was the highest in imputation methods. Around 35 % of deletion rate, general mean imputation became the worst performer imputation technique. The most striking result to emerge from the figure is that the pattern changes in MAE of row and column mean methods, were almost identical till 11th iteration (see Figure 22). Furthermore, until the end of iterations they had shown a similar pattern. It is not surprising to note that decision tree method achieved the best results after 25% of data deletion rate.

Figure 22 MAE Comparison - Column and Mean

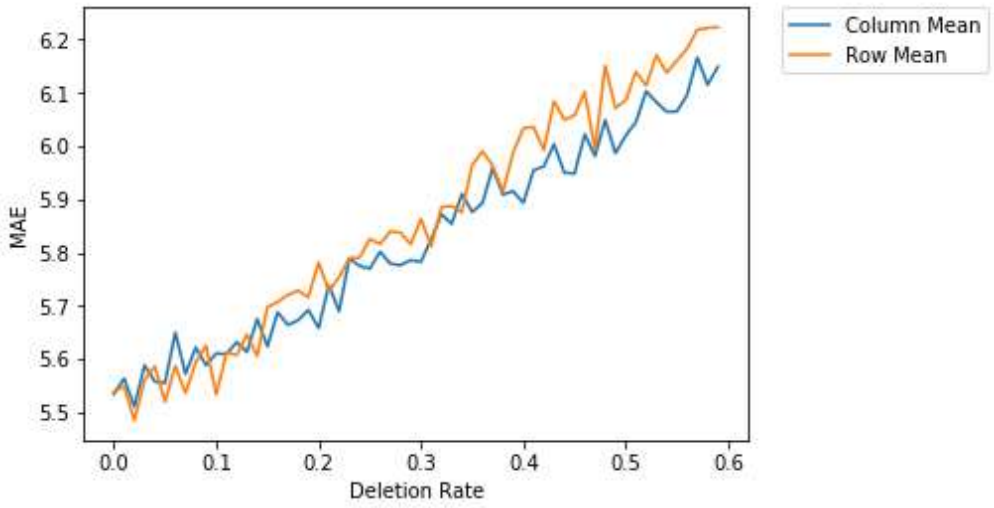
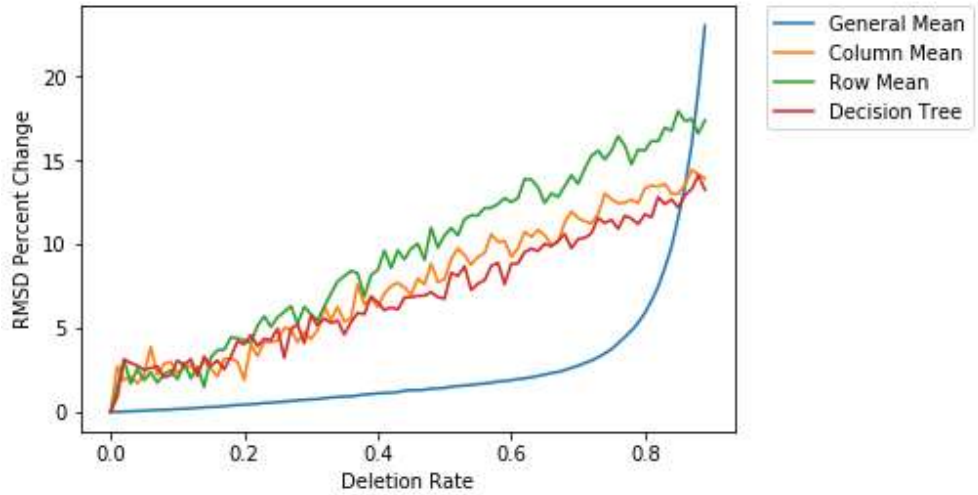
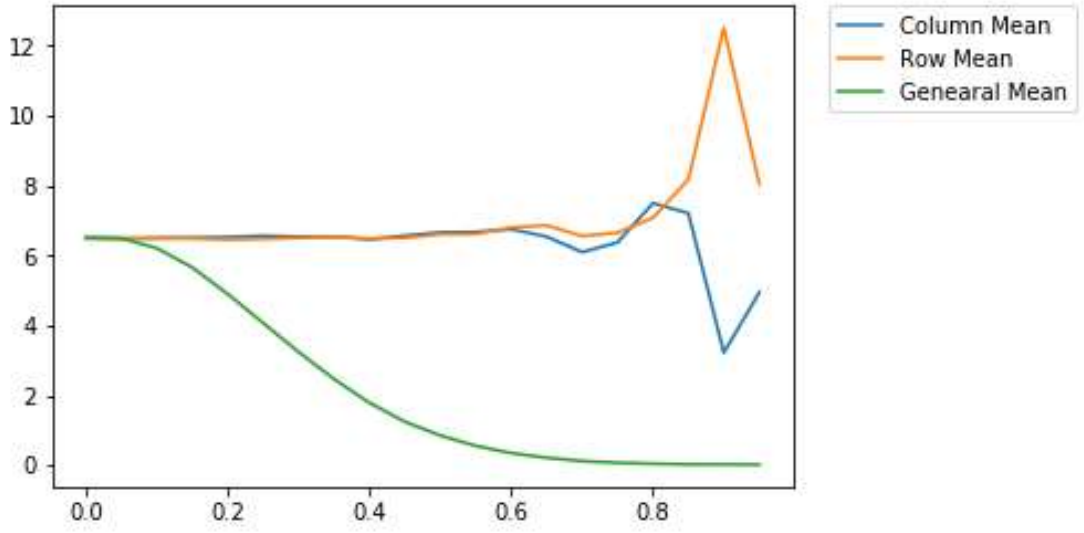


Figure 23 RMSD Percent Change Comparison



The last evaluation method to compare performance differences in imputation methods is root mean square deviation. In Figure 23, instead of RMSD values, the change of percentage in RMSD was calculated and plotted. Surprisingly, the change in RMSD was the smallest when data imputation was conducted with general mean substitution. Other imputation methods showed a similar performance until 40% of data deleted. In the following iterations, the performance difference between column mean and decision tree imputation methods slightly changed where decision tree based imputation technique contributed more than the column mean method, just like in all evaluation methods. Finally, the row mean imputation was the worst not only among segmentation based imputation methods, but also against general mean imputation.

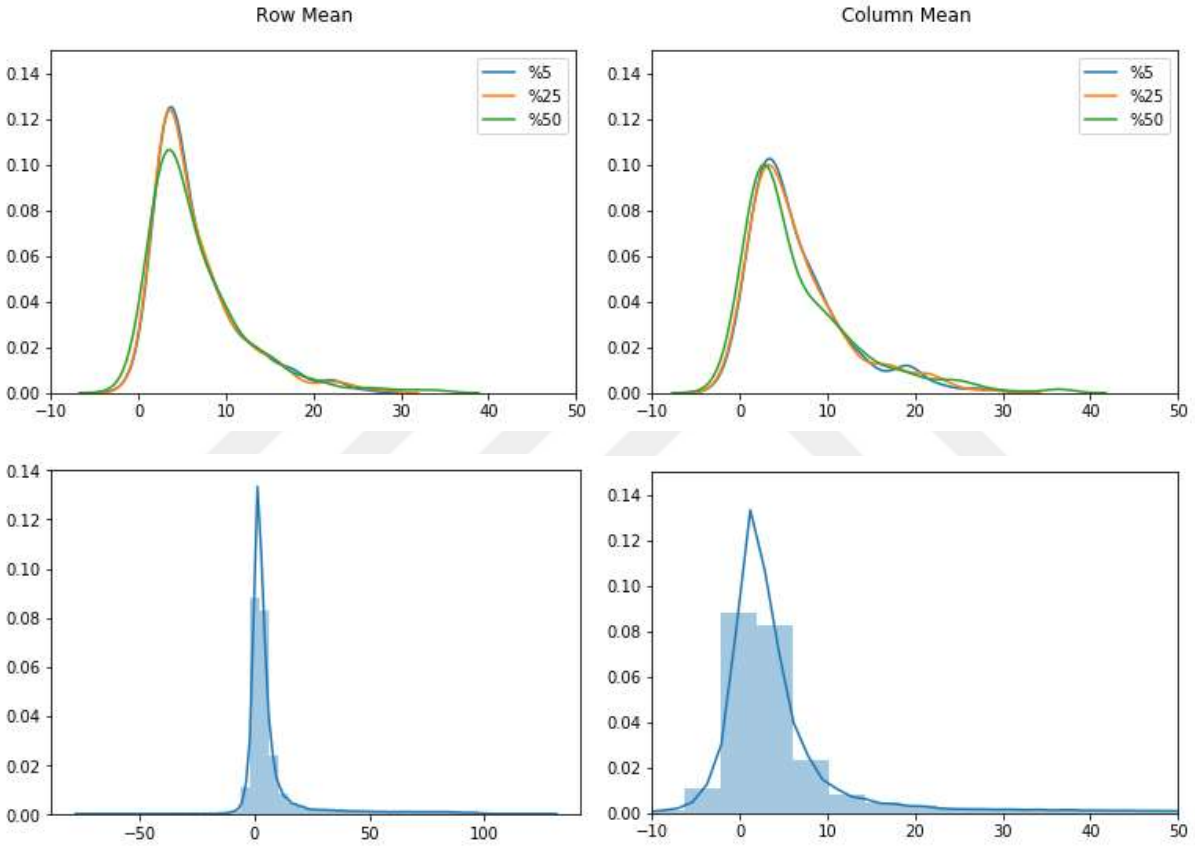
Figure 24 Imputation Value Comparison



The figure above demonstrates the change in imputation values along the iterations. For row and column mean imputation, average of the segment means (for each row and column) were calculated for visualization, since it is not possible to visualize 454 target means and 256 assay means in a single figure. Furthermore, figure should be considered as the visual tool for performance difference of general mean vs segment based mean imputation methods. The general mean imputation values are always decreasing as the missing data ratio is growing. However, after %60 of the data assigned 'NaN', drop in general mean imputation value decelerated. The reason of this deceleration is around 80% of the inhibition values in the full matrix is between 0 and 10. Therefore, it is not a surprising result to see that the general mean imputation values are converging to the median of the full matrix. On the contrary, the average

of row and column mean imputation values are holding on to the central tendency point with negligible differences till 41st iteration. The fraction occurred in row and column mean lines after 80% of deletion rate is a surprising result which can explain the performance gap between two methods. So that, in order to explain the difference between them, the distribution plots are illustrated below in Figure.

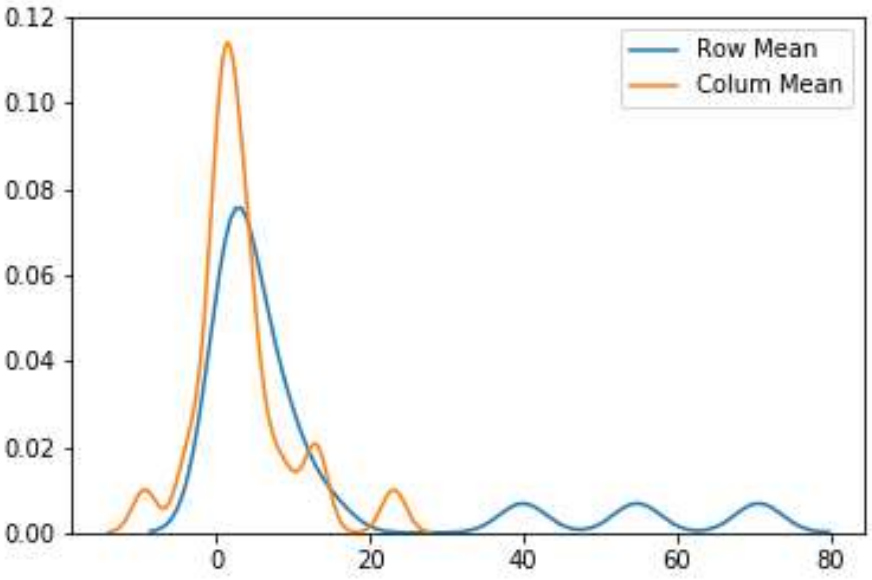
Figure 25 Comparison of row, mean and initial distributions



In the figure, the distribution of imputation values for row (Figure a) and column (Figure b) based method are plotted. Blue line is the distribution at 5% deletion rate, orange is at 25% deletion rate and green line is at 50% of deletion rate. Under the row and column distribution figures, the distribution plot of the activity matrix is illustrated. Figure c represents the full matrix where the figure in the right is limited with the same axis numbers of row and column figures for a better comparison. Row and column figures explain the fact that the average of imputation values of row and columns are similar till 50th iteration since column mean values are spread along negative axis as well, where row mean values are gathered around the median point. The performance difference can be explained as the column mean values range is closer

to the initial data distribution while row mean values are tending to converge to the median. On the other hand, training matrix consists of 454 columns where number of rows are 256. Thus, the data points can directly affect the imputation value distribution.

Figure 26 The distribution comparison of row and column at 90% deletion rate



In the figure above, row and column mean imputation values at 90% of deletion rate are distributed. This figure explains the fraction observed in the Figure where the average of row mean increased rapidly after %80 of deletion rate and as opposite column mean line showed a significant decrease. At 90% of deletion rate row mean almost lost its connection with negative values and population over 20 in terms of inhibition value increased its ratio in the distribution. Besides, column mean values started to merge into the 0-10 level. To be more precise, the behaviour of the methods has changed to total opposite after %80 of data deleted. Even though these results are representatives of multiple runs of the main model, they may be suffered from the randomness of data deletion. So that, deeper analyses should be undertaken in order to reveal the main reason of the difference between imputation methods.

5. CONCLUSION

The present study was designed to determine the effect of imputation methods to the tree-based ensemble multitask learning method trained and tested with published kinase inhibitor set made available by GSK. PKIS data is processed with Extended-connectivity fingerprints in order to make the compounds available for machine learning method. Then, a parameter search for random forest is conducted by using well-known heuristic search algorithm called particle swarm optimization. The parameter search showed that random forest algorithm is robust to change of hyperparameters. Therefore, remaining analyses are conducted with using the default parameters of random forest function in Python since experiments with optimized parameters required huge computational power and considered as time consuming. Thereafter, the training data, that consists of the percent inhibition values, was started filling with 'NaN' data type iteratively to simulate a sparse dataset until 95% of the data being assigned with 'NaN' values. The sparse data is imputed by general mean, column mean, row mean, and decision tree prediction approaches. The imputed data matrices trained tree based ensemble multitask algorithm and tested against the same test set. The models were evaluated by coefficient of determination, mean absolute error and root mean square deviation. In the end, explanatory analyses were conducted among the mean based imputation methods to find out the reason of differences in the performance loss.

This study has found that generally, after 40% of the data is missing, the performance loss of the multitask algorithms become significant. Especially after %60 of missing data is in the training set, performance loss accelerates, no matter which imputation technique is used. Nevertheless, the performance loss observed in the models that were trained with the matrices that was imputed by the general mean of the remaining variables, are more influenced than the other imputation methods. In all evaluation metrics, general mean imputation started increasing linearly until 60% of data is deleted. After 60% of data deletion rate is surpassed, the performance loss coming from general mean imputation started to increase sharply, look alike an exponential increment. The difference between column mean, row mean, and the decision tree imputation based multitask learning performance change was pretty much similar. However, decision tree based technique yielded better results than other imputation methods used in this thesis. Decision tree method is followed by column mean imputation method. In the evaluation metrics, row and column mean imputations yielded similar results, almost identical in some cases, but when the average of the imputation values per experiments were plotted there was a significant difference in those method's tendency. So that an

explanatory investigation had been done. The result of this investigation showed that the difference between the average row and column mean imputation values are coming from change of distribution of the remained values which can be explained by non-equal number of targets and assays in the training matrix and the randomness of the data deletion.

This research has several practical applications. Firstly, it points to the robustness of random forest algorithm to its parameter change. Secondly, it showed that the mean based imputation methods can be replaced by the predictive approach to the missing data issue. Finally, it pointed out to correlation between the missing data ratio and the performance of the machine learning algorithm.

The project was limited in several ways. First, in terms of applying regression based imputation techniques, only decision tree algorithm is used. Besides, since the aim of the project is to investigate the performance differences in imputation methods and there is only one regression imputation method was used, decision tree models were not tested to reveal the overall performance along the iterations. Second, the imputation methods were limited with regression and mean substitution. Last source of weakness in this study which could have affected the measurements of the effect of imputation was this study only focused and simulated one type of missing data which is called as missing completely at random.

This research has thrown up many questions in need of further investigation. Even though, random forest showed its robustness to parameter change, same parameter search process can be done by using genetic algorithm because genetic algorithm is more suitable for searching in discrete functions. It would be interesting to assess the effects of the other missing data types such as missing at random (MAR) and missing not at random (MNAR). The framework proposed in this thesis is needed to apply another chemical data set that contains drug activity values and in a shape for multitask learning. As well as using new data sets, it would be interesting to see the performance loss on the other multitask learning methods and compare them with tree based ensemble algorithm since algorithm like Deep Neural Networks may yield better results. Evaluation of decision tree imputation is needed to be undertaken, thereafter more powerful regression or predictive methods should be applied and compared with decision trees such as artificial neural networks. Eventually, future research should therefore concentrate on the investigation of different imputation methods, such as hot deck or cold deck imputation, and multiple imputation as well as random data imputation.

6. REFERENCES

Abdo, A., Chen, B., Mueller, C., Salim, N., & Willett, P. (2010). Ligand-Based Virtual Screening Using Bayesian Networks, 1012–1020.

Agarwal, S., Dugar, D., & Sengupta, S. (2010). Ranking chemical structures for drug discovery: A new machine learning approach. *Journal of Chemical Information and Modeling*, *50*(5), 716–731. https://doi.org/10.1021/ci9003865

Basak, S. C., Grunwald, G. D., Gute, B. D., Balasubramanian, K., & Opitz, D. (2000). Use of Statistical and Neural Net Approaches in Predicting Toxicity of Chemicals. *Journal of Chemical Information and Computer Sciences*, *40*(4), 885–890. https://doi.org/10.1021/ci9901136

Breiman, L. (1996). Bagging predictors. *Machine Learning*, *24*(2), 123–140. https://doi.org/10.1007/BF00058655

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1023/A:1010933404324

Briem, H., & Gunther, J. (2005). Classifying "kinase inhibitor-likeness" by using machine-learning methods. *ChemBioChem*, *6*(3), 558–566. https://doi.org/10.1002/cbic.200400109

Byvatov, E., Fechner, U., Sadowski, J., & Schneider, G. (2003). Comparison of Support Vector Machine and Artificial Neural Network Systems for Drug/Nondrug Classification. *Journal of Chemical Information and Computer Sciences*, *43*(6), 1882–1889. https://doi.org/10.1021/ci0341161

Dearden, J. C. (2017). *Advances in QSAR Modeling* (Vol. 24). https://doi.org/10.1007/978-3-319-56850-8

Devillers, J., & Balaban, A. T. (1999). *Topological indices and related descriptors in QSAR and QSPR*. Retrieved from https://books.google.co.uk/books?hl=tr&lr=&id=2KH25rClXpAC&oi=fnd&pg=PR7&dq=molecular+descriptors&ots=D00MpI6vL_&sig=b4GA-sHFhdbEv9Fmkn5f7BKML4Y&redir_esc=y#v=onepage&q=molecular descriptors&f=false

Dranchak, P., MacArthur, R., Guha, R., Zuercher, W. J., Drewry, D. H., Auld, D. S., &

Inglese, J. (2013). Profile of the GSK Published Protein Kinase Inhibitor Set Across ATP-Dependent and-Independent Luciferases: Implications for Reporter-Gene Assays. *PLoS ONE*, *8*(3), 1–9. https://doi.org/10.1371/journal.pone.0057888

Folguera, L., Zupan, J., Cicerone, D., & Magallanes, J. F. (2015). Self-organizing maps for imputation of missing data in incomplete data matrices. *Chemometrics and Intelligent Laboratory Systems*, *143*, 146–151. https://doi.org/10.1016/j.chemolab.2015.03.002

Hall, L. H., & Kier, L. B. (1995). Electrotopological State Indices for Atom Types: A Novel Combination of Electronic, Topological, and Valence State Information. *Journal of Chemical Information and Computer Sciences*, *35*(6), 1039–1045. https://doi.org/10.1021/ci00028a014

Hall, L. H., & Kier, L. B. (2007). The Molecular Connectivity Chi Indexes and Kappa Shape Indexes in Structure-Property Modeling (pp. 367–422). Wiley-Blackwell. https://doi.org/10.1002/9780470125793.ch9

Hert, J., Willett, P., Wilton, D. J., Acklin, P., Azzaoui, K., Jacoby, E., & Schuffenhauer, A. (2004). Comparison of Fingerprint-Based Methods for Virtual Screening Using Multiple Bioactive Reference Structures. *Journal of Chemical Information and Computer Sciences*, *44*(3), 1177–1185. https://doi.org/10.1021/ci034231b

Honório, K. M., & da Silva, A. B. F. (2005). A study on the influence of molecular properties in the psychoactivity of cannabinoid compounds. *Journal of Molecular Modeling*, *11*(3), 200–209. https://doi.org/10.1007/s00894-005-0243-z

Itskowitz, P., & Tropsha, A. (2005). kappa Nearest neighbors QSAR modeling as a variational problem: theory and applications. *Journal of Chemical Information and Modeling*, *45*(3), 777–85. https://doi.org/10.1021/ci049628+

Jerez, J. M., Molina, I., García-Laencina, P. J., Alba, E., Ribelles, N., Martín, M., & Franco, L. (2010). Missing data imputation using statistical and machine learning methods in a real breast cancer problem. *Artificial Intelligence in Medicine*, *50*(2), 105–115. https://doi.org/10.1016/j.artmed.2010.05.002

Jorissen, R. N., & Gilson, M. K. (2005). Virtual screening of molecular databases using a support vector machine. *Journal of Chemical Information and Modeling*, *45*(3), 549–561. https://doi.org/10.1021/ci049641u

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Neural Networks, 1995. Proceedings., IEEE International Conference On*, *4*, 1942–1948 vol.4. https://doi.org/10.1109/ICNN.1995.488968

Kier, L. B., & Hall, L. H. (1990). An Electrotopological-State Index for Atoms in Molecules. *Pharmaceutical Research*, *07*(8), 801–807. https://doi.org/10.1023/A:1015952613760

Klekota, J., & Roth, F. P. (2008). Chemical substructures that enrich for biological activity. *Bioinformatics*, *24*(21), 2518–2525. https://doi.org/10.1093/bioinformatics/btn479

Knapp, S., Arruda, P., Blagg, J., Burley, S., Drewry, D. H., Edwards, A., … Zuercher, W. J. (2013). A public-private partnership to unlock the untargeted kinome. *Nature Chemical Biology*, *9*(1), 3–7. https://doi.org/10.1038/nchembio.1113

Koutsoukas, A., Lowe, R., Kalantarmotamedi, Y., Mussa, H. Y., Klaffke, W., Mitchell, J. B. O., … Bender, A. (2013). In silico target predictions: Defining a benchmarking data set and comparison of performance of the multiclass Naïve Bayes and Parzen-Rosenblatt Window. *Journal of Chemical Information and Modeling*, *53*(8), 1957–1966. https://doi.org/10.1021/ci300435j

Lakshminarayan, K., Harp, S. A., Goldman, R., & Samad, T. (1996). Imputation of Missing Data Using Machine Learning Techniques. *KDD-96 Proceedings*, 140–145. https://doi.org/10.1016/j.inffus.2012.08.007

Lavecchia, A. (2015). Machine-learning approaches in drug discovery: Methods and applications. *Drug Discovery Today*, *20*(3), 318–331. https://doi.org/10.1016/j.drudis.2014.10.012

Leach, A. R., & Gillet, V. J. (2007). *An introduction to chemoinformatics*. *An Introduction To Chemoinformatics*. https://doi.org/10.1007/978-1-4020-6291-9

Leung, K. M. (2007). Naive bayesian classifier. *Polytechnic University Department of Computer Science/Finance and Risk Engineering*.

Livingstone, D. J. (2000). The Characterization of Chemical Structures Using Molecular Properties. A Survey. *Journal of Chemical Information and Computer Sciences*, *40*(2), 195–209. https://doi.org/10.1021/ci990162i

Lu, N., Zhou, J., He, Y., & Liu, Y. (2009). Particle Swarm Optimization for Parameter Optimization of Support Vector Machine Model. *2009 Second International Conference*

on Intelligent Computation Technology and Automation, (3), 283–286. https://doi.org/10.1109/ICICTA.2009.76

Ma, J., Sheridan, R. P., Liaw, A., Dahl, G. E., & Svetnik, V. (2015). Deep Neural Nets as a Method for Quantitative Structure-Activity Relationships.

Mannhold, R., & Van De Waterbeemd, H. (2001). Substructure and whole molecule approaches for calculating log P. *Journal of Computer-Aided Molecular Design*, *15*(4), 337–354. https://doi.org/10.1023/A:1011107422318

Mayr, A., Klambauer, G., Unterthiner, T., & Hochreiter, S. (2016). DeepTox: Toxicity Prediction using Deep Learning. *Frontiers in Environmental Science*, *3*(February). https://doi.org/10.3389/fenvs.2015.00080

Nordbotten, S. (1996). Neural Network Imputation Applied to the Norwegian 1990 Population Census Data. *Journal of Official Statistics*, *12*(4), 385–401.

Rahman, M. G., & Islam, M. Z. (2013). Missing value imputation using decision trees and decision forests by splitting and merging records: Two novel techniques. *Knowledge-Based Systems*, *53*, 51–65. https://doi.org/10.1016/j.knosys.2013.08.023

Ramsundar, B., Liu, B., Wu, Z., Verras, A., Tudor, M., Sheridan, R. P., & Pande, V. (2017). Is Multitask Deep Learning Practical for Pharma? *Journal of Chemical Information and Modeling*, *57*(8), 2068–2076. https://doi.org/10.1021/acs.jcim.7b00146

Randi, M. (2001). The connectivity index 25 years after. *Journal of Molecular Graphics and Modelling*, *20*(1), 19–35. https://doi.org/10.1016/S1093-3263(01)00098-5

Rogers, D., & Hahn, M. (2010). Extended-Connectivity Fingerprints. *Journal of Chemical Information and Modeling*, *50*(5), 742–754. https://doi.org/10.1021/ci100050t

Schneider, N., Jäckels, C., Andres, C., & Hutter, M. C. (2008). Gradual in silico filtering for druglike substances. *Journal of Chemical Information and Modeling*, *48*(3), 613–628. https://doi.org/10.1021/ci700351y

Shi, Y., & Eberhart, R. C. (1999). Empirical study of particle swarm optimization. *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress On*, *3*, 1–1950 Vol. 3. https://doi.org/10.1109/CEC.1999.785511

Silva-Ramírez, E. L., Pino-Mejías, R., López-Coello, M., & Cubiles-de-la-Vega, M. D.

(2011). Missing value imputation on missing completely at random data using multilayer perceptrons. *Neural Networks*, *24*(1), 121–129. https://doi.org/10.1016/j.neunet.2010.09.008

Simm, J., Magrans, I., Abril, D. E., & Sugiyama, M. (2014). Tree-Based Ensemble Multi-Task Learning Method for Classification and Regression. *IEICE Transactions on Information and Systems*, (6), 1677–1681.

Smith, D. H., Carhart, R. E., & Venkataraghavan, R. (1985). Atom Pairs as Molecular Features in Structure-Activity Studies: Definition and Applications. *Journal of Chemical Information and Computer Sciences*, *25*(2), 64–73. https://doi.org/10.1021/ci00046a002

So, S.-S., & Karplus, M. (1997). Three-dimensional quantitative structure-activity relationships from molecular similarity matrices and genetic neural networks. 1. Method and validations. *Journal of Medicinal Chemistry*, *40*(26), 4347–59. https://doi.org/10.1021/jm970487v

Sutariya, V. B., Groshev, A., & Pathak, Y. V. (2013). Artificial Neural Networks in Pharmaceutical Research, Drug Delivery and Pharmacy Curriculum. *2013 29th Southern Biomedical Engineering Conference*, *4749*, 91–92. https://doi.org/10.1109/SBEC.2013.54

Tin Kam Ho. (1995). Random decision forests. *Proceedings of 3rd International Conference on Document Analysis and Recognition*, *1*, 278–282. https://doi.org/10.1109/ICDAR.1995.598994

Xu, Y., Ma, J., Liaw, A., Sheridan, R. P., & Svetnik, V. (2017). Demystifying Multitask Deep Neural Networks for Quantitative Structure-Activity Relationships. *Journal of Chemical Information and Modeling*, *57*(10), 2490–2504. https://doi.org/10.1021/acs.jcim.7b00087