

Missing Data Imputation Using Machine Learning Algorithm

Submitted in partial fulfillment of the requirements for
the award of
Bachelor of Engineering degree in Computer Science and Engineering

by

Pragati Bhasawar (38110416)
Pranathi Yajjala (38110418)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI - 600 119

APRIL– 2022



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **PRAGATI BHASAWAR (38110416)** and **PRANATHI YAJJALA (38110418)** who carried out the project entitled **"MISSING DATA IMPUTATION USING MACHINE LEARNING ALGORITHM "** under my supervision from November 2021 to April 2022.

Internal Guide

Dr. D . USHA NANDINI M.E., Ph. D

Head of the Department

Dr.S. VIGNESHWARI M.E., Ph.D

Submitted for Viva voce Examination held on_____

Internal Examiner

External Examiner

DECLARATION

I PRAGATI BHASAWAR (38110416) and PRANATHI YAJJALA (38110418)

hereby declare that the Project Report entitled **MISSING DATA IMPUTATION USING MACHINE LEARNING ALGORITHM** done by us under the guidance of **Dr.D. USHA NANDINI M.E., Ph. D** (Internal) is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering / Technology degree in Computer Science and Engineering.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph. D, Dean, School of Computing Dr. L. Lakshmanan M.E., Ph.D., and Dr.S. Vigneshwari M.E., Ph.D. Heads** of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **DR.D. USHA NANDINI M.E., (Ph. D)** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

In supervised learning, missing values usually appear in the training set. The missing values in a dataset may generate bias, affecting the quality of the supervised learning process or the performance of classification algorithms. These imply that a reliable method for dealing with missing values is necessary. In this paper, we analyze the difference between iterative imputation of missing values and single imputation in real-world applications. We propose an EM-style iterative imputation method, in which each missing attribute-value is iteratively filled using a predictor constructed from the known values and predicted values of the missing attribute-values from the previous iterations. Meanwhile, we demonstrate that it is reasonable to consider the imputation ordering for patching up multiple missing attribute values, and therefore introduce a method for imputation ordering. We experimentally show that our approach significantly outperforms some standard machine learning methods for handling missing values in classification tasks.

TABLES OF CONTENTS

CHAPTER NO.	CHAPTER NAME	PAGE NO.
	ABSTRACT	v
	LIST OF FIGURES	vi
	LIST OF TABLES	vii
	LIST OF ABBREVIATION	viii
1	INTRODUCTION	1
2	LITERATURE SURVEY	2
	2.1 EXISISTING SYSTEM	4
	2.2 PROPOSED SYSTEM	4
3	METHODOLOGY	
	3.1 MACHINE LEARNING	5
	3.2 SYSTEM DESIGN	9
	3.3 MODULES	16
4	RESULT AND DISCUSSION	21
5	CONCLUSION	25
	REFERENCES	26
	APPENDICES	
	A. SAMPLE CODE	27
	B. SCREENSHOTS	32
	C. PLAGIARISM REPORT	35

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO.
3.1	ARCHITECTURE DIAGRAM	13
3.2	FLOW CHART	13
3.3	FLOW DIAGRAM	15
4.1	APPLICATION	22

LIST OF TABLES

TABLE NO.	TABLE NAME:	PAGE NO.
3.1	MECHANISM OF MISSING DATA	14

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
MCAR	Missing Completely at Random
MAR	Missing at Random
NMAR	Not Missing at Random
K-NN	K-Nearest Neighbor

CHAPTER 1

INTRODUCTION

The missing data problem is arguably the most common issue encountered by machine learning practitioners when analyzing real-world data. In many applications ranging from gene expression in computational biology to survey responses in social sciences, missing data is present to various degrees. As many statistical models and machine learning algorithms rely on complete data sets, it is key to handle the missing data appropriately. Missing data problem is a common issue in most real-world studies. Since most statistical models and data-dependent machine learning (ML) algorithms could only handle complete data sets, the issue of how to approach missing values plays an important role in statistical inferences. Let Y be an $(N \times K)$ data matrix with i -th row $y_i = (y_{i1}, y_{i2}, \dots, y_{iK})$ where y_{ij} is the value of j -th feature for the i -th sample. Define the subset of observed values as Y_{obs} and missing values as Y_{mis} . Also, let $M = [m_{ij}]$ be the missing indicator matrix, where m_{ij} indicates whether y_{ij} is missing or not. Rubin (19776) defines three different missing mechanisms according to the conditional probability of the missingness, $\{m_{ij} = 1\}$, given the data. The mechanism of missing data is completely at random (MCAR) if the probability of missingness is independent of all data values, missing or observed, $P(m_{ij} = 0|Y) = g(\phi)$, $i = 1, \dots, N$, $j = 1, \dots, K$, where $g(\cdot)$ is a known link function and ϕ is the vector of unknown mechanism parameters. The missing mechanism is called missing at random (MAR) if the probability of missingness depends only on the observed data values, $P(m_{ij} = 0|Y) = g(Y_{\text{obs}}; \phi)$, $i = 1, \dots, N$, $j = 1, \dots, K$. Finally, the mechanism is called missing not at random (MNAR) when the probability of missingness may also depend on the unobserved data even after conditioning on the observed ones. The missing mechanism for the likelihood inferences is ignorable when the MCAR or MAR assumptions hold with the additional condition of disjoint parameter spaces of the missing mechanism and the data model (see Little and Rubin, 2014; Tsiatis, 2007, for more details). One simple approach to analyze incomplete data is complete case (CC) analysis which discards all incomplete cases. This approach is logical only if the missing rate is considerably small or the missing data mechanism is MCAR (Little and Rubin, 2014). However, if the missing mechanism is MAR or MNAR or the missing rate is considerably high, the CC approach could highly influence statistical results. This is due to the fact that CC analysis makes no use of observed features of an incomplete case.

CHAPTER 2

LITERATURE SURVEY

1) To generalize to multivariate settings, a chained equation process initializing using random sampling (IEEE 2011):

Buuren and Groothuis-Oudshoorn, 2011 Missing completely at random (MCAR). This is the highest level of randomness. It occurs when the probability of an instance (case) having a missing value for an attribute does not depend on either the known values or the missing data. In this level of randomness, any missing data treatment method can be applied without risk of introducing bias on the data; 2. Missing at random (MAR). When the probability of an instance having a missing value for an attribute may depend on the known values, but not on the value of the missing data itself; 3. Not missing at random (NMAR). When the probability of an instance having a missing value for an attribute could depend on the value of that attribute. Several methods have been proposed in the literature to treat missing data. Many of these methods, such as case substitution, were developed for dealing with missing data in sample surveys, and have some drawbacks when applied to the Data Mining context. Missing data imputation can be harmful because even the most advanced imputation method is only able to approximate the actual (missing) value. The predicted values are usually more well-behaved, since they conform with other attributes values.

2) These decision tree based imputation methods are non-parametric approaches that do not rely upon distributional assumptions on the data(IEEE 2012) :

s(Stekhoven and Bühlmann, 2012) This method imputes each missing entry x_{id} as the mean of the d th dimension of the K -nearest neighbors that have observed values in dimension. A best attribute is selected to place at the root node of the tree and create one child node for each possible value of this selected attribute. For each child node, if

it isn't a leaf node, the entire process is then repeated recursively only using those training instances that actually reach this node. . Bayesian networks are often used for the classification problems, in which a learner attempts to construct Bayesian network classifiers from a given set of training instances with class labels. Assume that all attributes are fully independent given the class, then the resulting Bayesian network classifiers are called naive Bayesian classifiers (simply NB). We run our experiments on 36 UCI datasets published on the main web site of Weka platform [21], which represent a wide range of domains and data characteristics. We downloaded these data sets in the format of arff from the main web site of Weka. Due to the simplicity, effectiveness, and efficiency, C4.5 and NB are two very important algorithms for addressing the classification problems. In this paper, we propose a very simple, effective, and efficient algorithm based on C4.5 and NB. We simply denote it C4.5-NB. In C4.5-NB, C4.5 and NB are built and evaluated independently at the training time, and the class-membership probabilities are weightily averaged according to their classification accuracies.

3) Evidence from recent literature suggests that recent advances in optimization have driven significant progress in machine learning. (IEEE 2017):

s (Bertsimas and Van Parys, 2017; Bertsimas and Mazumder, 2014), Integer and convex optimization have been applied successfully but it does apply for median and sparse regression problems. Despite the frequent occurrence and the relevance of missing data problem, many Machine Learning algorithms handle missing data in a rather naive way. However, missing data treatment should be carefully thought, otherwise bias might be introduced into the knowledge induced. In this work we analyses the use of the k-nearest neighbor as an imputation method. Imputation is a term that denotes a procedure that replaces the missing values in a data set by some plausible values. One advantage of this approach is that the missing data treatment is independent of the learning algorithm used. Missing data imputation can be harmful because even the most advanced imputation method is only able to approximate the actual (missing) value. The predicted values are usually more well-behaved, since they conform with other attributes values. In the experiments carried out, as more attributes

with missing values were inserted and as the amount of missing data increased, more simple were the induced models. Our analysis indicates that missing data imputation based on the k-nearest neighbor algorithm can outperform the internal methods used by C4.5 and CN2 to treat missing data, and can also outperform the mean or mode imputation method, which is a method broadly used to treat missing values.

4) We reconsider the missing data problem from this perspective(IEEE 2017)

n (Bertsimas and Dunn, 2017; Bertsimas et al., 2017) . order to develop optimization-based methods for imputation with improved out-of-sample performance. Imputation techniques replace missing values with approximated ones depending on the data set. Options range from simple approaches like mean imputation to more complex methods based on attribute correlations. Here are some commonly used imputation methods: 1. Case swapping Most sample surveys employ this strategy. A nonsampled instance replaces a missing data instance (for example, a person who cannot be reached); 2. This is a popular strategy. It involves replacing missing data for a property with the mean (quantitative) or mode (qualitative) of all known values; 3. Hot/cold deck. The hot deck approach replaces a missing attribute value with a value from an approximated distribution for the current data. Hot deck usually comes in two phases. The data are first clustered. In the second step, each missing data instance is assigned to a cluster. A cluster's full cases are utilised to fill in missing data. Calculate the attribute's mean or mode within a cluster. 4. Prediction model. Prediction models handle missing data in complex ways. These strategies include building a predictive model to estimate values to replace missing data. It is utilised as a class attribute, and the rest as input for the prediction model. This method is supported by the fact that qualities usually have links (correlations). A predictive model for classification or regression for qualitative and quantitative features with missing data might be built using correlations. Some of these links may be preserved if the prediction model captures them. Because the missing values are anticipated using a set of qualities, the predicted values are likely to be more consistent with this set of attributes than the genuine (unknown) values. The second flaw is the necessity for attribute correlation

2.1 EXISTING SYSTEM:

- In data mining process the biggest task of data preprocessing is missing value imputation. Imputation is a statistical process of replacing missing data with substituted values.
- Many clinical diagnostic dataset are usually incomplete. Excluding incomplete dataset from the original dataset can bring more problem than simplification. In this paper the machine learning techniques for missing value imputation have been explored using Ionosphere data from UCI repository.
- The data imputation problem has been approached using well-know machine learning techniques.
- The experiments have shown that the final classifier performance when the algorithm is used. Experiments show that popular machine learning classifier techniques were found to outperform than standard mean/mode imputation techniques.

2.2 PROPOSED SYSTEM:

- In this proposed work, a Python-based data mining system capable of diagnosing the HD using a Decision Tree has been developed. In the methodology, the UCI data repository was taken into consideration with 14 Attributes.
- In the dataset, there are few missing values (yet found to be hyperparameter), and pre-processing with such missing values is a common yet challenging problem.
- A mere substitution will give biased results from the data to be observed for HD diagnosis and will certainly affect the value of the learning process in Machine Learning.
- Therefore, in the proposed work, a missing value imputation is done, which gave better accuracy, and it is trustable.
- In this approach, we impute each missing data attribute value by predicting its data value from non-missing data attributes.

CHAPTER 3

METHODOLOGY

3.1 MACHINE LEARNING

Steps of Machine Learning

- Step 1: Gathering Data.
- Step 2: Preparing that Data.
- Step 3: Choosing a Model.
- Step 4: Training.
- Step 5: Evaluation.
- Step 6: Hyper parameter Tuning.
- Step 7: Prediction.

Introduction:

In this blog, we will discuss the workflow of a Machine learning project this includes all the steps required to build the proper machine learning project from scratch.

We will also go over data pre-processing, data cleaning, feature exploration and feature engineering and show the impact that it has on Machine Learning Model Performance. We will also cover a couple of the pre-modelling steps that can help to improve the model performance. Python Libraries that would be need to achieve the task:

- 1) Numpy
- 2) Pandas
- 3) Sci-kit Learn
- 4) Matplotlib

We can define the machine learning workflow in 3 stages.

- 1) Gathering data
- 2) Data pre-processing
- 3) Researching the model that will be best for the type of data
- 4) Training and testing the model
- 5) Evaluation

The machine learning model is nothing but a piece of code; an engineer or data scientist makes it smart through training with data. So, if you give garbage to the model, you will get garbage in return, i.e., the trained model will provide false or wrong prediction

1. Gathering Data

The process of gathering data depends on the type of project we desire to make, if we want to make an ML project that uses real-time data, then we can build an IoT system that using different sensors data. The data set can be collected from various sources such as a file, database, sensor and many other such sources but the collected data cannot be used directly for performing the analysis process as there might be a lot of missing data, extremely large values, unorganized text data or noisy data. Therefore, to solve this problem Data Preparation is done. We can also use some free data sets which are present on the internet. **Kaggle** and **UCI Machine learning Repository** used the most for making Machine learning models. Kaggle is one of the most visited websites that is used for practicing machine learning algorithms, they also host competitions in which people can participate and get to test their knowledge of machine learning.

2. Data pre-processing

Data preprocessing is one of the most important steps in machine learning. It is the most important step that helps in building machine learning models more accurately. In machine learning, there is an 80/20 rule. Every data scientist should spend 80% time for data pre-processing and 20% time to actually perform the analysis.

Data preprocessing is a process of cleaning the raw data i.e. the data is collected in the real world and is converted to a clean data set. In other words, whenever the data is gathered from different sources it is collected in a raw format and this data isn't feasible for the analysis. Therefore, certain steps are executed to convert the data into a small clean data set, this part of the process is called as data pre-processing.

As we know that data pre-processing is a process of cleaning the raw data into clean data, so that can be used to train the model. So, we definitely need data pre-processing to achieve good results from the applied model in machine learning and deep learning projects. Most of the real-world data is messy, some of these types of data are:

- 1) Missing data: Missing data can be found when it is not continuously created or due to technical issues in the application (IOT system).
- 2) Noisy data: This type of data is also called outliers; this can occur due to human errors (human manually gathering the data) or some technical problem of the device at the time of collection of data.
- 3) Inconsistent data: This type of data might be collected due to human errors (mistakes with the name or values) or duplication of data.

Three Types of Data

- 1) Numeric e.g., income, age
- 2) Categorical e.g., gender, nationality
- 3) Ordinal e.g., low/medium/high

These are some of the basic pre — processing techniques that can be used to convert raw data.

- 1) **Conversion of data:** As we know that Machine Learning models can only handle numeric features, hence categorical and ordinal data must be somehow converted into numeric features.
- 2) **Ignoring the missing values:** Whenever we encounter missing data in the data set then we can remove the row or column of data depending on our need. This method is known to be efficient but it shouldn't be performed if there are a lot of missing values in the dataset.
- 3) **Filling the missing values:** Whenever we encounter missing data in the data set then we can fill the missing data manually, most commonly the mean, median or highest frequency value is used.
- 4) **Machine learning:** If we have some missing data then we can predict what data shall be present at the empty position by using the existing data.
- 5) **Outliers' detection:** There are some error data that might be present in our data set that deviates drastically from other observations in a data set. [Example: human weight
= 800 Kg; due to mistyping of extra 0]

In tackling the issue (1), we can rapidly utilize the principal innovation to begin giving the best outcomes. We will zero in on impairing the BCD) and handicapping (CD) calculations (Bertsekas, 1999). In Problem 1, we use Algorithm 1 (likewise called select. Credit), which involves BCD or CD in Problem (1). Table 2 sums up the qualities of the capacity $c(x)$ for the U, W, V, X changes and the K-NN, SVM, and calculation learning tree. Nitty gritty solutions to questions (2), (3), (4), (5) utilizing K-NN, SVM, or enlistment are given in 2.3-2.5. See Annex A for more data. Utilizing the chart, decrease the worth of the objective capacity to no less than 0 until the assignment is finished. Consequently, the quantity of steps needed to finish the calculation $d_1^0 c(U_0, W_0, V_0; X)$ e, W_0 and V_0 address the beginning stage, X is the information, and U_0 is the contention to Equation 1. . (2) However, there is no assurance that the calculation will open essentially the entire world (1). (Wright, 2015). In the accompanying segments, we will consider three models and three methods for tackling the issue..

3.2. SYSTEM DESIGN

Description(K-Nearest Neighbors (knn)):

A single-step, greedy method which imputes missing values using the K-nearest neighbors of an observation based upon Euclidean distance.

- The candidate neighbors must have non-missing values in the imputed feature.
- Averaged distance is used if some other coordinates are missing. Implemented using the impute package in R
- K-NN based (opt.knn):
- This method solves the optimal K-nearest neighbors problem .
- Convergence time depends upon the quality of the initial warm start.
- We run both block coordinate descent and coordinate descent for small data sets of size $n \leq 10,000$, and only coordinate descent for large data sets with higher n .
- The implementation was in the programming language Julia with fast algorithms for K-nearest neighbor calculations.

First-Order Method for the General Problem

- To obtain high-quality solutions to Problem (1), we can use first-order methods with random warm starts. In particular, we will focus on block coordinate descent (BCD) and coordinate descent (CD) (Bertsekas, 1999). Algorithm 1, which we refer to as `opt.impute`, implements BCD or CD for Problem (1). The variables U, W, V , and X as well as the cost function $c(\cdot)$ are summarized in Table 2 for K-NN, SVM, and trees. The detailed solution methods for Problems (2), (3), (4), and (5) for K-NN, SVM, and tree imputation models are described in Sections 2.3-2.5, respectively.
- By construction, the objective function value strictly decreases by at least δ_0 until termination. It follows that the number of steps needed for the algorithm to terminate is $d \leq \frac{1}{\delta_0} c(U_0, W_0, V_0; X)$, where W_0, V_0 are the initialization values, X is data, and U_0 is the argmin in Equation (2). However, the algorithm is not guaranteed to find a global minimum for Problem (1) (Wright, 2015).
- In the next sections, we discuss three example models and the optimization problem formulations. For each model and each first-order method, we derive the specific updates.
- Input:

$$X \in \mathbb{R}^{n \times p_0} \times \{1, \dots, k_1\} \times \dots \times \{1, \dots, k_{p_1}\},$$

a data matrix with some missing entries

$$M = \{(i, d) : x_{id} \text{ is missing}\}, \delta_0 > 0, \text{ and}$$

warm start $W_0 \in \mathbb{R}^{n \times p_0}$,

$$V_0 \in \{1, \dots, k_1\} \times \dots \times \{1, \dots, k_{p_1}\}.$$

- Output:

$\hat{X}$ a full matrix with imputed value.

- Procedure:

Initialize $\delta \leftarrow \infty$, $\mathbf{W}^{old} \leftarrow \mathbf{W}^0$, $\mathbf{V}^{old} \leftarrow \mathbf{V}^0$.

while $\delta > \delta_0$ **do**

- ① Update $\mathbf{U}^*$ using model dependent information:

$$\begin{aligned} \mathbf{U}^* &\leftarrow \arg \min_{\mathbf{U}} c(\mathbf{U}, \mathbf{W}^{old}, \mathbf{V}^{old}; \mathbf{X}) \\ \text{s.t. } &(\mathbf{U}, \mathbf{W}^{old}, \mathbf{V}^{old}) \in \mathcal{U}. \end{aligned} \quad (2)$$

- ② Update the imputation $\mathbf{W}^*$, $\mathbf{V}^*$, following either:

- ②a) block coordinate descent (BCD):

$$\begin{aligned} \mathbf{W}^*, \mathbf{V}^* &\leftarrow \arg \min_{\mathbf{W}, \mathbf{V}} c(\mathbf{U}^*, \mathbf{W}, \mathbf{V}; \mathbf{X}) \\ \text{s.t. } &w_{id} = x_{id} \quad (i, d) \in \mathcal{N}_0, \\ &v_{id} = x_{id} \quad (i, d) \in \mathcal{N}_1, \\ &(\mathbf{U}^*, \mathbf{W}, \mathbf{V}) \in \mathcal{U}, \end{aligned} \quad (3)$$

or

- ②b) coordinate descent (CD):

$$\begin{aligned} w_{jr}^* &\leftarrow \arg \min_{w_{jr}} c(\mathbf{U}^*, \mathbf{W}, \mathbf{V}; \mathbf{X}) \\ \text{s.t. } &w_{id} = x_{id} \quad (i, d) \in \mathcal{N}_0, \\ &v_{id} = x_{id} \quad (i, d) \in \mathcal{N}_1, \\ &w_{id} = w_{id}^* \quad (i, d) \in \mathcal{M}_0 \setminus (j, r), \\ &v_{id} = v_{id}^* \quad (i, d) \in \mathcal{M}_1, \\ &(\mathbf{U}^*, \mathbf{W}, \mathbf{V}) \in \mathcal{U}, \end{aligned} \quad (4)$$

Activ

$$\begin{aligned}
v_{jr}^* &\leftarrow \arg \min_{v_{jr}} c(\mathbf{U}^*, \mathbf{W}, \mathbf{V}; \mathbf{X}) \\
\text{s.t. } \quad &w_{id} = x_{id} \quad (i, d) \in \mathcal{N}_0, \\
&v_{id} = x_{id} \quad (i, d) \in \mathcal{N}_1, \\
&w_{id} = w_{id}^* \quad (i, d) \in \mathcal{M}_0, \\
&v_{id} = v_{id}^* \quad (i, d) \in \mathcal{M}_1 \setminus (j, r), \\
&(\mathbf{U}^*, \mathbf{W}, \mathbf{V}) \in \mathcal{U}. \\
\textcircled{3} \quad &\delta \leftarrow c(\mathbf{U}^*, \mathbf{W}^*, \mathbf{V}^*; \mathbf{X}) - c(\mathbf{U}^{old}, \mathbf{W}^{old}, \mathbf{V}^{old}; \mathbf{X}). \\
\textcircled{4} \quad &(\mathbf{U}^{old}, \mathbf{W}^{old}, \mathbf{V}^{old}) \leftarrow (\mathbf{U}^*, \mathbf{W}^*, \mathbf{V}^*). \\
\text{end while} \\
\mathbf{X}^{imp} &\leftarrow [\mathbf{W}^*; \mathbf{V}^*]
\end{aligned}$$

We first define a distance metric between rows $(\mathbf{w}_i, \mathbf{v}_i)$ and $(\mathbf{w}_j, \mathbf{v}_j)$ as

$$d_{ij} := \sum_{d=1}^{p_0} (w_{id} - w_{jd})^2 + \sum_{d=p_0+1}^{p_0+p_1} \mathbf{1}_{\{v_{id} \neq v_{jd}\}}. \quad (6)$$

Next, we introduce the binary variables:

$$z_{ij} = \begin{cases} 1, & \text{if } (\mathbf{w}_j, \mathbf{v}_j) \text{ is among the } K\text{-nearest neighbors of } (\mathbf{w}_i, \mathbf{v}_i) \\ & \text{with respect to distance metric (6),} \\ 0, & \text{otherwise.} \end{cases}$$

We further define the set of indices $\mathcal{I} := \{i : \mathbf{x}_i \text{ has at least one missing coordinate}\}$. The optimization problem for the K -NN based imputation model is:

$$\begin{aligned}
\min \quad &c(\mathbf{Z}, \mathbf{W}, \mathbf{V}; \mathbf{X}) := \sum_{i \in \mathcal{I}} \sum_{j=1}^n z_{ij} \left[\sum_{d=1}^{p_0} (w_{id} - w_{jd})^2 + \sum_{d=p_0+1}^{p_0+p_1} \mathbf{1}_{\{v_{id} \neq v_{jd}\}} \right] \\
\text{s.t. } \quad &w_{id} = x_{id} \quad (i, d) \in \mathcal{N}_0, \\
&v_{id} = x_{id} \quad (i, d) \in \mathcal{N}_1, \\
&z_{ii} = 0 \quad i \in \mathcal{I}, \\
&\sum_{j=1}^n z_{ij} = K \quad i \in \mathcal{I}, \\
&\mathbf{Z} \in \{0, 1\}^{|\mathcal{I}| \times n}
\end{aligned} \quad (7)$$

(5) is among the K -nearest
duces the missing value
ach point $(\mathbf{w}_i, \mathbf{v}_i)$, $i \in \mathcal{I}$ to
 $\mathcal{d}\}$ can be modeled with
tion problem with both
is difficult to solve to
us variables only.

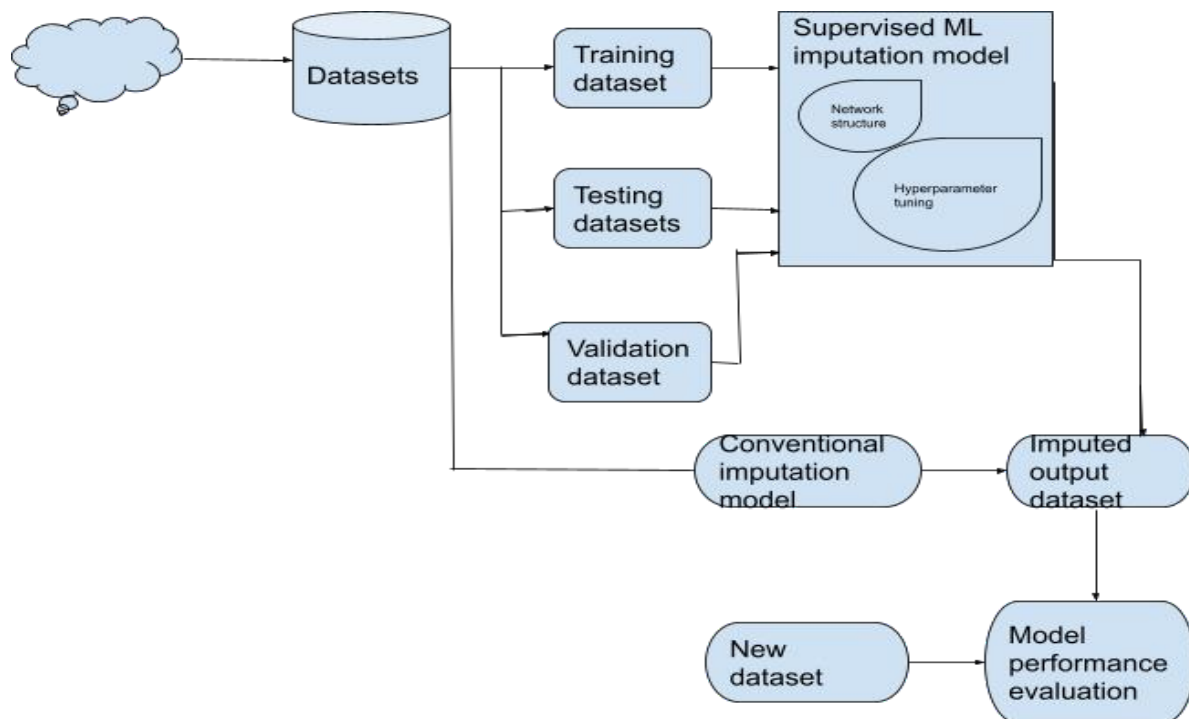


Fig.3.1: Architectural Diagram

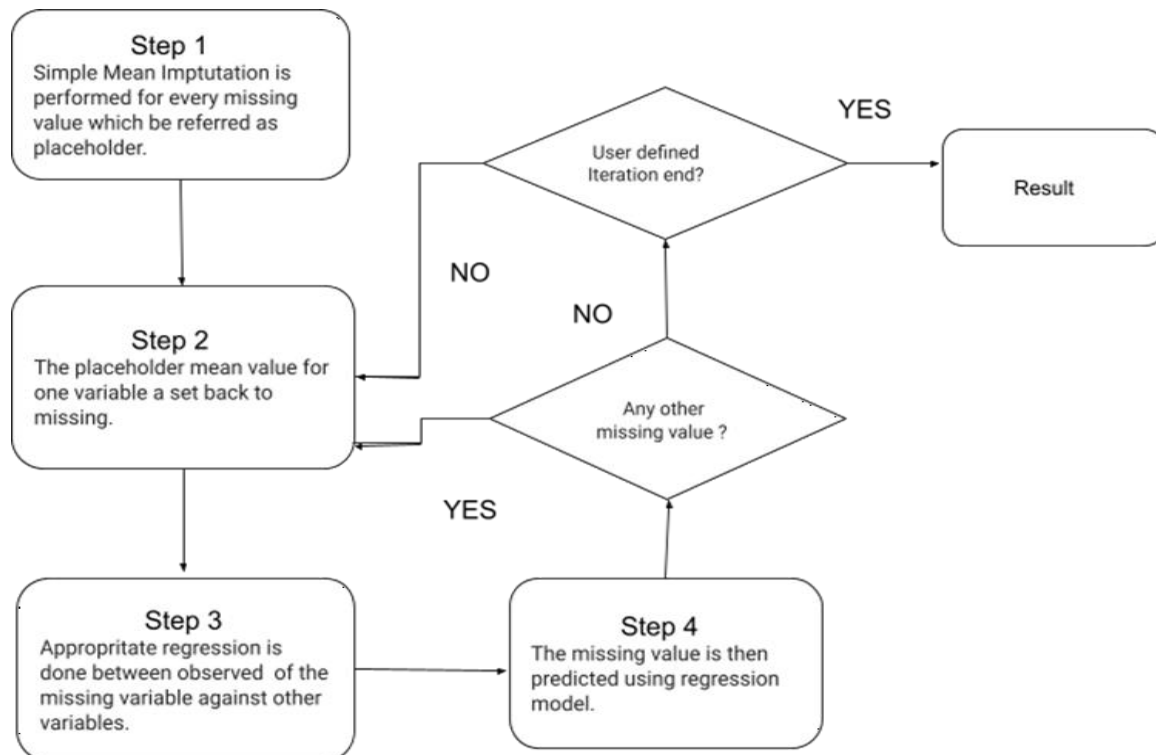


Fig.3.2: Flow chart

3.2.1 ALGORITHM USED

- K-Nearest Neighbor is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- The K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- K-NN is a **non-parametric algorithm**, which means it does not make any assumption on underlying data.
- It is also called a **lazy learner algorithm** because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

The K-NN working can be explained

- **Step-1:** Select the number K of the neighbors
- **Step-2:** Calculate the Euclidean distance of **K number of neighbors**
- **Step-3:** Take the K nearest neighbors as per the calculated Euclidean distance.
- **Step-4:** Among these k neighbors, count the number of the data points in each category.
- **Step-5:** Assign the new data points to that category for which the number of the neighbor is maximum.
- **Step-6:** Our model is ready.

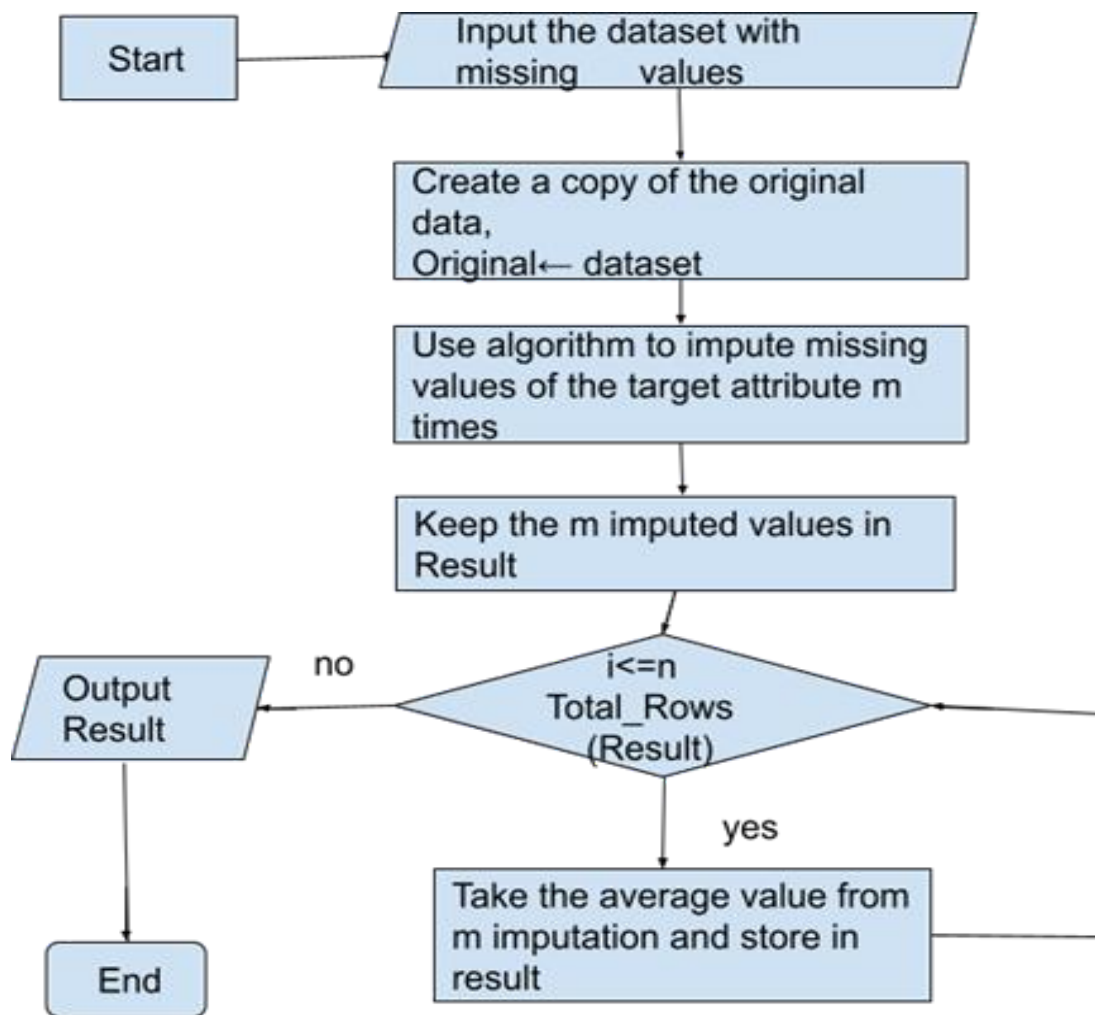


Fig 3.3: Flow diagram

Table.3.1: MECHANISM OF MISSING DATA

Missing Completely at Random (MCAR)
 Missing at Random (MAR)
 Not Missing at Random

$$P(i^b, m) = 0$$

$$P(i^b, m) = P(i^b) f(i^b, m)$$

m

(NMAR)

Problem Statement:

- Some algorithms cannot handle missing values properly, while some techniques give efficient results to estimate the missing values.
- It is very important to handle missing data because many machine learning algorithm performances reduces due to missing values.
- Missing values in the dataset have a negative impact on the imputation of accuracy; therefore, it could lead to a different output.
- some techniques are discussed which are employed to impute the missing data, and these techniques are compared using their drawbacks.

Advantages:

- handling missing data despite new advances and its potential in clinical, environmental, and health policy research.
- This review will discuss several benefits of this technique as well as the approaches that make the technique applicable to different types of data.
- Providing such examples may show investigators how considering this method may help in their future research.

Applications:

- Collecting, analyzing and drawing inferences from data is central to research in the medical and social sciences.
- The first is evaluating the reporting delay in AIDS surveillance systems to estimate survival time following AIDS diagnosis.
- An example of a school choice experiment is used to illustrate missing data and disobedience in randomized studies.

3.3 MODULES:

- Dataset collection
- Data Cleaning
- Feature Extraction
- Model training
- Testing model
- Performance Evaluation
- Prediction

3.3.1 Dataset collection

Collecting data allows you to capture a record of past events so that we can use data analysis to find recurring patterns. From those patterns, you build predictive models using machine learning algorithms that look for trends and predict future changes. Predictive models are only as good as the data from which they are built, so good data collection practices are crucial to developing high-performing models. The data need to be error-free (garbage in, garbage out) and contain relevant information for the task at hand. For example, a loan default model would not benefit from tiger population sizes but could benefit from gas prices over time. In this module, we collect the data from kaggle dataset archives. This dataset contains the information of divorce in previous years.

3.3.2 Data cleaning:

Data cleaning is a critically important step in any machine learning project. In this module data cleaning is done to prepare the data for analysis by removing or modifying the data that may be incorrect, incomplete, duplicated or improperly formatted. In tabular data, there are many different statistical analysis and data visualization techniques you can use to explore your data in order to identify data cleaning operations you may want to perform.

3.3.3 Feature Extraction:

This is done to reduce the number of attributes in the dataset hence providing advantages like speeding up the training and accuracy improvements. In machine learning, pattern recognition, and image processing, feature extraction starts from an initial set of measured data and builds derived values (features) intended to be informative and non-redundant, facilitating the subsequent learning and generalization steps, and in

some cases leading to better human interpretations. Feature extraction is related to dimensionality reduction

When the input data to an algorithm is too large to be processed and it is suspected to be redundant (e.g. the same measurement in both feet and meters, or the repetitiveness of images presented as pixels), then it can be transformed into a reduced set of features (also named a feature vector).

Determining a subset of the initial features is called feature selection. The selected features are expected to contain the relevant information from the input data, so that the desired task can be performed by using this reduced representation instead of the complete initial data.

3.3.4 Model training :

A training model is a dataset that is used to train an ML algorithm. It consists of the sample output data and the corresponding sets of input data that have an influence on the output.

The training model is used to run the input data through the algorithm to correlate the processed output against the sample output. The result from this correlation is used to modify the model. This iterative process is called “model fitting”. The accuracy of the training dataset or the validation dataset is critical for the precision of the model. Model training in machine language is the process of feeding an ML algorithm with data to help identify and learn good values for all attributes involved. There are several types of machine learning models, of which the most common ones are supervised and unsupervised learning.

In this module we use supervised classification algorithms like linear regression to train the model.

3.3.5 Testing model:

In this module we test the trained machine learning model using the test dataset, Quality assurance is required to make sure that the software system works according to the requirements. Were all the features implemented as agreed? Does the program behave as expected? All the parameters that you test the program against should be stated in the technical specification document.

Moreover, software testing has the power to point out all the defects and flaws during development. You don't want your clients to encounter bugs after the software is released and come to you waving their fists. Different kinds of testing allow us to catch bugs that are visible only during runtime.

3.3.6 Performance Evaluation:

In case the model performs poorly, we optimize the machine learning algorithms to improve the performance. Performance Evaluation is defined as a formal and productive procedure to measure an employee's work and results based on their job responsibilities. It is used to gauge the amount of value added by an employee in terms of increased business revenue, in comparison to industry standards and overall employee return on investment (ROI).

All organizations that have learned the art of "winning from within" by focusing inward towards their employees, rely on a systematic performance evaluation process to measure and evaluate employee performance regularly. Ideally, employees are graded annually on their work anniversaries based on which they are either promoted or are given suitable distribution of salary raises. Performance evaluation also plays a direct role in providing periodic feedback to employees, such that they are more self-aware in terms of their performance metrics. In this module, we evaluate the performance of trained machine learning model using performance evaluation criteria such as F1 score, accuracy and classification error.

3.3.7 Prediction :

Prediction” refers to the output of an algorithm after it has been trained on a historical dataset and applied to new data when forecasting the likelihood of a particular outcome, such as whether or not a customer will churn in 30 days. The algorithm will generate probable values for an unknown variable for each record in the new data, allowing the model builder to identify what that value will most likely be. The word “prediction” can be misleading. In some cases, it really does mean that you are predicting a future outcome, such as when you’re using machine learning to determine the next best action in a marketing campaign.

Other times, though, the “prediction” has to do with, for example, whether or not a transaction that already occurred was fraudulent.

In that case, the transaction already happened, but you’re making an educated guess about whether or not it was legitimate, allowing you to take the appropriate action.

In this module we use trained and optimized machine learning model to predict whether the patient the divers case asking some questions .

HARDWARE REQUIREMENTS

System : Pentium i3 Processor.

Hard Disk : 500 GB.

Monitor : 15” LED

Input Devices : Keyboard, Mouse

Ram : 2 GB

SOFTWARE REQUIREMENTS

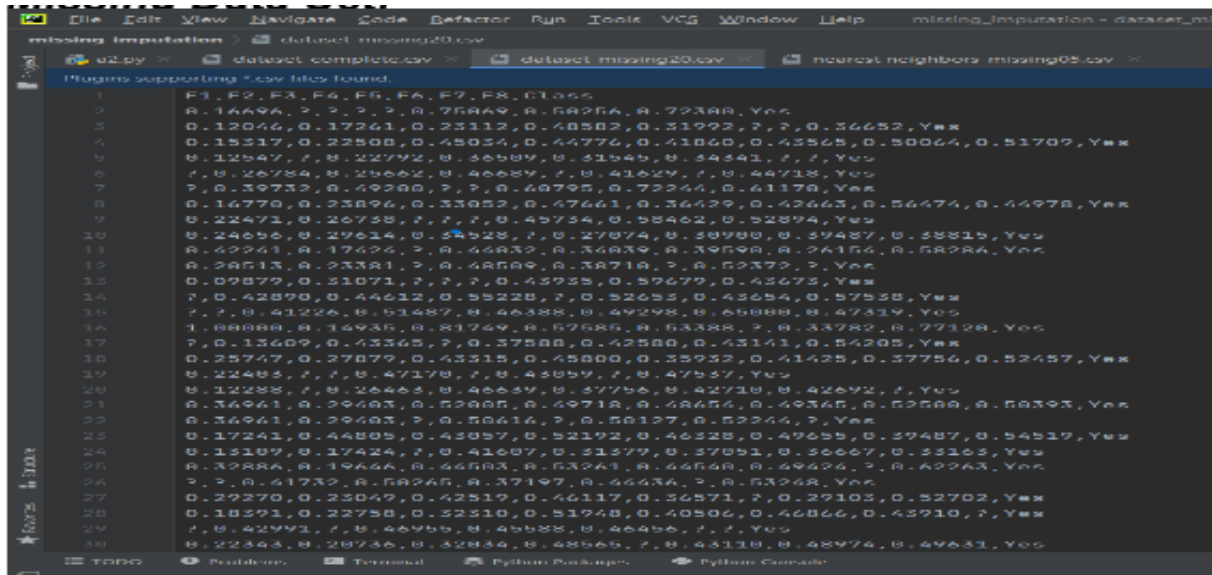
Operating system : Windows 10.

Coding Language : Python

CHAPTER 4

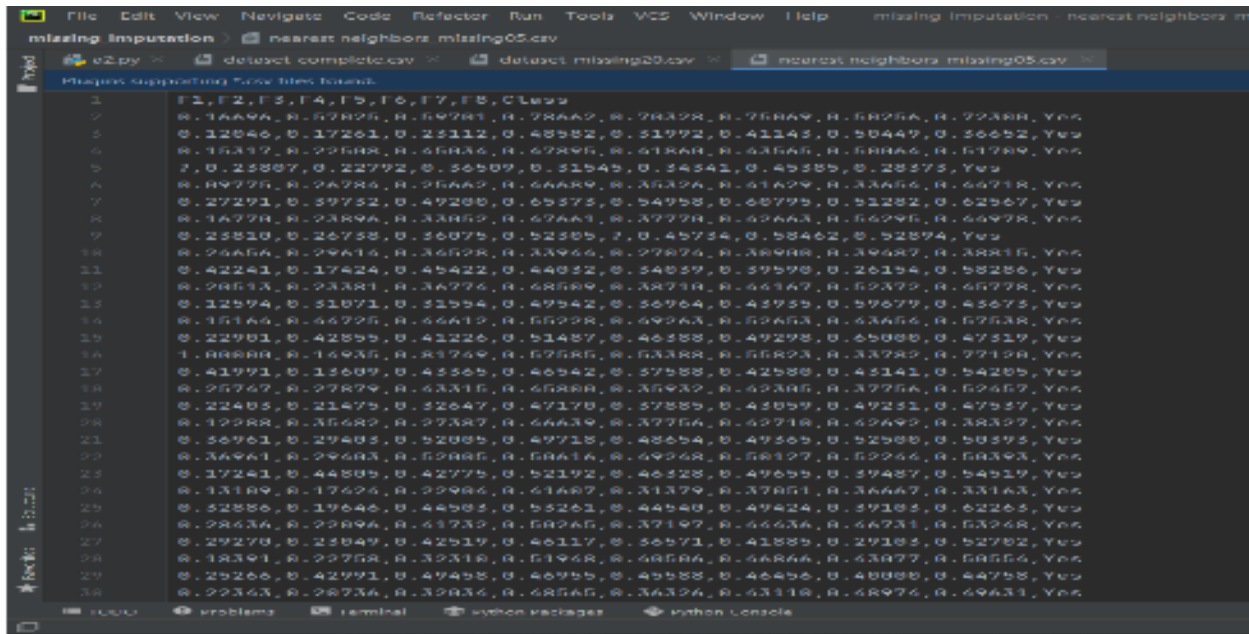
RESULTS and DISCUSSION

Here are a few ideas on the most proficient method to look or get an arrangement for tattoos. At times, the data given is utilized as a reason for exhibiting AI or data handling innovation. In different cases, it is utilized as a reason for looking at changed strategies. We utilized the technique on 84 informational indexes in the UCI Machine Learning data set. The size of this information is given from $n = 23$ to 5,875, with $p = 2$ to 124. In the accompanying areas, we first show the genuineness of everything. The fast and for the most part substitution technique prompts a decrease in MAE. Presently we show that the nature of conviction is extremely high as far as decision. Contrasted with the techniques utilized, it prompts further developed execution and lower back. We will keep on examining the outlines (K, cp, C), start temperature, how to fall (BCD or CD), the arrangement of the information, including the nature of the selfappraisal work contingent upon the missing technique. The pictures underneath shows the outcomes. Shapes of musk molecules: this dataset describes 92 molecules of which 47 are musks and 45 are non-musks. For each molecule $P=166$ features describe its conformation, but since a molecule can have many conformations due to rotating bonds, there are $n=476$ different low-energy conformations in the set. The classification into musk and non-musk molecules is removed.



```
File Edit View Navigate Code Refactor Run Tools VCS Window Help missing_imputation - dataset_mil
dataset_missing20.csv
dataset_missing20.csv
nearest_neighbors_missing05.csv
Program supporting files found.
1 P1,P2,P3,P4,P5,P6,P7,P8,Class
2 0.16696,?,?,?,?,0.75869,0.58256,0.72388,Yes
3 0.12046,0.17261,0.23112,0.48582,0.31292,?,?,0.36652,Yes
4 0.13317,0.22508,0.45034,0.44776,0.41860,0.43565,0.50064,0.51707,Yes
5 0.12567,?,0.22722,0.36589,0.51545,0.34541,?,?,Yes
6 ?,0.26724,0.25662,0.46639,?,0.41629,?,0.44713,Yes
7 0.39732,0.49280,?,?,0.48795,0.72244,0.41178,Yes
8 0.16778,0.23896,0.33852,0.47661,0.36429,0.43663,0.56474,0.44978,Yes
9 0.22471,0.26735,?,?,0.43734,0.55462,0.52594,Yes
10 0.24656,0.29614,0.35528,?,0.27874,0.38988,0.39487,0.38815,Yes
11 0.42261,0.17426,?,0.46832,0.34839,0.39598,0.26156,0.58286,Yes
12 0.28513,0.23321,?,0.48589,0.38718,?,0.52372,?,Yes
13 0.07879,0.31071,?,?,0.43235,0.59679,0.43673,Yes
14 ?,0.42590,0.44612,0.55228,?,0.52653,0.43654,0.57535,Yes
15 ?,?,0.41226,0.51487,0.46388,0.49298,0.65888,0.47319,Yes
16 1.88888,0.14935,0.81749,0.57585,0.53388,?,0.33782,0.77128,Yes
17 ?,0.13609,0.43365,?,0.37588,0.42580,0.43141,0.54205,Yes
18 0.25747,0.27879,0.43315,0.45800,0.35932,0.41425,0.37756,0.52457,Yes
19 0.22483,?,?,0.47178,?,0.43859,?,0.47537,Yes
20 0.12288,?,0.26663,0.46639,0.37756,0.42718,0.42692,?,Yes
21 0.34961,0.29483,0.52805,0.49718,0.48654,0.49365,0.52588,0.58393,Yes
22 0.34961,0.29483,?,0.50616,?,0.58127,0.52244,?,Yes
23 0.17241,0.44805,0.43057,0.52192,0.46328,0.49655,0.29487,0.54519,Yes
24 0.15189,0.17424,?,0.41607,0.51579,0.37051,0.36667,0.33163,Yes
25 0.32886,0.19646,0.46583,0.53261,0.46568,0.49424,?,0.42263,Yes
26 ?,?,0.41732,0.58265,0.37197,0.44636,?,0.53268,Yes
27 0.29270,0.25049,0.42519,0.46117,0.36571,?,0.29103,0.52702,Yes
28 0.18391,0.22738,0.32510,0.51945,0.40506,0.46866,0.43910,?,Yes
29 ?,0.42991,?,0.46955,0.45588,0.46456,?,?,Yes
30 0.22343,0.28736,0.32834,0.48565,?,0.43118,0.48974,0.49631,Yes
```

MISSING DATASET



NEW DATASET IMPUTATION

EVALUATION:

BEFORE IMPUTATION :

****Support Vector Machine****

ACCURACY= 88.62990335417851

Precision: 0.0

Recall: 0.0

Confusion Matrix :

[[1559 0]

[200 0]]

Process finished with exit code 0

AFTER IMPUTATION :

****Support Vector Machine****

ACCURACY= 90.0511654349062

Precision: 83.78378378378379

Recall: 15.5

Confusion Matrix :

[[1553 6]

[169 31]]

Process finished with exit code 0

APPLICATION:

MDVP_F0(Hz)		Shim_apq5	
MDVP_Fh(Hz)		MDVP_APQ	
MDVP_Fo(Hz)		SDDA	
MDVP__jitter		NHR	
MDVP__JABS		HNR	
MDVP__RAP		RPDE	
MDVP__PPQ		DFA	
JITTER_DDP		SPREAD1	
MDVP_Shim		SPREAD2	
MDVP_S_db		D__2	
Shim_apq3		PPE	
Submit			

Fig.4.1: *Application*

CHAPTER 5

CONCLUSION

In summary, we frame the classical missing data problem as a non-convex optimization problem based upon a variety of predictive models. We propose a family of new imputation methods, `opt.impute`, which finds high quality solutions to this problem using fast first order methods. Through extensive computational experiments on 84 data sets from the UCI Machine Learning Repository, we show that `opt.impute` yields statistically significant gains in imputation quality over state-of-the-art imputation methods, which leads to improved out-of-sample performance on downstream tasks.

This approach scales to large problem sizes, generalizes to multiple imputation, and improves over state-of-the-art methods across a broad range of missing data scenarios. We show on several real datasets coming from different biological and medical fields that `missForest` outperforms established imputation methods like *k*-nearest neighbours imputation or multivariate imputation using chained equations. Using our OOB imputation error estimates, `missForest` offers a way to assess the quality of an imputation without the need of setting aside test data nor performing laborious cross- validations. For subsequent analysis, these error estimates represent a mean of informal reliability check for each variable. The full potential of `missForest` is deployed when the data include complex interactions or non-linear relations between variables of unequal scales and different type.

REFERENCES:

- 1] Satija R, Farrell JA, Gennert D, Schier AF, Regev A. Spatial reconstruction of single- cell gene expression data. *Nature biotechnology*. 2015;33(5):495.
- 2] Bailey MH, Tokheim C, Porta-Pardo E, Sengupta S, Bertrand D, Weerasinghe A, et al. Comprehensive characterization of cancer driver genes and mutations. *Cell*. 2018;173(2):371-85. e18.
- 3] Fehrmann RS, Karjalainen JM, Krajewska M, Westra HJ, Maloney D, Simeonov A, et al. Gene expression analysis identifies global gene dosage sensitivity in cancer. *Nature genetics*. 2015;47(2):115.
- 4] Lima-Tenório MK, Pineda EAG, Ahmad NM, Fessi H, Elaissari A. Magnetic nanoparticles: In vivo cancer diagnosis and therapy. *International journal of pharmaceutics*. 2015;493(1-2):313-27.
- 5] Criscuolo E, Spadini S, Lamanna J, Ferro M, Burioni R. Bacteriophages and Their Immunological Applications against Infectious Threats. *Journal of immunology research*. 2017;2017.
- Fink O (2008) Vulnerability Identification.
- 6] Zheng, He, Wu, Hintz (2005) Character Recognition of Car Number Plates. *International Conference on Computer Vision* 1: 33-39.
- Kim KK, Kim KI, Kim JB, Kim HJ (2000) Learning Based Approach for License Plate Recognition, *Neural Networks for Signal Processing. IEEE Signal Processing Society Workshop* 2: 614-623.
- 7] Zheng Y, Li H, Doermann DS (2004) Machine Printed Text and Handwriting Identification in Noisy Document Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26: 337-353.
- 8] Zhao B, Liu Y, Xia SW (2000) Support Vector Machine and Its Application in Handwritten Numeral Recognition. *Proceeding's 15th International Conference on Pattern Recognition* 2: 720-723.

APPENDICES

A. Sample Code:

Complete Data Accuracy:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn import svm
from sklearn.metrics import accuracy_score, precision_score, recall_score, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
model = svm.SVC(kernel='linear')
scaler = StandardScaler()
def trainData():
    parkinsons_data = pd.read_csv('dataset_complete.csv')
    X = parkinsons_data.drop(columns=['Class'], axis=1)
    Y = parkinsons_data['Class']
# displaying heatmap
plt.show()
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=2)
scaler.fit(X_train)
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
from sklearn.metrics import confusion_matrix
model.fit(X_train, Y_train)
prediction=model.predict(X_test)
cf_matrix = confusion_matrix(Y_test, prediction)
acc = accuracy_score(Y_test, prediction)
print("****Support Vector Machine**** ")
print("ACCURACY= ", accuracy_score(Y_test, prediction)*100)
print("Precision:", precision_score(Y_test, prediction)*100)
print("Recall:", recall_score(Y_test, prediction)*100)
print("Confusion Matrix :\n", cf_matrix)
cm = confusion_matrix(Y_test, prediction)
plt.figure(figsize=(12, 12))
sns.heatmap(cm, annot=True, fmt=".0f", linewidths=.5, square=True, cmap='Blues');
plt.ylabel('Actual label');
plt.xlabel('Predicted label');
all_sample_title = 'Confusion Matrix of SVM for complete data'
plt.title(all_sample_title, size=15); plt.show()
```

Missing Data Accuracy:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn import svm
from sklearn.metrics import accuracy_score, precision_score,
recall_score, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
model = svm.SVC(kernel='linear')
scaler = StandardScaler()
def trainData():
    parkinsons_data = pd.read_csv('dataset_missing05R.csv')
    X = parkinsons_data.drop(columns=['Class'], axis=1)
    Y = parkinsons_data['Class']
# displaying heatmap
plt.show()
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=2)
scaler.fit(X_train)
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
model.fit(X_train, Y_train)
prediction=model.predict(X_test)
cf_matrix = confusion_matrix(Y_test, prediction)
acc = accuracy_score(Y_test, prediction)
print("****Support Vector Machine**** ")
print("ACCURACY= ", accuracy_score(Y_test, prediction)*100)
print("Precision:", precision_score(Y_test, prediction)*100)
print("Recall:", recall_score(Y_test, prediction)*100)
print("Confusion Matrix :\n", cf_matrix)
cm = confusion_matrix(Y_test, prediction)
plt.figure(figsize=(12, 12))
sns.heatmap(cm, annot=True, fmt=".0f", linewidths=.5, square=True, cmap='Blues');
plt.ylabel('Actual label');
plt.xlabel('Predicted label');
all_sample_title = 'Confusion Matrix of SVM- missing data'
plt.title(all_sample_title, size=15);
plt.show()
```

Imputation:

```
import pandas as pd
import math
import time

"""A method to get the average value of any column, ignoring question mark values"""
def getColAve(object1):
    counter = 0
    runningSum = 0
    # For Each Row Item (i) in the passed column
    for i in object1:
        # that is not equal to '?'
        if (i != '?'):
            # Make the value a float
            i = float(i)
            # Increment counter
            counter += 1
            #Add to sum
            runningSum = runningSum + i
    toReturn = (runningSum / counter)
    toReturn = round(toReturn,5)
    return toReturn

"""Replace all '?' with the calculated average
Fully functioning right now!"""
def replaceWithAverage(theDF):
    aDF = theDF
    ave = 0
    # For Each Column in the Dataframe Object    do stuff
    for i in aDF:
        # For Each Column that is not 'Class'    do stuff
        if (i != 'Class'):
            ave = getColAve(aDF[i])
            aDF[i] = aDF[i].replace('?', value = str(ave), inplace=False)
```

```
return aDF
```

Accuracy Of imputed Data:

```
import numpy as np
```

```
import pandas as pd
```

```
from sklearn.model_selection import train_test_split
```

```
from sklearn.preprocessing import StandardScaler
```

```
from sklearn import svm
```

```
from sklearn.metrics import accuracy_score, precision_score, recall_score,  
confusion_matrix
```

```
import matplotlib.pyplot as plt
```

```
import seaborn as sns
```

```
model = svm.SVC(kernel='linear')
```

```
scaler = StandardScaler()
```

```
def trainData():
```

```
    parkinsons_data = pd.read_csv('nearest neighbors_missing05_k1.csv')
```

```
    X = parkinsons_data.drop(columns=['Class'], axis=1)
```

```
    Y = parkinsons_data['Class']
```

```
# displaying heatmap
```

```
    plt.show()
```

```
    X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=2)
```

```
    scaler.fit(X_train)
```

```
    X_train = scaler.transform(X_train)
```

```
    X_test = scaler.transform(X_test)
```

```
    model.fit(X_train, Y_train)
```

```
    prediction=model.predict(X_test)
```

```
    cf_matrix = confusion_matrix(Y_test, prediction)
```

```
    acc = accuracy_score(Y_test, prediction)
```

```
    print("*****Support Vector Machine***** ")
```

```
    print("ACCURACY= ", accuracy_score(Y_test, prediction)*100)
```

```
    print("Precision:", precision_score(Y_test, prediction)*100)
```

```
    print("Recall:", recall_score(Y_test, prediction)*100)
```

```
    print("Confusion Matrix :\n", cf_matrix)
```

```
    cm = confusion_matrix(Y_test, prediction)
```

```
    plt.figure(figsize=(12, 12))
```

```

sns.heatmap(cm, annot=True, fmt=".0f", linewidths=.5, square=True, cmap='Blues');
plt.ylabel('Actual label');
plt.xlabel('Predicted label');
all_sample_title = 'Confusion Matrix of SVM- imputes'
plt.title(all_sample_title, size=15);
plt.show()

```

Application:

```

from flask import Flask, render_template, request
app = Flask(__name__)
from test import testData
@app.route('/')
def index():
    return render_template('cta.html')
@app.route('/sub', methods=["POST"])
def submit():
    if request.method == 'POST':
        input_dict = request.form
        input_data = tuple(input_dict.values())
        result = testData(input_data)
        return render_template('sub.html', n=result)
if __name__ == '__main__':
    app.run(host="localhost", port=7000, debug=True)

```

B Screen Shoots: COMPLETE DATA SET

```
1 F1,F2,F3,F4,F5,F6,F7,F8,Class
2 0.16696,0.57025,0.59701,0.78662,0.70328,0.75069,0.50256,0.72300,Yes
3 0.12046,0.17261,0.23112,0.48582,0.31992,0.41143,0.50449,0.36652,Yes
4 0.15317,0.22508,0.45034,0.44776,0.41860,0.43565,0.50064,0.51709,Yes
5 0.12547,0.23807,0.22792,0.36509,0.31545,0.34341,0.45385,0.28373,Yes
6 0.09775,0.26784,0.25662,0.46689,0.35326,0.41629,0.33654,0.44718,Yes
7 0.27291,0.39732,0.49208,0.65373,0.54958,0.60795,0.72244,0.61170,Yes
8 0.16770,0.23896,0.33052,0.47661,0.36429,0.42663,0.56474,0.44978,Yes
9 0.22471,0.26738,0.36075,0.52305,0.37598,0.45734,0.58462,0.52894,Yes
10 0.24656,0.29614,0.34528,0.33944,0.27074,0.30900,0.39487,0.38815,Yes
11 0.42241,0.17424,0.45422,0.44032,0.34039,0.39590,0.26154,0.58286,Yes
12 0.20513,0.23381,0.36774,0.48509,0.38710,0.44167,0.52372,0.45778,Yes
13 0.09879,0.31071,0.31554,0.49542,0.36964,0.43935,0.59679,0.43673,Yes
14 0.15164,0.42890,0.44612,0.55228,0.49263,0.52653,0.43654,0.57538,Yes
15 0.22901,0.42855,0.41226,0.51487,0.46388,0.49298,0.65000,0.47319,Yes
16 1.00000,0.14935,0.81749,0.57585,0.53388,0.55823,0.33782,0.77120,Yes
17 0.41991,0.13609,0.43365,0.46542,0.37588,0.42580,0.43141,0.54205,Yes
18 0.25747,0.27879,0.43315,0.45800,0.35932,0.41425,0.37756,0.52457,Yes
19 0.22403,0.21475,0.32647,0.47170,0.37885,0.43059,0.49231,0.47537,Yes
20 0.12288,0.35482,0.26463,0.46639,0.37756,0.42710,0.42692,0.38327,Yes
21 0.36961,0.29403,0.52005,0.49718,0.48654,0.49365,0.52500,0.50393,Yes
22 0.36961,0.29403,0.52005,0.58616,0.49248,0.50127,0.52244,0.50393,Yes
23 0.17241,0.44005,0.43057,0.52192,0.46328,0.49655,0.39487,0.54519,Yes
24 0.13109,0.17424,0.22904,0.41607,0.31379,0.37051,0.36667,0.33163,Yes
25 0.32886,0.19646,0.44503,0.53261,0.44540,0.49424,0.39103,0.62263,Yes
26 0.28436,0.22096,0.41732,0.50265,0.37197,0.44436,0.46731,0.53248,Yes
27 0.29270,0.23049,0.42519,0.46117,0.36571,0.41885,0.29103,0.52702,Yes
28 0.18391,0.22758,0.32310,0.51948,0.40506,0.46866,0.43910,0.50554,Yes
29 0.25264,0.42991,0.49458,0.46955,0.45588,0.46456,0.40000,0.44758,Yes
30 0.22343,0.20736,0.32034,0.48565,0.36326,0.43110,0.48974,0.49631,Yes
```

```
8769 0.23146,0.17186,0.34312,0.46852,0.40596,0.44123,0.34423,0.50967,No
8770 0.25461,0.14321,0.32881,0.46542,0.39343,0.43381,0.34423,0.49370,No
8771 0.25461,0.05729,0.29118,0.43455,0.35026,0.39725,0.26987,0.47773,No
8772 0.26363,0.08897,0.39371,0.40176,0.35339,0.38076,0.30385,0.49812,No
8773 0.28760,0.08897,0.39371,0.43908,0.30757,0.41675,0.30385,0.51465,No
8774 0.26363,0.08897,0.33993,0.43232,0.34479,0.39354,0.25128,0.49812,No
8775 0.26363,0.08897,0.33993,0.43790,0.34671,0.39746,0.25128,0.49812,No
8776 0.31157,0.08897,0.42957,0.43672,0.37549,0.40995,0.30385,0.54772,No
8777 0.26363,0.08897,0.33993,0.43168,0.34466,0.39313,0.29231,0.48158,No
8778 0.12562,0.23319,0.21890,0.50749,0.40739,0.46317,0.39058,0.40554,No
8779 0.12363,0.20399,0.23586,0.38935,0.28951,0.34484,0.21410,0.35777,No
8780 0.26401,0.15682,0.35416,0.45909,0.35666,0.41358,0.38654,0.49300,No
8781 0.25461,0.11457,0.36944,0.38825,0.36428,0.37834,0.28269,0.50967,No
8782 0.23146,0.17186,0.32581,0.45723,0.39529,0.43019,0.34423,0.49370,No
8783 0.10023,0.10632,0.18466,0.54389,0.37649,0.46896,0.27372,0.34741,No
8784 0.20031,0.11457,0.29118,0.45495,0.38301,0.42334,0.30833,0.49370,No
8785 0.08761,0.09293,0.10037,0.45717,0.28211,0.37852,0.23654,0.36914,No
8786 0.14441,0.17871,0.23006,0.45408,0.37505,0.41923,0.27756,0.40103,No
8787 0.13928,0.19049,0.22369,0.49706,0.18171,0.44576,0.40128,0.43388,No
8788 0.33084,0.11497,0.36952,0.44088,0.34081,0.40552,0.26090,0.52240,No
8789 0.27457,0.18295,0.38576,0.38098,0.31160,0.35014,0.26731,0.51486,No
8790 0.22303,0.18400,0.10171,0.51588,0.41922,0.47315,0.58013,0.52768,No
8791 0.17786,0.25405,0.30571,0.55037,0.42731,0.49569,0.39744,0.54222,No
8792 0.14380,0.04449,0.11583,0.28646,0.16115,0.23005,0.18333,0.25009,No
8793 0.00000,0.34137,0.23712,0.36414,0.30969,0.34026,0.31282,0.47382,No
8794 0.20656,0.17382,0.24279,0.48380,0.36546,0.43110,0.39359,0.42159,No
8795 0.26845,0.21495,0.29377,0.58141,0.45106,0.52349,0.44038,0.51295,No
8796 0.12112,0.07494,0.10879,0.44614,0.28915,0.37570,0.25577,0.27054,No
```

MISSING DATASET-1

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help missing_imputation - dataset_missing05.csv
missing_imputation dataset_missing05.csv
a2.py nearest_neighbors_missing05_k1.csv nearest_neighbors_missing20_k2.csv dataset_complete.csv dataset_missing05.csv dataset_missing20.csv
Plugins supporting *.csv files found. Install plugins Ignore extension

1 F1,F2,F3,F4,F5,F6,F7,F8,Class
2 0.16696,0.57025,0.59701,0.78662,0.70328,0.75069,0.50256,0.72300,Yes
3 0.12046,0.17261,0.23112,0.48582,0.31992,0.41143,0.50449,0.36652,Yes
4 0.15317,0.22508,0.45034,?,0.41860,0.43565,0.50064,0.51709,Yes
5 ?,0.23807,0.22792,0.36509,0.31545,0.34341,0.45385,0.28373,Yes
6 0.09775,0.26784,0.25662,0.46689,0.35326,0.41629,0.33654,0.44718,Yes
7 0.27291,0.39732,0.49200,0.65373,0.54958,0.60795,?,?,Yes
8 0.16770,0.23896,0.33052,0.47661,?,0.42663,?,0.44978,Yes
9 ?,0.26738,0.36075,0.52305,?,0.45734,0.58462,0.52894,Yes
10 0.24656,0.29614,0.34528,0.33944,0.27074,0.30900,0.39487,0.38815,Yes
11 0.42241,0.17424,0.45422,0.44032,0.34039,0.39590,0.26154,0.58286,Yes
12 0.20513,0.23301,0.36774,0.48509,0.38710,0.44167,0.52372,0.45778,Yes
13 ?,0.31071,0.31554,0.49542,0.36964,0.43935,0.59679,0.43673,Yes
14 0.15164,?,0.44612,0.55228,0.49263,0.52653,0.43654,0.57538,Yes
15 0.22901,0.42855,0.41226,0.51487,0.46388,0.49298,0.65000,0.47319,Yes
16 1.00000,0.14935,0.81749,0.57585,0.53388,0.55823,0.33782,0.77120,Yes
17 0.41991,0.13609,0.43365,0.46542,0.37588,0.42580,0.43141,0.54205,Yes
18 0.25747,0.27879,0.43315,0.45800,0.35932,?,0.37756,0.52457,Yes
19 0.22403,0.21475,0.32647,0.47170,0.37885,0.43059,0.49231,0.47537,Yes
20 0.12288,0.35482,?,0.46639,0.37756,0.42710,0.42692,0.38327,Yes
21 0.36961,0.29403,0.52005,0.49718,0.48654,0.49365,0.52500,?,Yes
22 0.36961,0.29403,0.52005,0.50616,0.49248,0.50127,0.52244,0.50393,Yes
23 0.17241,0.44805,?,0.52192,0.46328,0.49655,0.39487,0.54519,Yes
24 0.13109,0.17424,0.22904,0.41607,0.31379,0.37051,0.36667,0.33163,Yes
25 0.32886,0.19646,0.44503,0.53261,0.44540,0.49424,0.39103,0.62263,Yes
26 0.28436,0.22096,0.41732,0.50265,0.37197,0.44436,0.46731,0.53248,Yes
27 0.29270,0.23049,0.42519,0.46117,0.36571,0.41885,0.29103,0.52702,Yes
28 0.18391,0.22758,0.32310,0.51948,0.40506,0.46866,?,0.50554,Yes
29 0.25266,0.42991,0.49458,0.46955,0.45588,0.46456,0.40000,0.44758,Yes
30 0.22343,0.20736,0.32034,0.48565,0.36326,0.43110,0.48974,0.49631,Yes

TODO Problems Terminal Python Packages Python Console
4043:1 LF UTF-8 4 spaces Python 3.9 (missing_imputation)
23:12 02-03-2022
```

AFTER MISSING DATA IMPUTATION-1

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help missing_imputation - nearest_neighbors_missing05_k1.csv
missing_imputation nearest_neighbors_missing05_k1.csv
a2.py nearest_neighbors_missing05_k1.csv nearest_neighbors_missing20_k2.csv dataset_complete.csv dataset_missing05.csv dataset_missing20.csv
Plugins supporting *.csv files found. Install plugins Ignore extension

1 F1,F2,F3,F4,F5,F6,F7,F8,Class
2 0.16696,0.57025,0.59701,0.78662,0.70328,0.75069,0.50256,0.72300,Yes
3 0.12046,0.17261,0.23112,0.48582,0.31992,0.41143,0.50449,0.36652,Yes
4 0.15317,0.22508,0.45034,0.50064,0.41860,0.43565,0.50064,0.51709,Yes
5 0.27748,0.23807,0.22792,0.36509,0.31545,0.34341,0.45385,0.28373,Yes
6 0.09775,0.26784,0.25662,0.46689,0.35326,0.41629,0.33654,0.44718,Yes
7 0.27291,0.39732,0.49200,0.65373,0.54958,0.60795,0.44482,0.52851,Yes
8 0.16770,0.23896,0.33052,0.47661,0.44166,0.42663,0.44482,0.44978,Yes
9 0.27748,0.26738,0.36075,0.52305,0.44166,0.45734,0.58462,0.52894,Yes
10 0.24656,0.29614,0.34528,0.33944,0.27074,0.30900,0.39487,0.38815,Yes
11 0.42241,0.17424,0.45422,0.44032,0.34039,0.39590,0.26154,0.58286,Yes
12 0.20513,0.23301,0.36774,0.48509,0.38710,0.44167,0.52372,0.45778,Yes
13 0.27748,0.31071,0.31554,0.49542,0.36964,0.43935,0.59679,0.43673,Yes
14 0.15164,0.29188,0.44612,0.55228,0.49263,0.52653,0.43654,0.57538,Yes
15 0.22901,0.42855,0.41226,0.51487,0.46388,0.49298,0.65000,0.47319,Yes
16 1.00000,0.14935,0.81749,0.57585,0.53388,0.55823,0.33782,0.77120,Yes
17 0.41991,0.13609,0.43365,0.46542,0.37588,0.42580,0.43141,0.54205,Yes
18 0.25747,0.27879,0.43315,0.45800,0.35932,0.47964,0.37756,0.52457,Yes
19 0.22403,0.21475,0.32647,0.47170,0.37885,0.43059,0.49231,0.47537,Yes
20 0.12288,0.35482,0.42874,0.46639,0.37756,0.42710,0.42692,0.38327,Yes
21 0.36961,0.29403,0.52005,0.49718,0.48654,0.49365,0.52500,0.52851,Yes
22 0.36961,0.29403,0.52005,0.50616,0.49248,0.50127,0.52244,0.50393,Yes
23 0.17241,0.44805,0.42874,0.52192,0.46328,0.49655,0.39487,0.54519,Yes
24 0.13109,0.17424,0.22904,0.41607,0.31379,0.37051,0.36667,0.33163,Yes
25 0.32886,0.19646,0.44503,0.53261,0.44540,0.49424,0.39103,0.62263,Yes
26 0.28436,0.22096,0.41732,0.50265,0.37197,0.44436,0.46731,0.53248,Yes
27 0.29270,0.23049,0.42519,0.46117,0.36571,0.41885,0.29103,0.52702,Yes
28 0.18391,0.22758,0.32310,0.51948,0.40506,0.46866,0.44482,0.50554,Yes
29 0.25266,0.42991,0.49458,0.46955,0.45588,0.46456,0.40000,0.44758,Yes
30 0.22343,0.20736,0.32034,0.48565,0.36326,0.43110,0.48974,0.49631,Yes

TODO Problems Terminal Python Packages Python Console
1:1 CRLF UTF-8 4 spaces Python 3.9 (missing_imputation)
23:12 02-03-2022
```

MISSING DATA-2

The screenshot shows an IDE window titled "missing_imputation - dataset_mi". The active file is "dataset_missing20.csv". The code editor displays a dataset with 30 rows and 9 columns: F1, F2, F3, F4, F5, F6, F7, F8, and Class. The data contains several missing values, represented by question marks (?). For example, in the first row, F1 is 0.16696, F2 is ?, F3 is ?, F4 is ?, F5 is 0.75069, F6 is 0.50256, F7 is 0.72300, F8 is Yes, and Class is Yes. The IDE interface includes a menu bar (File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help), a toolbar with icons for file operations, a sidebar with "Project", "Structure", and "Favorites" views, and a bottom status bar with "TODO", "Problems", "Terminal", "Python Packages", and "Python Console" tabs.

```
1 F1,F2,F3,F4,F5,F6,F7,F8,Class
2 0.16696,?,?,?,0.75069,0.50256,0.72300,Yes
3 0.12046,0.17261,0.23112,0.48582,0.31992,?,0.36652,Yes
4 0.15317,0.22508,0.45034,0.44776,0.41860,0.43565,0.50064,0.51709,Yes
5 0.12547,?,0.22792,0.36509,0.31545,0.34341,?,?,Yes
6 ?,0.26784,0.25662,0.46689,?,0.41629,?,0.44718,Yes
7 ?,0.39732,0.49200,?,0.60795,0.72244,0.61170,Yes
8 0.16770,0.23896,0.33052,0.47661,0.36429,0.42663,0.56474,0.44978,Yes
9 0.22471,0.26738,?,?,0.45734,0.58462,0.52894,Yes
10 0.24656,0.29614,0.34528,?,0.27074,0.30900,0.39487,0.38815,Yes
11 0.42241,0.17424,?,0.44032,0.34039,0.39590,0.26154,0.58286,Yes
12 0.20513,0.23381,?,0.48509,0.38710,?,0.52372,?,Yes
13 0.09879,0.31071,?,?,0.43935,0.59679,0.43673,Yes
14 ?,0.42890,0.44612,0.55228,?,0.52653,0.43654,0.57538,Yes
15 ?,0.41226,0.51487,0.46388,0.49298,0.65000,0.47319,Yes
16 1.00000,0.14935,0.81749,0.57585,0.53388,?,0.33782,0.77120,Yes
17 ?,0.13609,0.43365,?,0.37588,0.42580,0.43141,0.54205,Yes
18 0.25747,0.27879,0.43315,0.45800,0.35932,0.41425,0.37756,0.52457,Yes
19 0.22403,?,0.47170,?,0.43059,?,0.47537,Yes
20 0.12288,?,0.26463,0.46639,0.37756,0.42710,0.42692,?,Yes
21 0.36961,0.29403,0.52005,0.49718,0.48654,0.49365,0.52500,0.50393,Yes
22 0.36961,0.29403,?,0.50616,?,0.50127,0.52244,?,Yes
23 0.17241,0.44805,0.43057,0.52192,0.46328,0.49655,0.39487,0.54519,Yes
24 0.13109,0.17424,?,0.41607,0.31379,0.37051,0.36667,0.33163,Yes
25 0.32886,0.19646,0.44503,0.53261,0.44540,0.49424,?,0.62263,Yes
26 ?,0.41732,0.50265,0.37197,0.44436,?,0.53248,Yes
27 0.29270,0.23049,0.42519,0.46117,0.36571,?,0.29103,0.52702,Yes
28 0.18391,0.22758,0.32310,0.51948,0.40506,0.46866,0.43910,?,Yes
29 ?,0.42991,?,0.46955,0.45588,0.46456,?,?,Yes
30 0.22343,0.20736,0.32034,0.48565,?,0.43110,0.48974,0.49631,Yes
```

AFTER MISSING DATA IMPUATATION-2

The screenshot shows the same IDE window, but now the active file is "nearest_neighbors_missing05.csv". The data has been imputed, and all missing values have been replaced with numerical values. For example, in the first row, F1 is 0.16696, F2 is 0.57025, F3 is 0.59701, F4 is 0.78662, F5 is 0.70328, F6 is 0.75069, F7 is 0.50256, F8 is 0.72300, and Class is Yes. The IDE interface is identical to the previous screenshot, showing the same menu bar, toolbar, sidebar, and status bar.

```
1 F1,F2,F3,F4,F5,F6,F7,F8,Class
2 0.16696,0.57025,0.59701,0.78662,0.70328,0.75069,0.50256,0.72300,Yes
3 0.12046,0.17261,0.23112,0.48582,0.31992,0.41143,0.50449,0.36652,Yes
4 0.15317,0.22508,0.45034,0.47895,0.41860,0.43565,0.50064,0.51709,Yes
5 ?,0.23807,0.22792,0.36509,0.31545,0.34341,0.45385,0.28373,Yes
6 0.09775,0.26784,0.25662,0.46689,0.35326,0.41629,0.33654,0.44718,Yes
7 0.27291,0.39732,0.49200,0.65373,0.54958,0.60795,0.51282,0.62567,Yes
8 0.16770,0.23896,0.33052,0.47661,0.37770,0.42663,0.54295,0.44978,Yes
9 0.23810,0.26738,0.36075,0.52305,?,0.45734,0.58462,0.52894,Yes
10 0.24656,0.29614,0.34528,0.33944,0.27074,0.30900,0.39487,0.38815,Yes
11 0.42241,0.17424,0.45422,0.44032,0.34039,0.39590,0.26154,0.58286,Yes
12 0.20513,0.23381,0.36774,0.48509,0.38710,0.44167,0.52372,0.45778,Yes
13 0.12594,0.31071,0.31554,0.49542,0.36964,0.43935,0.59679,0.43673,Yes
14 0.15164,0.44725,0.44612,0.55228,0.49263,0.52653,0.43654,0.57538,Yes
15 0.22901,0.42855,0.41226,0.51487,0.46388,0.49298,0.65000,0.47319,Yes
16 1.00000,0.14935,0.81749,0.57585,0.53388,0.55823,0.33782,0.77120,Yes
17 0.41991,0.13609,0.43365,0.46542,0.37588,0.42580,0.43141,0.54205,Yes
18 0.25747,0.27879,0.43315,0.45800,0.35932,0.42305,0.37756,0.52457,Yes
19 0.22403,0.21475,0.32647,0.47170,0.37885,0.43059,0.49231,0.47537,Yes
20 0.12288,0.35482,0.27387,0.46639,0.37756,0.42710,0.42692,0.38327,Yes
21 0.36961,0.29403,0.52005,0.49718,0.48654,0.49365,0.52500,0.50393,Yes
22 0.36961,0.29403,0.52005,0.50616,0.49248,0.50127,0.52244,0.50393,Yes
23 0.17241,0.44805,0.42775,0.52192,0.46328,0.49655,0.39487,0.54519,Yes
24 0.13109,0.17424,0.22904,0.41607,0.31379,0.37051,0.36667,0.33163,Yes
25 0.32886,0.19646,0.44503,0.53261,0.44540,0.49424,0.39103,0.62263,Yes
26 0.28436,0.22096,0.41732,0.50265,0.37197,0.44436,0.46731,0.53248,Yes
27 0.29270,0.23049,0.42519,0.46117,0.36571,0.41885,0.29103,0.52702,Yes
28 0.18391,0.22758,0.32310,0.51948,0.40506,0.46866,0.43077,0.50554,Yes
29 0.25266,0.42991,0.49458,0.46955,0.45588,0.46456,0.40000,0.44758,Yes
30 0.22343,0.20736,0.32034,0.48565,0.36326,0.43110,0.48974,0.49631,Yes
```

C Plagiarism Report:

An optimization-based approach to missing data imputation

Name: Pragati
Student: 38110416
Department of CSE
Sathyabama Institute of
science and technology
Maharashtra, India
Email Id:
pragatibhasawar@gmail.com

Name: Pranathi Yajjala
Student: 38110418
Department of CSE
Sathyabama Institute of science
and technology
Vishakapatnam, India
Email id:
pranathiyajjala2000@gmail.com

Name: Usha Nandini
Student: 38110416
Assistant professor
Sathyabama Institute of
Science and Technology
Chennai, India
Email Id:
pragatibhasawar@gmail.com

ABSTRACT

AI has forever been significant in getting and deciphering data, and it regularly assists with taking care of fragmented issues. These lacks in the informational collection might think twice about the nature of the preparation. This stresses the significance of a solid system to address lost worth (information). In this article, we investigate the distinction between the exhibition of an action on the planet and the repeat of deficient data. In view of this, we propose a machine-based EM-based characterization framework to make up for the lost worth of each lost property by utilizing it in light of future and anticipated data and to supplant the missing worth. Volunteer abilities (1). Pause and interface (Bertsekas, 1999). 1.AI (1) in BCD. Contrasts BETWEEN SVM PROCEDURES AND TREE STUDIES (3) 2.5 Use of K-NN or SVM (2) Appendix A Size - 5. It should likewise be 0. Here's the reason, uwv (2). little world (1), Moderate 3, Policy 3 New-first book There are different ways to deal with this deficient data book. Accordingly, we join writing with fragmented material and spotlight on AI abilities.

KEYWORDS :

Missing value implementation classification for the training dataset, dataset,
and algorithm used for handling missing values.

INTRODUCTION

Absence of data esteem is normal to all parts of data and results from an assortment of worries, for example, execution debasement, media investigation difficulties and uneven arrangements because of deficient and full worth contrasts. Missing issues might be a typical issue for machine administrators while investigating continuous information. Data sets are frequently fragmented and mistaken. Extra data in light of first experience will assist you with getting ready when the investigation. A similar strategy can be utilized to address deficient factors, for example, graphic, obsolete, neighbor-K, and relative examination. There are numerous measurable models and AI calculations that rely upon complete data, so it is essential to oversee fragmented data accurately. Exceptionally controlled learning methods depend on composed

insights and disregard the way that plain data can be utilized to foresee missing qualities. In any case, it is essential to recollect that the main conceivable arrangement is one in light of plan and cautious investigation [20]. This is on the grounds that presentation examination depends on various factors, for example, the sort of calculation utilized, the idea of the determination, and the displaying strategy.

LITERATURE SURVEY

Missing data, supervised learning, and machine learning have all been studied in-depth using a variety of scholarly publications. Existing functioning systems were understood in several articles that provided knowledge similar to the rationale

for this effort. While the topic of missing data has been addressed in statistical research, the majority of this work has been focused on the statistical analysis of data with missing values, as outlined in 1977 by **Dempster, Laird, and Rubin**. The algorithm, for example, uses maximum-likelihood approaches. The problem with this approach is that it takes a long time to put into practice. Meaningless values are substituted for each other in dimension d using the mean of all observed values, as stated by **Little and Rubin** in 1987. A benefit of the mean imputation is that it overestimates the variance and overlooks the connection between characteristics. However, it suffers from poor imputation. An overview of statistical analysis and imputation for modal data was first presented by **R. Ubin** in 1987. Since this strategy is still under investigation, we may compare its accuracy with the ML algorithms previously discussed. According to **Ghahramani and Jordan** in 1994, multivariate density functions are generally inadequate for describing the complexity of mixed data types. When categorical variables are independent, the benefit is clear. Using Gaussians and multinomials for imputation has a drawback. **Zhang** and colleagues published their work in 2006 and 2008 as "Low Dimensional representation-based imputation." Because of this, an SVD analysis of the full dataset is performed to find the primary eigenvectors in the basic technique.

Burgette and Reiter. The benefit is that random-forest extensions have demonstrated good outcomes as well. However, this technique has a drawback in that it uses random sampling and consecutive imputations until convergence. Using random sampling, **Buuren and Groothuis-Oudshoorn** developed a chained equation approach for generalizing to multivariate environments in 2011. The benefit of successively running univariate imputations until convergence is that it is

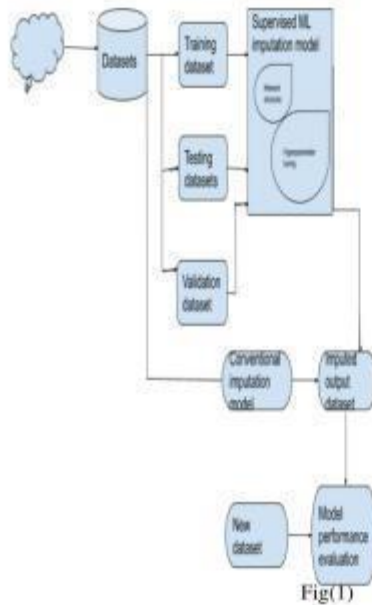
more efficient. The fact that it operates slowly is a drawback. They presented these non-parametric techniques for imputation based on decision trees in 2012, saying that they do not depend on distributional assumptions about the data. This technique has a significant benefit. The challenge is that statistical models and optimization difficulties are tightly linked in this case.

According to **Bertsimas, Van Parys, and Mazumder**, recent advancements in optimization have led to considerable development in machine learning between 2014 and 2017. With this, the benefit is that effective implementations of integer and convex optimization may be claimed. The drawback is that it has issues with median and sparse regression. As a result, the missing data issue is revisited in this study. Using optimization-based approaches for imputation, we can increase our imputation systems' performance outside of the sample. Cons: It's an explicit optimization problem formulation.

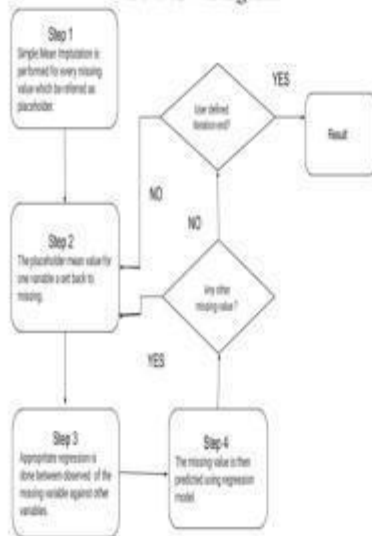
PROPOSED WORK AND DESIGN

The venture is relied upon to foster a Python-based mining framework that can gauge HD utilizing a tree. This procedure utilized 14 things from UCI. There are many of the clarifications given (not yet known as hyperparameters) and it is normal however hard to anticipate those defects. A basic substitute will settle on a choice in view of the HD screen information, which will unavoidably diminish the presentation of the machine. Accordingly, the arranged exploration was utilized to underrate the worth of trustworthiness. We compute the worth of each missing thing by distinguishing the worth of the missing thing. It is done in investigate. The updated statistics was utilized to supplant inadequate qualities from 1.98 percent to 50.65 percent in medical clinic data.

A. BLOCK DAIGRAM



B. Flow Diagram



Fig(2)

METHODOLOGY USED

In tackling the issue (1), we can rapidly utilize the principal innovation to begin giving the best outcomes. We will zero in on impairing the BCD) and handicapping (CD) calculations (Bertsekas, 1999). In Problem 1, we use Algorithm 1 (likewise called select. Credit), which involves BCD or CD in Problem (1). Table 2 sums up the qualities of the capacity $c(x)$ for the U, W, V, X changes and the K-NN, SVM, and calculation learning tree. Nitty gritty solutions to questions (2), (3), (4), (5) utilizing K-NN, SVM, or enlistment are given in 2.3-2.5. See Annex A for more data. Utilizing the chart, decrease the worth of the objective capacity to no less than 0 until the assignment is finished. Consequently, the quantity of steps needed to finish the calculation d 1 0 c (U0, W0, V0; X) e, W0 and V0 address the beginning stage, X is the information, and U0 is the contention to Equation 1. (2) However, there is noassurance that the calculation will open essentially the entire world (1). (Wright, 2015). In the accompanying segments, we will consider three models and three methods for tackling the issue. We give explicit data about each model and every one of the underlying phases of the association between them. Specifically, in this review, we utilized numbers that were known to every one of the sacred writings, and fostered a technique for losing data from 10% to half of the absolute informational collection. Truth comprises of all the data that makes a X without losing anything. We utilize the most well-known and exceptional calculations to ascertain this data to foresee what's to come..

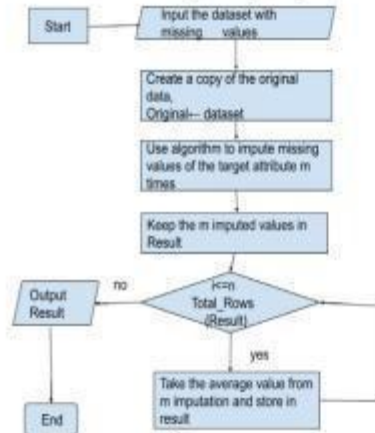
ALGORITHMS:

K-Nearest Neighbor (KNN) classification method:

The article describes a single-step greedy strategy for imputation missing values that uses an observation's K-nearest neighbours based on Euclidean distance to impute missing values.

In the absence of any additional coordinates, the averaged distance is used. The answer was developed using the impute package in R. Using K-NN, a model sort that depends on learning, or sluggish realization, the calculation is slow and slow until it is accomplished. The k-NN

calculation is the easiest of all the artificial intelligence calculations to carry out successfully. No neighbours were considered when categorizing things (for k-



NN classifications) or assessing property values (for k-NN withdrawals).

STAGE 1: start

STAGE 2: $D = (x_1, c_1), \dots, (x_N, c_N)$ input

STAGE 3: The quantity of new cases to be carried out is $x = (x_1 \dots x_n)$.

STAGE 4: Determine the worth of d for each case in which it is composed (x_i, c_i) .

(x_i, x) .

STAGE 5: $d(x_i, x)$ I = rising request

1 ... H).

STAGE 6: Select the K issue close to x : D_{kx} .

STAGE 7: Score high

From d_{kx} to x .

STAGE 8: Conclusion

To track down the best answer for the issue (1), we can utilize the primary technique to order abrupt temperatures. Specifically, we will zero in on disturbance coordination (BCD) and interruption coordination (CD) (Bertsekas, 1999). Calculation 1, called

opt.impute, carries out BCD or CD in issue (1). The factors U, W, V, X and the worth capacity $c(\cdot)$ are summed up in table 2 for K-NN, SVM, and trees. Infinite techniques for taking care of issues (2), (3), (4) and (5) of K-NN, SVM and tree including are portrayed in 2.3-2.5.

In a helpful manner, the worth of viable activities diminishes by something like kugeza0 as far as possible. It tends to be seen that the quantity of advances expected to finish the calculation is $d \ 1 \ 60 \ c(U_0, W_0, V_0: X) \ c, W_0, V_0$ is the main worth, X is the information, and U_0 is the argmin in equation (2). In any case, the calculation isn't permitted to track down the littlest size on the planet (1) (Wright, 2015). For each model and the primary request of sorts, we make novel advancements

C. FLOW DAIGRAM

Fig(3)

Thoughts on how M data is created that isn't on the X informational collection. The primary concentration for this situation is inadequate plan X^{miss}, X^{obs} forthcoming and fragmented data.

Table 1.1

Mechanism of Missing Data	Assumption
Missing Completely at Random (MCAR)	$f(M X^{obs}, X^{miss}) = f(M)$
Missing at Random (MAR)	$f(M X^{obs}, X^{miss}) = f(M X^{obs})$
Not Missing at Random (NMAR)	$f(M X^{obs}, X^{miss}) \text{ is } X^{miss}$

Input:

$$X \in \mathbb{R}^{n \times p_0} \times \{1, \dots, k_1\} \times \dots \times \{1, \dots, k_{p_1}\},$$

a data matrix with some missing entries

$M = \{(i, d): x_{id} \text{ is missing}\}, \delta_0 > 0 \text{ and}$
Warm start $W^0 \in \mathbb{R}^{n \times p_0}$,
 $V^0 \in \{1, \dots, k_1\} \times \dots \times \{1, \dots, k_{p_1}\}$.
Output:
 X^{imp} a full matrix with imputed values.
Procedure :
Initialize $\delta \leftarrow \infty, W \leftarrow W^0, V^{old} \leftarrow V^0$.
while $\delta > \delta_0$ **do**.
 1. Update U^* **using model dependent information:**
 $U^* \leftarrow \arg \min_U c(U, W^{old}, V^{old}; X)$
 s. t. $(U, W^{old}, V^{old}) \in U$
 2. Update the imputation W^*, V^* , **following either:**
 $w^*, v^* \leftarrow \arg \min_{w, v} c(U^*, W, V; X)$
 s. t. $w_{id} = x_{id} \quad (i, d) \in N_0$
 $v_{id} = x_{id} \quad (i, d) \in N_1$
 $(U^*, W, V) \in U$,
 3. $\delta \leftarrow c(U^*, W^*, V^*; X) - c(U^{old}, W^{old}, V^{old}; X)$,
 Type equation here.
 4. $(U^{old}, W^{old}, V^{old}) \leftarrow (U^*, W^*, V^*)$
 end while
 $X^{imp} \leftarrow [W^*; V^*]$
 we use in our optimization-based imputation approach for the variables U , W , **and** V

We first define a distance metric between rows (w_i, v_i) and (w_j, v_j) as

$$d_{ij} = \sum_{d=p_0+1}^{p_0+p_1} (w_{id} - w_{jd})^2 + \sum_{d=p_0+1}^{p_0+p_1} (v_{id} - v_{jd})^2$$

- The Optimization problem for the K-NN Based imputation model is:

$$Z \in \{0, 1\}^{id \times a}$$

MODULES

- Dataset collection
- Data Cleaning
- Feature Extraction
- Model training
- Testing model
- Performance Evaluation
- Prediction

Dataset collection

Information assortment can assist you with tracking down ways of following previous occasions utilizing information examination. To do this, you use web crawlers to make prescient formats and change what's to come. In this module, we gather data from the chronicles of the kaggle data set. These figures contain data on yearly contrasts.

Data Cleaning

Information cleaning is a significant stage in all AI exercises. It has a ton of abilities in measurable investigation and information recovery, so you can utilize it to look for data to discover what information handling exercises you can do.

Feature Extraction:

In AI, picture acknowledgment, and picture handling, mining starts at the front of estimated, useful data (credits) pointed toward guaranteeing, adjusting, following, and normalizing data, and now and again prompting greater lucidity. Wipe out the properties related with aspect decrease.

Model training:

An illustration of this preparing is the informational collection used to prepare the ML calculation. It comprises of applicable info definitions that influence information testing and result. Plan language preparing

is a method for giving data about the ML calculation and assist with deciding and get familiar with the best importance of every one of its highlights.

Testing model:

Programming testing is fit for distinguishing formative blunders and oversights. You don't need your clients to have issues after the product is delivered. Various sorts of tests permit just apparent blunders to happen during activity.

Performance Evaluation :

Execution examinations are characterized as guidelines and successful proportions of representative execution and execution, contingent upon the worker's liabilities. It is utilized to quantify representative worth included request to increment business income contrasted with industry norms and gross profit from venture (ROI).

Prediction:

The calculation gives the potential qualities of the obscure factors for each new information archive and permits the creator to know what they mean.

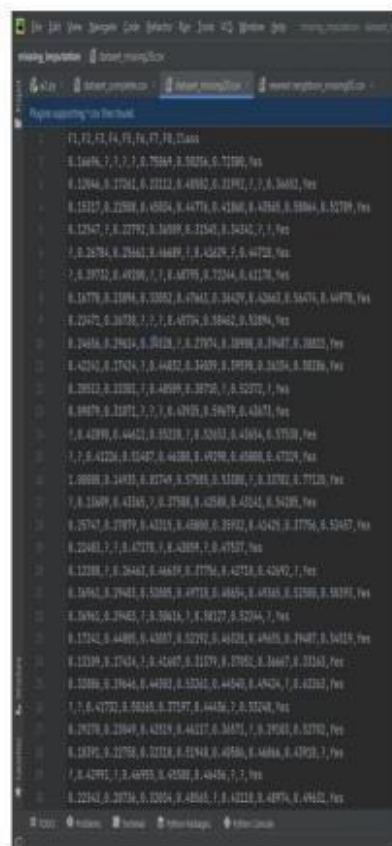
In this module, we utilize a prepared and created AI technique to decide whether the patient is posing similar inquiries and posing similar inquiries.

RESULTS

Here are a few ideas on the most proficient method to look or get an arrangement for tattoos. At times, the data given is utilized as a reason for exhibiting AI or data handling innovation. In different cases, it is utilized as a reason for looking at changed strategies. We utilized the technique on 84 informational indexes in the UCI Machine Learning data set. The size of this information is given from $n = 23$ to 5,875, with $p = 2$ to 124. In the accompanying areas, we first show the genuineness of everything. The fast and for the most part substitution technique prompts a decrease in MAE. Presently we show that the nature of conviction is extremely high as far as decision. Contrasted with the techniques utilized, it prompts further developed

execution and lower back. We will keep on examining the outlines (K, cp, C), start temperature, how to fall (BCD or CD), the arrangement of the information, including the nature of the self-appraisal work contingent upon the missing technique. At long last, we think about the computational heaps of every technique. The pictures underneath shows the outcomes.

MISSING DATA SET:



Fig(4)

An optimization (2).docx

ORIGINALITY REPORT

3 %	0 %	0 %	3 %
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to University of Oklahoma	3 %
	Student Paper	
2	www.testmagazine.biz	<1 %
	Internet Source	

Exclude quotes On
Exclude bibliography On

Exclude matches Off