

Final Report: Prediction with Incomplete Data, with application to Automated Valuation Models

**Jaesung Park**

[**Shyjp1@nottingham.edu.cn**](mailto:Shyjp1@nottingham.edu.cn)

**Supervised by Anthony Graham Bellotti**

I hereby declare that this dissertation is all my own work, except as indicated in the text.

Date 26 / April / 2021

School of Computer Science University of Nottingham Ningbo China

**Abstract**

Dealing with incomplete observation is one of the challenging tasks in machine learning. Incomplete observation can lead to erroneous predictions, and the consequences of erroneous predictions can sometimes be disastrous. It usually arises when the data is omitted in the collection process, when it does not meet the quality control criteria, or does not exist in the first place. This dissertation starts from the one question: “What if the user has a short amount of time to give incomplete observations, but still wants valuable results?” There are still a lot of studies that handle missing observations in training sets alone. However, background research shows that there are not many studies that cover how to deal with missing observations on the testing set when the training set has completely filled.

This dissertation compares among four methods -Deletion, Mean Imputation, Regression Imputation, and custom K nearest neighbor Imputation- that deal with missing observations. Also, there are two housing property datasets to see how methods react on different structures of a dataset. The goal of this project was to provide an in-depth comparison among different methods that handle missing observation. The dissertation premises that the training set has no missing observation.

**Contents**

**1 Introduction**

1.1 Background ..................................................................................................................... 4

1.2 Motivation ....................................................................................................................... 5

1.3 Aims and Objectives ...................................................................................................... 5

1.4 Final Report Outline ...................................................................................................... 6

2 Related Research

2.1 Related Research ............................................................................................................. 7

3 Machine Learning Techniques

3.1 K-fold Cross Validation ................................................................................................. 8

3.2 Linear Regression ........................................................................................................... 8

3.3 Ridge Regression ............................................................................................................ 9

3.4 Imputation ....................................................................................................................... 9

3.4.1 Mean Imputation ......................................................................................... 10

3.4.2 Regression Imputation ............................................................................... 10

4 Implementation

4.1 Data Processing .............................................................................................................11

4.1.1 Pre-Processing ............................................................................................. 11

4.1.2 Data Preparation ......................................................................................... 12

4.1.3 Coefficients .................................................................................................. 13

4.1.4 Learning Curve ........................................................................................... 14

4.2 AVM Results ..................................................................................................................15

4.2.1 K-fold Validation ........................................................................................ 15

4.2.2 Comparison.................................................................................................. 15

4.2.2.1 Deletion ..................................................................................................... 15

4.2.2.2 Mean Imputation ..................................................................................... 17

4.2.2.3 Regression Imputation ............................................................................ 18

5 Progress

5.1 Project Management .....................................................................................................20

6 Reflection ..........................................................................................................................................22

References ............................................................................................................................................23

Appendices

Appendix A .........................................................................................................................................24

Appendix B ..........................................................................................................................................27

Appendix C .........................................................................................................................................28

**List of Figures**

**List of Tables**

**List of Abbreviations**

**AVM**  Automated Valuation Model

**RMSE** Root Mean Square Error

**CV** Cross Validation

**LR** Linear Regression

**RR** Ridge Regression

**GDA** Gradient Descent Algorithm

**MAE** Mean Absolute Error

**MSE** Mean Squared Error

**Chapter 1**

**Introduction**

According to the definition from Oxford Languages, “Valuation is an estimation of something’s worth, especially one carried out by a professional appraiser” [0]. Ever since people have begun selling their items, they needed a professional valuation to value at a fair price. Automated Valuation Model, in the real estate market, is a software-based pricing model that produces estimated value for a property by a machine learning model. It uses historical databases that contain property information of the surrounding areas.

Before the Automated Valuation Model was released, valuations were entirely determined by human appraisers. In the late 1990s, AVM was only used by institutional investors to determine risk when purchasing loans [1]. Along with the development of technology and machine learning, AVM has been advanced rapidly and it has been becoming more popular. Often Appraisals review 3-5 comparable properties, similar to requested property in terms of location, physical appearance, and overall condition. There is a limitation to find truly comparable properties. On the other hand, AVM allows to estimate value based on every single element of properties, and it is not limited to neighborhood. Moreover, AVM has additional strength of less time consuming, smaller risk of human error, and lower service price. Nowadays, Multiple AVM providers offer their AVM to users, including commercial platforms like CoreLogic, Freddie Mac, and Equifax; as well as free platforms like Zillow and Trulia [2].

This chapter describes the background and motivation of this project, moreover, explains the aims and objectives.

**1.1 Background**

In recent research shows that more than 25% of rural appraisals exceed the contract price by 5%, because of appraisals’ bias [3]. On the other hand, AVM is based on a large amount of data as inputs without human interference, thereby eliminating the risk of bias and subjectivity. As even 1% in real estate property price is gigantic, one of the strengths of AVM that eliminating human-bias makes AVM more attractive.

Though AVMs also have limitation, they are only as accurate as the data behind them, in other words, they possibly are incorrect or outdated. Recent studies show that as the model requires non-standard data, AVMs are not suitable for unique property valuation [4][5]. The research shows that RMSE differences of the result with the basic data versus result with the enhanced data were from 19% to 35% [6]. Meaning, they require an accurate, fully-filled, and well-structured dataset. Also, based on the structure of the dataset and type of data, engineers encounter different challenges, since AVMs are highly dependent on the dataset and model.

To overcome these limitations of AVMs, countless studies and discussions are actively ongoing, in terms of machine learning and database: data cleaning on missing and noisy data, data transformation, adding more data, or finding better adequate ML algorithms. Thanks to these efforts, AVMs and, in October 2018, representatives from the AVM sector in Leaders Forum in the Netherlands concluded that the AVM’s usefulness will expand in the future [7]. 이들을 포함해서 미래에 확장될거라 믿고있다.

**1.2 Motivation**

However, in operational circumstances, there are situations where: a customer has a short amount of time to get all property information and still needs to get an estimated sale price or a customer is short on budget to hire an appraiser to get the property price. Moreover, the situation also occurs when data will be missing for a new property. This dissertation is to reflect and simulate this operational problem. Searching through handling missing data on AVM that are on the field now, it can be concluded that there are far fewer studies in progress that handling missing data in testing set compare to handling missing in the training set. Plus, AVM providers do not share their automated valuation models, implementations, and data with the public for their reasons. Some AVM providers’ websites manually do not let the customer leave blank on their inputs and that might cause less accurate valuation if a customer has no clue on certain specific information and input their guess. It is worth investigating AVM prediction with incomplete data on the testing set.

**1.3 Aims and Objectives**

The main aim of this project is to find the best algorithmic approach when testing with an incomplete test set. To build a testing environment before testing, certain steps have to be accomplished: understanding dataset structure and pre-processing dataset. Those steps are also included in Chapter 4. The key objectives of this dissertation are:

1. Preprocessing the dataset to fit the model
2. Implement a cross-validation environment to obtain stable results
3. Test to find an adequate machine learning algorithm that works with the full size of the dataset.
4. Investigate imputation and non-imputation to fill up missing observations.
5. Compare different algorithmic approaches.
6. Evaluate the performance of the model with the chosen machine learning algorithm and imputation method.

**1.4 Dissertation Outline**

This chapter has summarized the background of AVM, provided a general overview of the automated valuation model, and the aim and objectives of this dissertation. Chapter 2 shows related research when handling missing values and Chapter 3 explains different machine learning techniques and algorithms that have been handled in this research. Chapter 4 presents the results of the implementation. Chapter 5 shows the project management. In closing, Chapter 6 expresses the reflections and considerations for future work.

**Chapter 2**

**Related Research**

As explained in introduction, there is a limitation on stating related AVM companies’ research or related studies, because when researchers study about missing observation in machine learning, it generally meant for missing data on training set.

**2.1 Related Research**

According to UCLA Statistical Consulting seminar, it introduced multiple imputation techniques. The goal of handling missing data they proposed were: 1. Minimize bias 2. Maximize use of available information 3. Obtain appropriate estimates of uncertainty. Also, there are three different types in missing values which are Missing completely at random (MCAR), Missing at random (MAR), and Missing not at random (MNAR). Different types of missing data require different treatments.

**텍스트, 시계이(가) 표시된 사진

자동 생성된 설명**

Figure 2.1. Graphical representation of MCAR, MAR, and MNAR. X represents

completely observed, Y partly missing, Z component of the causes of missingness unrelated to X and Y, R represents the missingness.

According to seminar, choice of distribution, auxiliary variables, and number of imputations can affect the quality of the imputation. It also mentioned multiple imputation is always superior to any of the single imputation methods because a single imputed value is never used, and the variance estimates reflect the appropriate amount of uncertainty surrounding parameter estimates.

**Chapter 3**

**Machine Learning Techniques**

* 1. **K-fold Cross-Validation**

Cross-Validation is a resampling procedure used to evaluate models on a limited data sample. It is powerful because of several reasons: First, it allows to use of all of the data. Let’s say, we have little data, splitting it into training and test set might leave us with a tiny test set. Then, the prediction cannot lead to any real conclusion. By splitting into k numbers of the fold, it allows predicting with every observation in the dataset. Secondly, it allows us to get more metrics. If we run a single evaluation, we can get only one prediction, meaning we cannot ensure that the result is reliable, because it could be predicted by chance or biased. With K-fold CV, it produces k numbers of predictions on a single run, and it allows to see whether the predictions are consistent or inconsistent. These benefits are useful, especially when proceeding with parameter tuning. By doing cross-validation, parameter tuning can be done using a single set.

K-fold cross-validation has a parameter called k that refers to the number of groups that a given data is to be split into. For example, if variable k is 5 and the sample is consisting of 1,000 observations with 10 features, each fold set has 200 observations with 10 features. Each fold’s test set does not get redundant with any other test set’s data, so it helps lower the bias of the data.

* 1. **Linear Regression**

Linear regression is the most widely used algorithm because of its simplicity: easy to implement and easy to interpret the output coefficients, but still powerful [8]. Linear Regression is a supervised machine learning algorithm where the output is continuous and has a constant slope. It is used to predict continuous values rather than classification. The multivariable linear regression equation is as follows:

The variables x represent the pieces of information with m predictor and variable w represents the weight. These are the residual terms of the model.

Minimizing differences between predicted values and the actual values is the key step to optimize the linear regression model: minimizing the cost function. With linear regression equation: , W and x should be vectors and y is an outcome variable. The cost function is as follows:

Gradient Descent Algorithm is one of the algorithms to minimize the cost function. GDA starts from a random initial value, calculates gradient, and gives changes to W and b to minimize the gradient. It repeats the process on the point where gradient mostly decreased and finally finds the minimum cost function point.

The cost function is a little changed to give a clear calculation. If we substitute cost function to GDA,

If we simplify,

* 1. **Ridge Regression**

Ridge regression is a type of regularized linear regression that is used to data that suffers from multicollinearity or overfitting. RR adds a squared magnitude of coefficient as penalty term to the loss function. Following is the equation of Ridge Regression:

The equation can be interpreted as:

Ridge regression is a sum of residual sum of squares + penalty term. If lambda, ridge function alpha, is zero, the equation is the default OLS, but if it is greater than zero, it adds a constraint to the coefficient. This constraint leads to a minimized coefficient that tends to go to zero the greater the value of the lambda. Decreasing the coefficient stops the variance and continues the error value. Thus, Ridge regression reduces the complexity of the model, but does not reduce the number of variables, but reduces its effect. According to Scikit-Learn, ridge regression is strong when the data is less than a thousand samples or when we have more parameters than samples.

Additionally, the following is the equation of Lasso regression:

By giving the absolute sum of the coefficients on the cost function, in contrast to Ridge regression’s squared of the cost function, it leads to making coefficients to absolute zero when Ridge never sets the value of coefficient to absolute zero. In other words, Lasso is not punishing high values of the coefficients but as setting them to zero if it is irrelevant. Therefore, Lasso can lead to fewer features in the model.

In this dissertation, Lasso regression hasn’t been used for the following reasons: Firstly, Lasso works well if there are a small number of significant features and the rest has a small influence on the response. Since our two datasets had many large parameters of about the same value or there are too few features to be removed from, Ridge regression was more suitable to the model.

* 1. **Imputation**

The imputation method is a widely used method when dealing with missing data. If variables are missing from the data entirely (with no bias), there is an option to omit those, but if there is some kind of potential in variables like bias, imputation is a good option to proceed. Different from eliminating the data, it leads to maintain the sample size and all variables. The imputation method is simply replacing the missing value with a substituted value.

In this dissertation, the missing observations are only located in the test set. To clarify, the imputation needs to be mapped from the train observations. There are reasons why should not be mapped from the test set: First, in a real-life situation, the model’s input should be able to process one row. Taking mean or median of one row is the same value which does not do anything. Secondly, the model should create be independent of the test set. By using the test set to calculate observation, it leads the model to depend on the test set size. For instance, the number would be different when there are 10 observations and when there are 100 observations. This, mapping from a training set, logic is applied to all imputation methods in this dissertation.

* + 1. **Mean Imputation**

Mean Imputation is one of the simple imputation methods that any missing observations in each feature are replaced with the mean of that specific feature in the training set. For this research, if there is missing observation in one feature, it is assumed that there is no data in that specific feature. So, for example, to get mean imputation, get a mean from a specific feature in the training set and impute to the specific feature in the testing set.

* + 1. **Regression Imputation**

Regression Imputation is an advanced imputation method that replacing missing values with a predicted result based on a regression line. Each missing feature has its model to predict its values. Let’s say, there are A and B features are missing in the test set, firstly, build a model with known features in the training set, and predicted value from the model of A get imputed to feature A in the test set. Same process for feature B.

Regression imputation lets the dataset preserve relationships among variables while eliminating variability around predicted values.

* + 1. **Custom K-Nearest Neighbor Imputation**

The original K-Nearest neighbor imputation algorithms substitute the mean of K numbers of neighbors that are closest to the missing value. It uses a different type of distance metrics to calculate the distance between the feature that contains the missing value to the neighbor, in this case, the model used the Euclidean distance metric. It calculates the shortest distance between two points and the following is the formula.

Where D represents the distance, p and q represent two different points, and n represents the number of features. To sum up, it calculates the distance from the known features on the observation that contains an unknown variable to every single observation that is in the training set without the feature that is unknown on the test set. After calculating distances, the normal KNN imputation method would get the mean from K numbers of neighbors’ feature values and substitute.

On the other hand, the custom KNN reflect the part of Monte Carlo method’s concept that relies on repeated sampling to obtain the result and it doesn’t get mean directly from neighbors, rather use all K numbers of neighbor to regression and get the mean of predicted results. The custom KNN method, not only allows us to get the K numbers of the closest neighbor but also, presents more accurate and stable predictions.

**Chapter 4**

**Implementation**

This chapter specifies the implementation steps of the dataset and models. This chapter also explains the data structure (Section 4.1) and how the training set got pre-processed.

**4.1 Dataset**

There are two different housing datasets from the US, one from Ames’ county and the other from King’s county. To be easier, Ames’ county dataset will be called dataset A and King’s county dataset as dataset B.

Both dataset’s features represent different features of housing property. Dataset A consists of both numerical and categorical values and it has a total of 2930 observations with 78 features and dataset B consists of only numerical values and it has total of 21,613 observations with 21 features. The observations represent the number of housing properties. For additional information for features on both datasets, refer to Appendix A and B.

**4.1.1 Pre-Processing**

Unlike dataset B, dataset A had numerous categorical variables, which cannot be entered into the regression equation. One-hot encoding technique is to add a new feature according to the type of feature value and display 1 in the column corresponding to the unique value and 0 in the remaining columns. By using the *get\_dummies* function from Pandas, each categorical column is transformed into three numerical columns. Then, dropping one column on each transformed set of columns increases the accuracy by adding ambiguity or noise [9]. After changed all categorical values to numeric values, the sample size of dataset A changed from (2930,83) to (2930, 160).

Secondly, there were few un-useful rows of observation on dataset A, as only 2 out of 83 features are available. Thus, those 145 numbers of observations are deleted. After deletion, the sample size got reduced from (2930,160) to (2785,160).

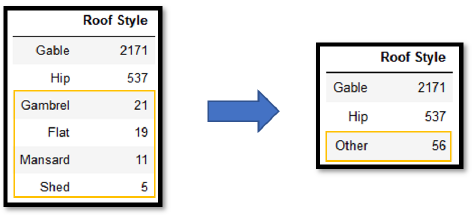


Figure 4.1. Example of creating a new level, ‘other’

Lastly, by creating a new level and naming ‘other’ as shown in Figure 4.1, NAs and levels with few variables are combined into one new level. By approaching this technique, each feature can only contain major levels that have large numbers of a variable while no need to delete observation that has NA.

On the other hand, dataset B was more of the complete dataset compared to dataset A. There was not any missing value or categorical variables. However, both models were suffering from multicollinearity, in which one or more variables are linear or very nearly linear on each other. Consequently, these features had to be removed or refactored. These steps are discussed in 4.1.2. Coefficient section.

**4.1.2 Coefficients**

On both datasets, models were suffered from multicollinearity. In dataset A, there were 4 columns with an abnormally high coefficient which was 9.6e+12 when other coefficients were around ±10,000. Four columns were ‘Total Bsmt SF’, ‘Bsmt Unf SF’, ‘BsmtFin SF 1’, and ‘BsmtFin SF 2’. As we can assume in the similar feature names, four features seem related. Table 4.1. shows MAPE result on deletion among those 4 features.

|  |  |  |  |
| --- | --- | --- | --- |
| Environment | Full data | With 1 feature deletion | With 4 features deletion |
| Linear Regression on dataset A | -300.93% | 89.33% | 89.10% |

Table 4.1: Linear Regression model MAPE results after deletion of high coefficient

The difference between the result on 1 deletion and 4 deletions seems small by 0.23%. The deletion of one feature from four features solved the issue. According to the results, we can conclude that deletion of 1 feature, Total Bsmt SF, removes major collinearity problem with the other three columns. After one column deletion, the final sample size decreased by one number of features from (2785,160) to (2785, 159).

On dataset B, multiple features had unusually high coefficients and some features are re-factored. For example, feature ‘date’ and feature ‘yr\_built’ have refactored into one feature, date – year\_built = ‘house\_age’. After deletions and refactoring, the final dataset size decreased from (21613,21) to (21613,12). For detailed coefficients on both datasets, refer to Appendix.

**4.1.3 Data Preparation**

For this dissertation, the dataset must have a training set with no missing values and a test set containing missing values. To see a variety of results over a different number of missing features, it needed categories. Each dataset was formatted into 7 categories. It is divided equally from 5% to 95%, and the category 15% means that the test set has 15% of known features and the remaining 85% are unknown features. The number of features on each category on both datasets is shown in Table 4.1.

|  |  |  |
| --- | --- | --- |
| Percentage of known features | # of features on set A | # of features on set B |
| 5% | 4 | 1 |
| 15% | 12 | 2 |
| 30% | 24 | 4 |
| 50% | 39 | 6 |
| 70% | 55 | 8 |
| 85% | 67 | 10 |
| 95% | 74 | 11 |
| Full dataset | 78 | 12 |

Table 4.1: The number of features on each category

There are two different approaches to leaving theoretical blanks on test sets: 1. Use-case approach 2. Systematic approach. The use-case approach is based on a use-case where a customer has a short amount of time to find out information on his property and he only finds out part of features. Hence, the 5% category contains the top 5% effortless features that users might easily find in the real world, let’s say in few minutes, for instance, Month sold, Year sold on dataset A.

In a similar sense, a systematic approach is based on the ranking of predictions on each feature. After testing each feature to the sale price, the highest 5% ranked features are set into the 5% category. Thus, in category 85%, the rest 15% are the features that are ranked lowest in a systematic approach.

|  |  |  |
| --- | --- | --- |
| Ranking | Name of feature | R\_squared |
| 1 | Gr Liv Area | 0.548 |
| 2 | Garage Area | 0.410 |
| 3 | Garage Cars | 0.406 |
| 4 | 1st Flr SF | 0.344 |
| 5 | Exter Qual | 0.329 |
| 6 | Kitchen Qual | 0.321 |
| 7 | Year Built | 0.309 |
| 8… | … | … |

Table 4.2: Example of the ranking system on dataset A

To give example with Table 4.2., in the systematic approach, from ground living area to 1st floor square feet, a total of 4 are picked as 5% category. The use-case approach is more focused on realistic when the systematic approach is focused on predictions. For additional information about categories on both datasets, refer to Appendix B.

**4.1.4 Learning Curve**

This learning curve shows the learning performance over time, in terms of sample size. This step was necessary because if the sample size were too small, it would be pointless to investigate reasonable output. The following are learning curves on both datasets with a linear regression algorithm.

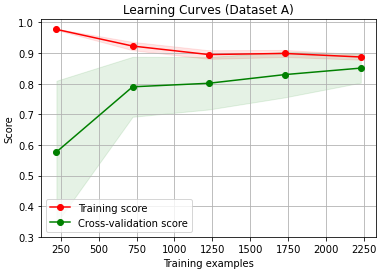
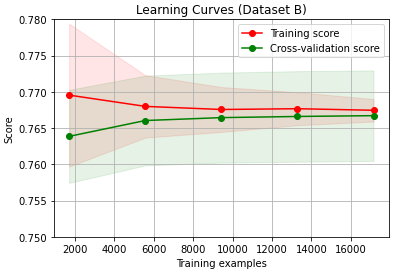


Figure 4.2 Learning Curve on A Figure 4.3 Learning Curve on B

Table 4.2 shows that there is a possibility that the result may better if we have a larger sample size. But still, the learning curve seems acceptable, especially when there is a limitation on adding more data. Table 4.3 shows that dataset B has enough sample size as the lines getting closer as they reach 6,000 samples.

**4.2 AVM Results**

**4.2.1 Evaluation metric**

This chapter gives results on different methods. There are four different regression model evaluation metrics, MAE, MSE, RMSE, and R-squared.

* Mean Absolute Error (MAE): the difference between the original and predicted values extracted by averaged the absolute difference
* Mean Squared Error (MSE): the difference between the original and predicted values extracted by squared the average difference
* Root Mean Squared Error (RMSE): error rate by the square root of MSE to give a clear view by providing a large number.
* R-squared: It is also called the coefficient of determination and it represents the percentage of the response variable variation, meaning how close the data are to the fitted regression line.

**4.2.2 Comparison**

**4.2.2.1 Deletion of missing data**

In this subsection, there are deletions of missing data models on both datasets. On this deletion section, the category of 15% means only 15% of the full data is being tested. Following table 4.2 and 4.3 are the result of dataset A with two different approaches.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 47548 | 40018 | 25140 | 21173 | 18859 | 17907 | 17716 | 16510 |
| MSE | 4901149788 | 3221328514 | 1587883672 | 1117779601 | 981081085 | 956947544 | 911645295 | 742568256 |
| RMSE | 69822 | 56440 | 38876 | 32746 | 30640 | 30161 | 29532 | 26925 |
| **R^2** | **0.226** | **0.489** | **0.754** | **0.821** | **0.837** | **0.844** | **0.845** | **0.878** |

Table 4.2: Dataset A: deletion of features, Use-case approach

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 30464 | 21815 | 18506 | 18110 | 17611 | 17650 | 17460 | 16510 |
| MSE | 2114972316 | 1210707380 | 1008236781 | 995151484 | 949098419 | 937962580 | 891651075 | 742568256 |
| RMSE | 45740 | 34143 | 30816 | 30199 | 29936 | 29978 | 29299 | 26925 |
| **R^2** | **0.663** | **0.803** | **0.844** | **0.847** | **0.853** | **0.850** | **0.859** | **0.878** |

Table 4.3: Dataset A: deletion of features, systematic approach

Although multicollinearity has been handled by deleting one column (Chapter 4.1.3), it was worth trying Ridge Regression which is strong against multicollinearity. It seems the result is similar when there are most known features, but there is a huge difference (R\_Squared < 0.437) in the low percentage of known data between the two approaches. Meaning, in a systematic approach, that several features are affecting the result hard so that even if there is only 30% of dataset presence, there’s a relatively small difference to the result of the full dataset. The systematic approach shows that if there is sufficient power with a small number of features, the deletion method works, even though it lost part of its data set. Following Table 4.4 and 4.5 represent the result with dataset B.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 220821 | 207776 | 161512 | 147201 | 136455 | 127333 | 129389 | 125843 |
| MSE | 122272124534 | 103458574057 | 64192553548 | 55916115105 | 45249155640 | 41822508173 | 43167136781 | 41000867961 |
| RMSE | 348394 | 320942 | 253118 | 236164 | 212424 | 204011 | 207214 | 201701 |
| **R^2** | **0.093** | **0.227** | **0.524** | **0.584** | **0.663** | **0.689** | **0.679** | **0.695** |

Table 4.4: Dataset B: deletion of features, Use-case approach

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 173833 | 164733 | 162541 | 157090 | 134277 | 135518 | 128476 | 125843 |
| MSE | 6852794471 | 62907564075 | 61967404807 | 57537189668 | 4558713531 | 46162559834 | 42542010569 | 41000867961 |
| RMSE | 261169 | 250235 | 248060 | 239660 | 213121 | 214489 | 205421 | 201701 |
| **R^2** | **0.490** | **0.533** | **0.541** | **0.573** | **0.662** | **0.657** | **0.686** | **0.695** |

Table 4.5: Dataset B: deletion of features, systematic approach

There is a similar trend with the result of dataset A. The high percentage of known features showed stable results on both approaches, but another big difference between the two approaches on a small percentage (<50%) of known features. When we think about the real-world scenario, it would be better to avoid prediction with a small number of features, since there is a little chance that customer’s inputs are the same as the ones in systematic approach picks. Especially when the number of all features is less than 20 like our dataset B. But, the result with deletion brought out to be genuinely higher than expected, compared to other imputation methods that follow up next.

**4.2.2.2 LR and RR with mean imputation**

In this section, the mean imputation method is tested with Linear Regression and Ridge Regression. To clarify the steps, if a certain feature is missing on the test set, we get a mean of that certain feature from the training set and fill it into the test set. Then, fit LR or RR to predict.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 57592 | 57218 | 48993 | 26810 | 22929 | 19039 | 18154 | 16510 |
| MSE | 6248489203 | 5973483292 | 4450774405 | 1642295186 | 1255275296 | 1007111627 | 958886654 | 742568256 |
| RMSE | 78877 | 76939 | 66609 | 40324 | 35225 | 30909 | 30002 | 26925 |
| **R^2** | **0.01** | **0.05** | **0.293** | **0.739** | **0.799** | **0.839** | **0.847** | **0.87** |

Table 4.6: Dataset A: LR with mean imputation, Use-case approach

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 39777 | 26680 | 22025 | 20920 | 21165 | 19758 | 17747 | 16510 |
| MSE | 3339005049 | 1775732093 | 123188996 | 1163925880 | 1110830007 | 1022911675 | 924,944900 | 742568256 |
| RMSE | 57578 | 41804 | 34553 | 32841 | 32908 | 31464 | 29446 | 26925 |
| **R^2** | **0.468** | **0.717** | **0.804** | **0.815** | **0.820** | **0.836** | **0.851** | **0.878** |

Table 4.7: Dataset A: LR with mean imputation, systematic approach

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 0.415 | 0.405 | 0.319 | 0.310 | 0.274 | 0.209 | 0.200 | 0.196 |
| MSE | 0.279 | 0.264 | 0.159 | 0.150 | 0.115 | 0.072 | 0.066 | 0.064 |
| RMSE | 0.529 | 0.514 | 0.399 | 0.387 | 0.340 | 0.268 | 0.257 | 0.254 |
| **R^2** | **-0.009** | **0.045** | **0.425** | **0.457** | **0.582** | **0.740** | **0.760** | **0.766** |

Table 4.8: Dataset B: LR with mean imputation, Use-case approach

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 11183251225 | 0.327 | 0.308 | 0.294 | 0.219 | 0.221 | 0.199 | 0.196 |
| MSE | 5.352e+20 | 0.170 | 0.151 | 0.136 | 0.079 | 0.080 | 0.066 | 0.064 |
| RMSE | 14749732081 | 0.413 | 0.389 | 0.369 | 0.281 | 0.283 | 0.258 | 0.254 |
| **R^2** | **-1.936** | **0.383** | **0.453** | **0.500** | **0.714** | **0.710** | **0.759** | **0.766** |

Table 4.9: Dataset B: LR with mean imputation, systematic approach.

Table 4.6 – 4.9 are results of mean imputation with linear regression on both datasets with two different approaches. The first thing to notice is that results with dataset B (Table 4.8,4.9) have different units. It is because, on dataset B, only the ‘price’ feature had relatively large numbers compared to other features. By taking a log of the target, it compresses outliers making the distribution normal. The result overall similar to the deletion method on high percentage categories (< 70%), but it looks unstable on low percentage categories. R-Squared is not normally linear to the percentage of features, especially on the use-case approach on both datasets.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 58020 | 56562 | 49005 | 26561 | 21509 | 17877 | 17844 | 17656 |
| MSE | 6306888285 | 5929240846 | 4541210779 | 1652718772 | 1214323645 | 955324898 | 924822477 | 1005904918 |
| RMSE | 79236 | 76914 | 67257 | 40335 | 34550 | 30075 | 29800 | 30096 |
| **R^2** | **0.003** | **0.06** | **0.281** | **0.741** | **0.807** | **0.848** | **0.849** | **0.855** |
| alpha | 10.0 | 10.0 | 10.0 | 10.0 | 10.0 | 10.0 | 10.0 | 10.0 |

Table 4.10: Dataset A: RR with mean imputation, Use-case approach

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 39731 | 26468 | 20886 | 19961 | 19099 | 18433 | 17528 | 17656 |
| MSE | 3331558302 | 1768409663 | 1199489301 | 1066501674 | 985177202 | 957411147 | 932520836 | 1005904918 |
| RMSE | 57587 | 41810 | 34119 | 32123 | 30662 | 30442 | 29552 | 30096 |
| **R^2** | **0.473** | **0.719** | **0.812** | **0.829** | **0.846** | **0.844** | **0.854** | **0.855** |
|  | 10.0 | 10.0 | 10.0 | 10.0 | 10.0 | 10.0 | 10.0 | 10.0 |

Table 4.11: Dataset A: RR with mean imputation, systematic approach

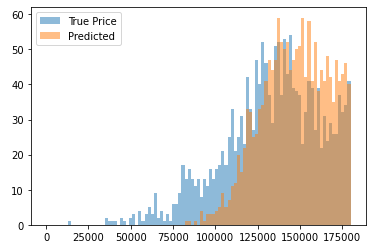
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 244299 | 235779 | 180443 | 172197 | 156498 | 145577 | 125759 | 125748 |
| MSE | 143483215404 | 132918933995 | 83604345680 | 76005852772 | 55506349688 | 47754530249 | 40984542518 | 40950694754 |
| RMSE | 377956 | 363940 | 288397 | 274920 | 235049 | 218062 | 201580 | 201824 |
| **R^2** | **-0.065** | **0.014** | **0.380** | **0.437** | **0.588** | **0.645** | **0.697** | **0.697** |

Table 4.12: Dataset B: RR with mean imputation, Use-case approach

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 189778 | 159415 | 166776 | 171442 | 148099 | 150639 | 127347 | 125748 |
| MSE | 93697131975 | 67406807566 | 65320330635 | 62176350764 | 48988788950 | 49727496564 | 42105171108 | 40950694754 |
| RMSE | 305041 | 259297 | 255026 | 249094 | 221170 | 222648 | 204839 | 201824 |
| **R^2** | **0.307** | **0.500** | **0.515** | **0.539** | **0.636** | **0.630** | **0.688** | **0.697** |
| alpha | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 125748 |

Table 4.13: Dataset B: RR with mean imputation, systematic approach

Table 4.10 – 13 also represents the mean imputation but with Ridge regression. This time log of the price wouldn’t be necessary, because the L2 penalty from Ridge does a similar job for the model. Based on background research, the mean imputation is one of the popular imputation methods that people usually suggest. It is still powerful when there are small numbers of missing observations (for instance, 95% known in category), but unlike the expectation, the result seems disappointing especially from 5 to 30%.

Overall, the difference between LR and RR was small, and it shows that mean imputation acts worse than deletion of missing data. The reason why is that each variable in a dataset has much more information and mean imputation reduces the variance of the data and that it changes the variance of the dataset. Followings are the histograms of the true price distribution, along with distributions of predictions in the full dataset, half, 30%, 5%, respectively on dataset A.

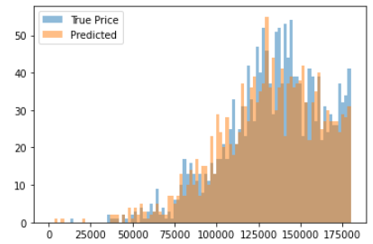
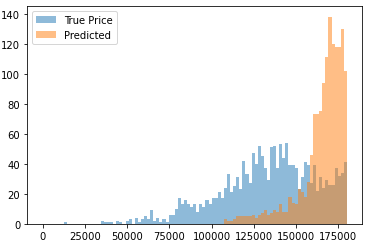
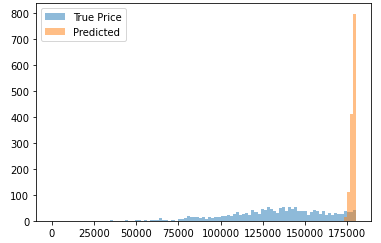


Figure 4.4. Mean imputation histogram of full dataset to 5%, left to right.

Three histograms show that the fewer features are known, the more the overall shape of the prediction graph is tilted to the right. Regardless of the shape of the actual price graph, the predictions on all four histograms are getting narrow while keeping their shape.

**4.2.2.3 Regression imputation**

The previous testing shows small differences between LR and RR, thereby the regression imputation is tested with linear regression.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 47527 | 39973 | 25049 | 21233 | 18788 | 17554 | 17530 | 16510 |
| MSE | 4959045614 | 3215411549 | 1557958634 | 1109255910 | 994890369 | 948530235 | 928527939 | 742568256 |
| RMSE | 70132 | 56545 | 39008 | 32680 | 30822 | 30140 | 29427 | 26925 |
| **R^2** | **0.208** | **0.491** | **0.753** | **0.822** | **0.841** | **0.845** | **0.852** | **0.878** |

Table 4.14: Dataset A: regression imputation, use-case approach

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 30437 | 21753 | 18443 | 18044 | 17764 | 17882 | 17951 | 16510 |
| MSE | 2121715667 | 1228716163 | 977855595 | 993163117 | 949980068 | 980815242 | 927066038 | 742568256 |
| RMSE | 45552 | 34089 | 30788 | 30198 | 29734 | 30327 | 29861 | 26925 |
| **R^2** | **0.667** | **0.796** | **0.845** | **0.842** | **0.848** | **0.842** | **0.854** | **0.878** |

Table 4.15: Dataset A: regression imputation, systematic approach

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 220764 | 207737 | 164150 | 147681 | 131760 | 127877 | 127434 | 125843 |
| MSE | 122174387984 | 103355127772 | 6421312810 | 55411920913 | 44187552871 | 42368473764 | 42059419099 | 41000867961 |
| RMSE | 348732 | 320362 | 252886 | 235232 | 209768 | 205538 | 204834 | 201701 |
| **R^2** | **0.095** | **0.235** | **0.524** | **0.588** | **0.673** | **0.684** | **0.687** | **0.695** |

Table 4.16: Dataset B: regression imputation, use-case approach

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 173972 | 169101 | 162506 | 157128 | 135061 | 135285 | 128254 | 125843 |
| MSE | 68597281372 | 63164811465 | 61975756090 | 57587039107 | 46179449594 | 46168047442 | 42586501988 | 41000867961 |
| RMSE | 261437 | 253120 | 248100 | 239699 | 214622 | 214250 | 205680 | 201701 |
| **R^2** | **0.487** | **0.501** | **0.540** | **0.571** | **0.658** | **0.657** | **0.684** | **0.695** |

Table 4.17: Dataset B: regression imputation, systematic approach

Unlike the mean imputation method, category of 15% and 30% are on the right track. Even systematic approach on dataset A (Table 4.15) has R\_Squared 0.667 on 5% category, which is highest among all methods. Although overall results aren’t significantly higher than the deletion method, it is much higher than the mean imputation and surely more stable than the mean imputation method. Therefore, it is certain to use rather a regression imputation than the mean imputation especially when filling missing features in the test set.

**4.2.2.4 Custom KNN Imputation**

The last imputation method is the custom K-nearest neighbor imputation method.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 64535 | 53717 | 48351 | 25493 | 20462 | 19651 | 17874 | 16510 |
| MSE | 7920335358 | 6023196080 | 4562676931 | 1400200489 | 1049586407 | 1007067754 | 925320502 | 742568256 |
| RMSE | 88904 | 77414 | 67418 | 37104 | 32057 | 30893 | 29600 | 26925 |
| **R^2** | **-0.269** | **0.036** | **0.262** | **0.780** | **0.829** | **0.843** | **0.853** | **0.878** |

Table 4.14: Dataset A: regression imputation, use-case approach

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 33483 | 24302 | 22161 | 20171 | 18366 | 18281 | 17743 | 16510 |
| MSE | 2446245490 | 1373270459 | 1193632045 | 1064611561 | 933564136 | 934139646 | 897138288 | 742568256 |
| RMSE | 49319 | 36630 | 34159 | 1064611561 | 30010 | 30130 | 29035 | 26925 |
| **R^2** | **0.599** | **0.784** | **0.802** | **0.826** | **0.850** | **0.851** | **0.859** | **0.878** |

Table 4.15: Dataset A: regression imputation, systematic approach

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 0.443 | 0.422 | 0.331 | 0.334 | 0.293 | 0.202 | 0.198 | 0.196 |
| MSE | 0.318 | 0.282 | 0.171 | 0.177 | 0.137 | 0.067 | 0.065 | 0.064 |
| RMSE | 0.563 | 0.530 | 0.414 | 0.421 | 0.370 | 0.259 | 0.255 | 0.254 |
| **R^2** | **-0.110** | **0.013** | **0.399** | **0.380** | **0.522** | **0.764** | **0.762** | **0.766** |

Table 4.16: Dataset B: regression imputation, use-case approach

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| %/result | **5%** | **15%** | **30%** | **50%** | **70%** | **85%** | **95%** | **100%** |
| MAE | 0.379 | 0.344 | 0.330 | 0.324 | 0.286 | 0.217 | 0.188 | 0.196 |
| MSE | 0.224 | 0.186 | 0.171 | 0.165 | 0.126 | 0.079 | 0.064 | 0.064 |
| RMSE | 0.473 | 0.431 | 0.413 | 0.406 | 0.345 | 0.281 | 0.231 | 0.254 |
| **R^2** | **0.167** | **0.300** | **0.404** | **0.425** | **0.556** | **0.722** | **0.706** | **0.766** |

Table 4.17: Dataset B: regression imputation, systematic approach

From the results of the table from 4.14 to 4.17, the custom KNN imputation method performs significantly badly on 5%. Mid-range (15%-50%) performance is similar to linear regression and deletion, but high percentages (85% – 95%) scored highest among all methods especially on dataset B. The result seems understandable because when meaning imputation and regression imputation treat observations in a missing feature as a whole, custom KNN imputation counts each into each. For example, pretend there are 3 missing observations on both the mean imputation model and custom KNN model, the mean imputation model will substitute the same 1 variable into 3 observations. On the other hand, the KNN model treats each observation separately, so that 3 different variables are substituted into 3 observations and that gives the model diversity of the data.

**Chapter 5**

**Progress**

This chapter describes the progress that has been made.

**5.1 Project Management**

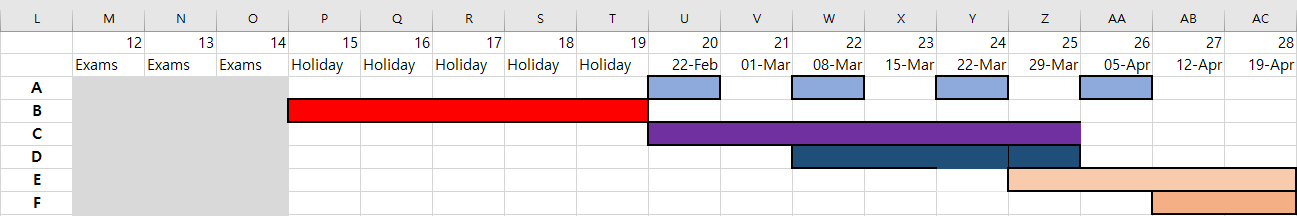
~~For updated Gannt chart~~

1. ~~Evaluate the newly found technique and compare with others~~
2. ~~Background research on techniques that handle missing value problem~~
3. ~~Implement own revised-KNN algorithm for imputation~~
4. ~~Present and understand prediction interval~~
5. ~~Conclude the research by choosing the best technique and ML algorithm~~
6. ~~Write final dissertation~~

~~테이블이(가) 표시된 사진

자동 생성된 설명~~

~~Figure 5.1: Previous Gannt chart from the proposal~~

**~~~~**

~~Figure 5.2: Updated Gannt chart for future plan~~

~~There are changes that have been made in future project plan. More different machine learning algorithm plans to be tested in updated plan. Since most of testing environment implementation are done in first semester, it would be easier focusing on testing in second semester.~~

~~Due to nature of the research, there is no promise of following the plan perfectly. Some plans might be added with high percentage and some plans might be discarded. Above Gannt chart is the big picture of the next semester.~~

**Chapter 6**

**Reflection**

**References**

[1] Kok,Nils; Koponen, Elija-Leena; Martinez-Barbosa, Carmen Adriana. Big Data in Real Estate? From Manual Appraisal to Automated Valuation (2017)

[2] Andrew Fortelny, Dr Richard Reed. The increasing use of automated valuation models in Australian mortgage market (2005)

[3] Alexander N.Bogin; Jessica Shui. Appraisal Accuracy and Automated Valuation Models in Rural Area (2020)

# [4][Faishal Ibrahim, Muhammad](https://www.ingentaconnect.com/search;jsessionid=780hn8ucuc45b.x-ic-live-02?option2=author&value2=Faishal+Ibrahim,+Muhammad" \o "Search for articles by this author); [Jam Cheng, Fook](https://www.ingentaconnect.com/search;jsessionid=780hn8ucuc45b.x-ic-live-02?option2=author&value2=Jam+Cheng,+Fook); [How Eng, Kheng](https://www.ingentaconnect.com/search;jsessionid=780hn8ucuc45b.x-ic-live-02?option2=author&value2=How+Eng,+Kheng). Automated valuation model: an application to the public housing resale market in Singapore (2005)

[5] William J.McCluskey, Michael Mccord, Peadar Thomas Davis, Martin Haran, D.Mclhatton. Prediction accuracy in mass appraisal: A comparison of modern approaches (2013)

[6] Peter Rossini; Paul John Kershaw. Automated valuation model accuracy: some empirical testing (2008)

[7] Sander Scheurwater, Atuomated Valuation Models – which future? (2018)

[8] Su X; Yan X; Tsai C. Linear regression, Volume 4, issue 3, pp. 275-294 (2012)

[9] Nikolas M; Imputations: Benefits, Risks, and a Method for Missing Data (2013)

[10] Geron, A. Hands-on machine learning with Scikit-Learn, Keras and TensorFlow: concepts, tools, and techniques to build intelligent systems (2nd ed.). page 92-140 (2019)

[11] Tan Yigitcanlar. Smart cities: an effective urban development and management model? (2015)

[12] Gregory S. Yovanof, George N. Hazapis. An architectural framework and enabling wireless technologies for digital cities and intelligent urban environments (2009)

[13] Yaling Zheng. Machine Learning with incomplete information (2011)

[14] Joachim Schork. Regression imputation Stochastic vs. Deterministic and R example https://statisticsglobe.com/regression-imputation-stochastic-vs-deterministic/

# [15] [Grigorios Papageorgiou](javascript:;), [Stuart W Grant](javascript:;), [Johanna J M Takkenberg](javascript:;), [Mostafa M Mokhles](javascript:;). Statistical primer: how to deal with missing data in scientific research? (2018)

# [16] UCLA Seminar. Multiple Imputation In Stata. <http://stats.idre.ucla.edu/stata/seminars/mi_in_stata_pt1_new/>. Accessed 10-December-2020

**Appendix A**

**Data Feature Variables Description**

The description below is detailed explanation on each feature variable to give more understanding of the dataset.

**MSSubClass**: Identifies the type of dwelling involved in the sale.

**MSZoning**: Identifies the general zoning classification of the sale.

**LotFrontage**: Linear feet of street connected to property

**LotArea**: Lot size in square feet

**Street**: Type of road access to property

**Alley**: Type of alley access to property

**LotShape**: General shape of property

**LandContour**: Flatness of the property

**Utilities**: Type of utilities available

**LotConfig**: Lot configuration

**LandSlope**: Slope of property

**Neighborhood**: Physical locations within Ames city limits

**Condition1**: Proximity to various conditions

**Condition2**: Proximity to various conditions (if more than one is present)

**BldgType**: Type of dwelling

**HouseStyle**: Style of dwelling

**OverallQual**: Rates the overall material and finish of the house

**OverallCond**: Rates the overall condition of the house

**YearBuilt**: Original construction date

**YearRemodAdd**: Remodel date (same as construction date if no remodeling or additions)

**RoofStyle**: Type of roof

**RoofMatl**: Roof material

**Exterior1st**: Exterior covering on house

**Exterior2nd**: Exterior covering on house (if more than one material)

**MasVnrType**: Masonry veneer type

**MasVnrArea**: Masonry veneer area in square feet

**ExterQual**: Evaluates the quality of the material on the exterior

**ExterCond**: Evaluates the present condition of the material on the exterior

**Foundation**: Type of foundation

**BsmtQual**: Evaluates the height of the basement

**BsmtCond**: Evaluates the general condition of the basement

**BsmtExposure**: Refers to walkout or garden level walls

**BsmtFinType1**: Rating of basement finished area

**BsmtFinSF1**: Type 1 finished square feet

**BsmtFinType2**: Rating of basement finished area (if multiple types)

**BsmtFinSF2**: Type 2 finished square feet

**BsmtUnfSF**: Unfinished square feet of basement area

**TotalBsmtSF**: Total square feet of basement area

**Heating**: Type of heating

**HeatingQC**: Heating quality and condition

**CentralAir**: Central air conditioning

**Electrical**: Electrical system

**1stFlrSF**: First Floor square feet

**2ndFlrSF**: Second floor square feet

**LowQualFinSF**: Low quality finished square feet (all floors)

**GrLivArea**: Above grade (ground) living area square feet

**BsmtFullBath**: Basement full bathrooms

**BsmtHalfBath**: Basement half bathrooms

**FullBath**: Full bathrooms above grade

**HalfBath**: Half baths above grade

**Bedroom**: Bedrooms above grade (does NOT include basement bedrooms)

**Kitchen**: Kitchens above grade

**KitchenQual**: Kitchen quality

**TotRmsAbvGrd**: Total rooms above grade (does not include bathrooms)

**Functional**: Home functionality

**GarageType**: Garage location

**GarageYrBlt**: Year garage was built

**GarageFinish**: Interior finish of the garage

**GarageCars**: Size of garage in car capacity

**GarageArea**: Size of garage in square feet

**GarageQual**: Garage quality

**GarageCond**: Garage condition

**PavedDrive**: Paved driveway

**WoodDeckSF**: Wood deck area in square feet

**OpenPorchSF**: Open porch area in square feet

**EnclosedPorch**: Enclosed porch area in square feet

**3SsnPorch**: Three season porch area in square feet

**ScreenPorch**: Screen porch area in square feet

**PoolArea**: Pool area in square feet

**PoolQC**: Pool quality

**Fence**: Fence quality

**MiscFeature**: Miscellaneous feature not covered in other categories

**MiscVal**: $Value of miscellaneous feature

**MoSold**: Month Sold (MM)

**YrSold**: Year Sold (YYYY)

**SaleType**: Type of sale

**SaleCondition**: Condition of sale

**Appendix B**

**Columns in Level 1 and Level 2**

The description below is showing each category’s designated feature variables. Level 1 has total 11 feature variables and Level 2 has its own 15 plus level 1, which is total 26.

### **Level 1** Easy to get in 5 mins' (level 1) - total 11

1. MS SubClass
2. lot Area
3. Street
4. Alley
5. BldgType
6. Fireplaces
7. GarageType
8. GarageCars
9. MoSold
10. YrSold
11. SaleType

### **Level 2** 'Need a bit of time to get' (level 2) - total 15 + level 1 = 26

1. Lot shape
2. Utilities
3. Lot config
4. HouseStyle
5. YearBuilt
6. YearRemodAdd
7. RoofStyle
8. Foundation
9. Heating
10. Bedroom
11. Kitchen
12. TotRmsAbvGrd
13. GarageFinish
14. MiscFeature
15. SaleCondition

**Appendix C**

**Code Samples**

The code sample below is the script for K-fold cross validation with regression imputation. It is written in Python programming language with Scikit learn. The implementation below specifically for Level 1 stage. The list ‘eleven’ contains 67 column names out of 78 names, excluding 11 known columns.