# Missing Clinical Data Imputation

**Anusha Manur**
ALL G12
amanur
North Carolina State University
amanur@ncsu.edu

**Kartik Shah**
ALL G12
kcshah
North Carolina State University
kcshah@ncsu.edu

## 1 Background

Researchers mine clinical patient data to derive new clinical knowledge for diagnostic precision to improve medical treatment. Clinical data mining faces a key challenge of missing data as most machine learning models require completely filled datasets for learning. Unfortunately, missing data is a common occurrence in clinical studies. Reasons for missing data include lost data, patient case report forms incorrectly filled out, personnel error, limitations of measurement device, data not entered or updated and so forth. In this project we develop machine learning models to impute missingness in a clinical dataset of longitudinal multivariable laboratory test results.

## 2 Dataset

The dataset consists of test results for 13 commonly measured analytes (clinical laboratory tests) measured over a time period for 6000 patients. Each row consists of 13 columns for each analyte and each patient has a varying number of records with varying time gap between visits. We observed that for certain patients there is a lot of time lapse between consecutive readings.

As seen in fig 1, a nullity matrix which gives a data-dense display to visually pick out the missing data patterns in the dataset. Also, the sparkline on the right gives a summary of the general shape of the data completeness. We see that columns X6-X10 have more missing data when compared to the rest. This give us an idea of the missing data pattern.

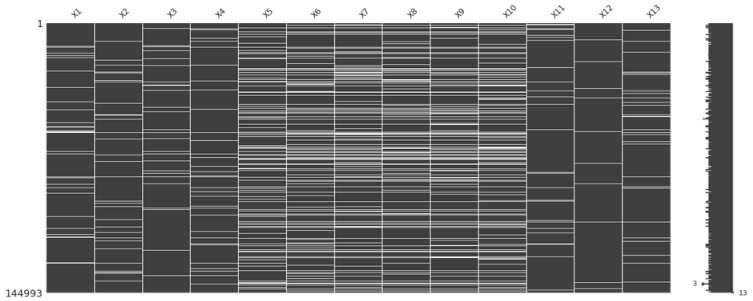


Figure 1: Missing data pattern

Table 1 gives us the percentage of missing data per analyte over 6000 patients.

Table 1: Missing Values

| X1 | X2 | X3 | X4 | X5 | X6 | X7 | X8 | X9 | X10 | X11 | X12 | X13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 5.31 | 5.47 | 5.53 | 5.39 | 16.53 | 19.15 | 19.29 | 18.62 | 18.84 | 19.42 | 4.88 | 4.83 | 6.85 |

This data analysis helps us in understanding the dataset and hence choose the right models for missing data imputation.

## 3 Methods and Experiment Setup

Our goal was to try different approaches for missing data imputation ranging from conventional methods to machine learning models.

### 3.1 Conventional methods

We applied some conventional methods of data imputation to the given clinical data.

#### 3.1.1 Simple Mean Imputation

This involved a simple mean imputation of missing values. We calculated a mean value per analyte by considering all of the 6000 patients in the training data and imputed the missing values with the mean of the respective columns. Even though this is a fast algorithm and is easy to implement, it does not factor the correlations between features. It only works on a column level. It is not very accurate and does not account for the uncertainty in the imputations. It also leads to biased estimates of the variances and covariance. When compared with the ground truth, this method results in a very high nRMSD value of 1.035.

#### 3.1.2 Simple Mean Imputation per Patient

Since each patient's features are unique and a single mean will not accommodate the variation in the clinical data over 6000 patients over a time period, we impute missing values using mean per column and per patient. With a nRMSD of 0.2990, we see a huge decrease in nRMSD error when compared to simple mean imputation.

#### 3.1.3 Back Fill

Since the patient information is in time succession, the patient may have roughly comparable values in the adjacent record, considering minimum time gap between patient visits. Back fill is an algorithm which imputes its succeeding value in the previous missing record. Since this algorithm uses the immediate records to fill missing values, consecutive missing values will lead to incorrect imputation as the error builds with each imputation. As seen in fig 1, we see that some patients have persistent missing values for some columns and this leads to inaccurate imputation resulting in an nRMSD error of 0.3033.

#### 3.1.4 Back Fill Front Fill

In this procedure, we fill in missing values from the back and front of the dataset for each patient simultaneously. This accounts for the variation in patient clinical data over a time period. However, this faces the same issue as the Backfill algorithm and leads to inaccurate results when errors in imputation get accumulated in consecutive records with missing values. We get a total error of 0.309.

#### 3.1.5 4Mean BackFrontFill

In this strategy, we made specific changes in the algorithm that got us a significant improvement and the resulted total error dropped to 0.2482. We found that certain columns were missing completely at random and certain not at random. We took four consecutive records which had a missing value. In the event that the four records had just one missing record, we take the mean of the 4 records for imputation. If it has in excess of three missing records out of four, which can lead to a biased imputation as we are considering mean, we use back and front fill for imputation. In case of four consecutive missing values in addition to the missing record in the past check as well as front check, we take the mean of all the observed data for that attribute. Though this method showed improvements for specific features, it demonstrated more prominent error for certain columns.

### 3.2 Statistical Imputation

#### 3.2.1 Multiple Imputation by Chained Equations (MICE)

Multiple imputation is a process where the missing values are filled multiple times to create "complete" datasets. The MICE algorithm works by running multiple regression models and each missing value is modeled conditionally depending on the observed (non-missing) values. We use scikit learn's Iterative Imputer class which is very flexible and can be used with a variety of estimators to do regression in a round-robin manner, treating every variable as an output in turn. We use a linear regressor and a KNN Regressor as our estimators to estimate the missing values of each feature

column. One disadvantage is that MICE depends on the observed values for a given record and the relations observed in the data for other features. This might lead to inaccurate results if our data is not suitable for regression. Using this method, we get an overall error of 0.5601 for linear regression and a value of 0.250 for a KNN regressor.

### 3.3 Machine learning methods

#### 3.3.1 Deep Learning

It is a technique that utilizes Deep Neural Networks to impute missing values. It uses hidden layers to progressively to extract higher level features from the input and updates weights in every iteration. It usually gives better results in contrast to conventional techniques. Through our deep neural network result, we observed that it demanded large datasets and it was very time consuming. Since we try to model each patient individually to retain features specific to each patient, the limited number of records per patient was not sufficient to train a deep neural network which resulted in large error value of 0.28.

#### 3.3.2 K-Nearest Neighbors

The k nearest neighbours algorithm uses 'feature similarity' to predict the values of new data points by considering the values of the nearest k neighbors. We make predictions about the missing values by finding the k closest neighbours to the observation with missing data and then impute them based on the non-missing values in the neighbourhood. The assumption behind using KNN for missing values is that a point value can be approximated by the values of the points that are closest to it, based on other variables. To get accurate imputation, we need to calibrate the value of K i.e, the number of neighbors. We pick the value of K such that it is not too low which will lead to an increase in the influence of noise or a high value of K which will tend to blur local effects. As seen in fig 2 , we choose k=5 as the optimal number of neighbours.

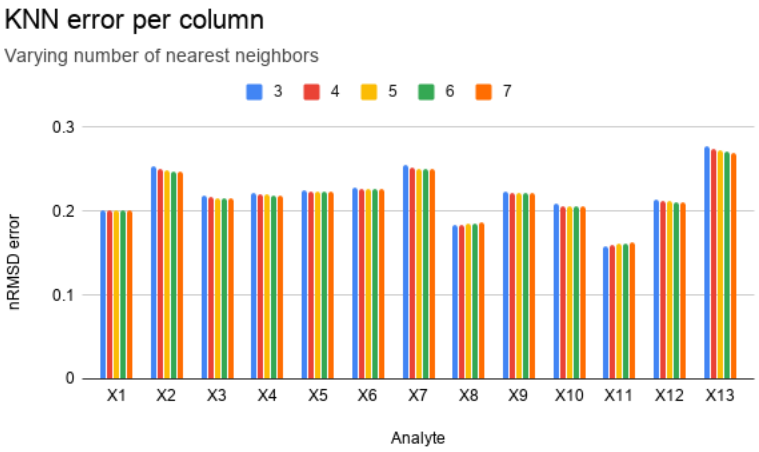


Figure 2: Varying values of K to find the optimal number of neighbours which result in least nRMSD.

## 4 Results

### 4.1 Results on training dataset

As seen in fig 3, we see that the average performance of KNN is the best on the training data across all features. From fig 4, we observe that the overall performance of KNN gives the least error for the training data when compared to all the chosen models.

In the training dataset, we vary K, the number of nearest neighbors to tune the KNN algorithm, using mean to impute missing values. We chose optimal value of K=5 which results in minimal error. The total error obtained is 0.21858563.

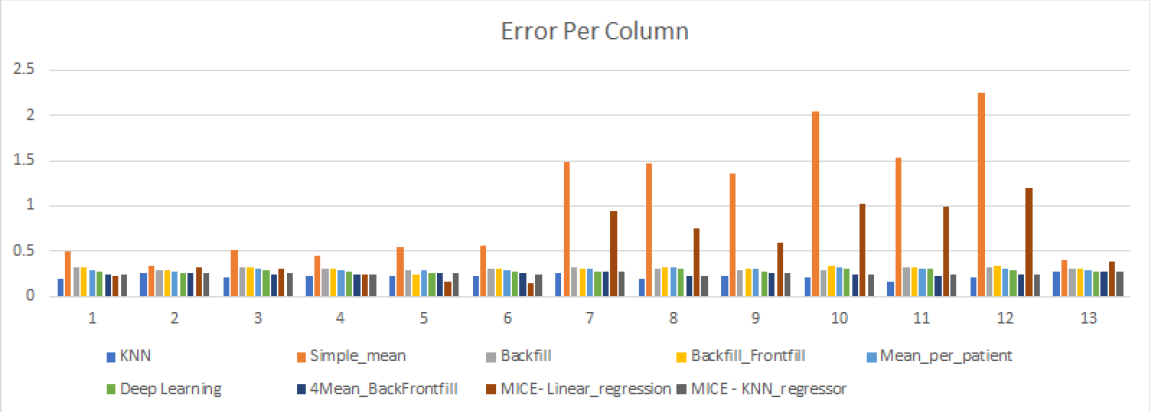


Figure 3: Errors per column for missing data imputation using different techniques

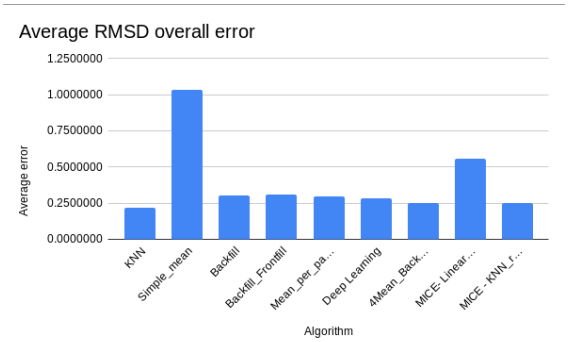


Figure 4: Overall error for the chosen algorithms

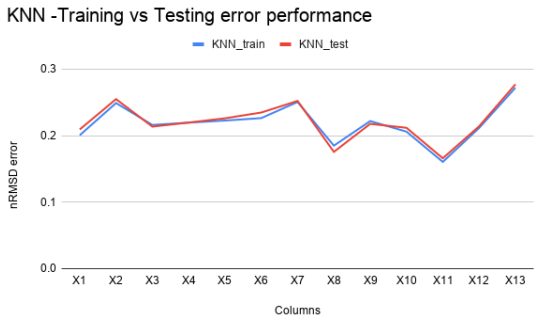


Figure 5: KNN training vs testing performance

### 4.2 Results on testing dataset

When KNN is applied on the testing dataset, results in error per column as shown in Table 2 and an average nRSMD of 0.2185.

Table 2: Testing error per column

| X1 | X2 | X3 | X4 | X5 | X6 | X7 | X8 | X9 | X10 | X11 | X12 | X13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.209 | 0.254 | 0.213 | 0.219 | 0.226 | 0.234 | 0.252 | 0.175 | 0.217 | 0.211 | 0.166 | 0.213 | 0.277 |

### 4.3 Leaderboard Results

Our model performed better compared to 3D-MICE and achieved a rank of 8.

## 5 Conclusion

To achieve a goal of missing data imputation on clinical data,we tried different approaches. According to our results as seen in Fig. 4, we observe that KNN with K=5 is the most suitable for clinical missing data imputation

Also, as seen in Fig. 5, we see that our model's testing data performance is similar to the training data performance.

### 5.1 Code Link to Github for AP1 Project

AP1 G12 KNN Imputation Code