MISSING DATA IMPUTATION WITH DENOISING AUTOENCODER USING MIDAS ON REAL LIFE DATASET

Ibrahim A. I. Hameed[1]

[1]*M. Tech Computer Science and Technology, Department of Computer Science and System, AU College of Engineering (A) Andhra University.*

---***---

Abstract - Missing information is substantial trouble impacting all domains. State-of-the-artwork framework for minimizing lacking information bias is more than one imputation, for which the selection of an imputation version stays non- trivial. We introduce more than one imputation version primarily based totally on whole deep denoising autoencoders. To impute missing values (MVs), a regular concept is to discover the correlations among the attributes of the information. However, the correlations are normally complicated and accordingly hard to identify. Accordingly, a brand-new deep mastering version is known as Missing Data Imputation denoising Autoencoder (MIDIAS) that efficiently imputes the MVs in a given dataset with the aid of using exploring non-linear correlations among lacking values and non-locking values. Additionally, with the aid of using thinking about numerous information lacking patterns. We are trying out to introduce new version to test the accuracy of making use of MIDAS on actual real life dataset, we used financial institution dataset which classifies whether a customer will subscribe or no, for trying out the accuracy of MIDAS we divided the information set into 30% and 70% dataset and delete 2%,6% and 10% of information on every occasion and used MLP, and Voting classifiers to test the accuracy with the aid of using evaluating the outcomes with the accuracy of MLP, and Voting Classifiers of 30% dataset.

Key Words: MIDAS, MVs, Autoencoder.

1. INTRODUCTION

Missing data is an important issue, even small proportions of missing data can adversely impact the quality of learning process, leading to biased inference .[1,2] Many methods have been proposed over the past decades to minimize missing data bias .[2,3] and can be divided into two categories: One that attempt to model the missing data process and use all available partial data for directly estimating model parameters and two that attempt to fill in/impute missing values with plausible predicted values. Imputation methods are preferred for their obvious advantage, that is, providing users with a complete dataset that can be analyzed using user specified models.[4]
Along with the advances in computer hardware, ever-increasing computing power, and many promising real-life applications, deep learning and neural networks (NN) have received tremendous attention in recent years. Among the various well-knownNN models, AutoEncoder (AE) first encodes a data record into a low-dimensional latent vector which in turn is decoded back to the original data record in order to embed the inherent properties and the (usually non-linear) correlations amongst attributes into a latent vector. More specifically, by training the network to minimize a distortion measure between inputs and outputs (both are the input data record itself), an AE aims to learn a low-dimensional latent vector (called embedding) of the data record, to obtain a property-preserving representation of the raw input data record. Furthermore, to achieve a good embedding of the data for handling the issues of noisy data and data sparsity, denoising AutoEncoder (dAE) .[5,6]

Machine learning is a subfield of artificial intelligence (AI). The purpose of machine learning typically is to recognize the shape of data and in shape that data into models that may be understood and used by people.

An artificial neural network (ANN) is the piece of a computing system designed to simulate the manner the human mind analyzes and processes information. It is the muse of artificial intelligence (AI) and solves problems that might show not possible or difficult by human or statistical standards. ANNs have self-learning capabilities that permit them to provide higher outcomes as extra data turns into available.

Data cleaning, also called data cleansing or scrubbing, deals with detecting and removing errors and inconsistencies from data in order to improve the quality of data. Data quality problems are present in single data collections, such as files and databases, e.g., due to misspellings during data entry, missing information or other invalid data. When multiple data sources need to be integrated, e.g., in data warehouses, federated database systems or global web-based information systems, the need for data cleaning increases significantly. This is because the sources often contain redundant data in different representations. In order to provide access to accurate and consistent data, consolidation of different data representations and elimination of duplicate information become necessary.[14]

2. METHODOLOGY

This section introduce how we impute missing values with denoising autoencoder using MIDAS on Bank dataset, also using MLP, and Voting Classifiers for accuracy test.

MIDAS Algorithm:

The algorithm we have developed to implement MIDAS takes an incomplete dataset D as its input and returns M completed datasets.
The algorithm proceeds in three stages, each comprising a number of smaller steps. In the first stage, the input data D are prepared for Shaded blocks represent data points (shades denote different variables); crosses indicate missing values.

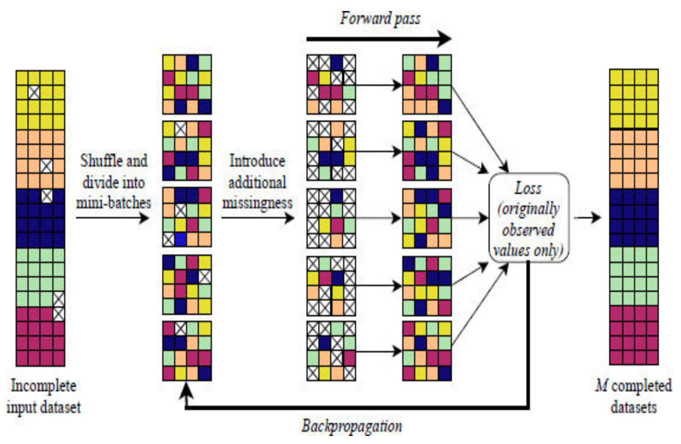

Forward pass
Shuffle and divide into mini-batches
Introduce additional missingness
Loss (originally observed values only)
Incomplete input dataset
M completed datasets
Backpropagation

Figure 1: Schematic of MIDAS Training Steps.

Training Categorical variables are “one-hot” encoded (i.e., converted into separate dummy variables for each unique class) and continuous variables are rescaled between 0 and 1 to improve convergence. In addition, a missingness indicator matrix R is constructed for D, allowing us to later distinguish between Dmis and Dobs, and all elements of Dmis are set to 0. A DA is then initialized according to the dimensions of D; the default architecture is a three-layer network with 256 nodes per layer.[18]

In the training stage, the following five steps are repeated (see Figure 1 for a visual schematic and Online Appendix 2B for a more formal description): (1) D and R are shuffled and sliced row wise into paired mini-batches (B1, B2, ..., Bn) to accelerate convergence; (2) mini-batch inputs are partially corrupted through multiplication by a Bernoulli vector v (default p = 0.8); (3) in line with standard implementations of dropout, outputs from half of the nodes in hidden layers are corrupted using the same procedure; (4) a forward pass through the DA is conducted and the reconstruction error on predictions of ˜xobs is calculated using the loss functions defined in Equation 7; and (5) loss values are aggregated into a single term and backpropagated through the DA, with the resulting error gradients used to adjust weights for the next epoch.[18]
Finally, once training is complete, the whole of D is passed into the DA, which attempts to reconstruct all (i.e., originally observed and originally missing) corrupted values. A completed dataset is then constructed by replacing Dmis with predictions of the originally missing values from the network’s output. This stage is repeated M times.[18]

MLP Classifier:

A multilayer perceptron can be used to perform classification. Consider the network shown in Figure 2. Each node in the network performs a simple function as shown in Figure 3. The input to this network is the feature vector extracted from the object to be classified, and the output is typically a block code where one output is high indicating the class of the object and all other outputs are low. The weights connecting the nodes are determined using some training rule with a set of feature vectors, the training set. The network shown uses two hidden layers. Cybenko has shown that at most one hidden layer is required for approximating functions. However, experience has shown that a two hidden layer network will train more quickly than a one hidden layer network on some classification problems. The number of layers used is problem dependent, as is the number of nodes in each hidden layer.[21]

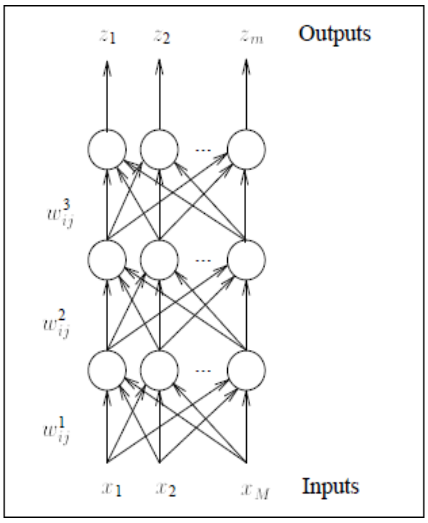


Figure 2: Multilayer Perceptron.

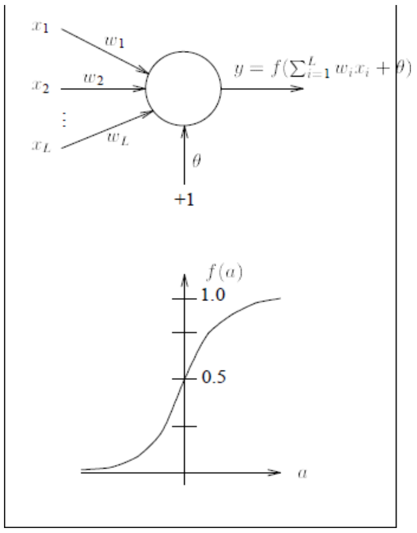


Figure 3:Function of Network Node.

Tensorflow:

TensorFlow Federated (TFF) is an open-supply framework for device gaining knowledge of and different computations on decentralized statistics. TFF has been evolved to facilitate open studies and experimentation with Federated Learning (FL), a method to device gaining knowledge of in which a shared worldwide version is educated throughout many taking part customers that hold their schooling statistics locally. For example, FL has been used to educate prediction model for cellular keyboards without importing touchy typing statistics to servers.

DataSet:

The data is related with direct marketing campaigns of a Portuguese banking institution. The marketing campaigns were based on phone calls. Often, more than one contact to the same client was required, in order to access if the product (bank term deposit) would be ('yes') or not ('no') subscribed. The classification goal is to predict if the client will subscribe (yes/no) a term deposit (variable y). [28]

Dataset attribute information:

input variables:
# bank client data:
1 - age (numeric)
2 - job : type of job (categorical: 'admin.','blue-collar','entrepreneur','housemaid','management','retired','self-employed','services','student','technician','unemployed','unknown')
3 - marital : marital status (categorical: 'divorced','married','single','unknown'; note: 'divorced' means divorced or widowed)
4 - education (categorical: 'basic.4y','basic.6y','basic.9y','high.school','illiterate','professional.course','university.degree','unkn own')
5 - default: has credit in default? (categorical: 'no','yes','unknown')

6 - housing: has housing loan? (categorical: 'no','yes','unknown') 7 - loan: has personal loan? (categorical: 'no','yes','unknown')
# related with the last contact of the current campaign:
8 - contact: contact communication type (categorical: 'cellular','telephone')
9 - month: last contact month of year (categorical: 'jan', 'feb', 'mar', ..., 'nov', 'dec')
10 - day_of_week: last contact day of the week (categorical: 'mon','tue','wed','thu','fri')
11 - duration: last contact duration, in seconds (numeric). Important note: this attribute highly affects the output target (e.g., if duration=0 then y='no'). Yet, the duration is not known before a call is performed. Also, after the end of the call y is obviously known. Thus, this input should only be included for benchmark purposes and should be discarded if the intention is to have a realistic predictive model.
# other attributes:
12 - campaign: number of contacts performed during this campaign and for this client (numeric, includes last contact)
13 - pdays: number of days that passed by after the client was last contacted from a previous campaign (numeric; 999 means client was not previously contacted)
14 - previous: number of contacts performed before this campaign and for this client (numeric) 15 - poutcome: outcome of the previous marketing campaign (categorical: 'failure','nonexistent','success')
# social and economic context attributes
16 - emp.var.rate: employment variation rate - quarterly indicator (numeric) 17 - cons.price.idx: consumer price index - monthly indicator (numeric)
18 - cons.conf.idx: consumer confidence index - monthly indicator (numeric) 19 - euribor3m: euribor 3 month rate - daily indicator (numeric)
20 - nr.employed: number of employees - quarterly indicator (numeric)

Output variable (desired target):
21 - y - has the client subscribed a term deposit? (binary: 'yes','no').[28]

Experiment:

The dataset is a real dataset for bank marketing campaign , contain 45211 instances without missing values .We divided the dataset into 70% “training dataset” and 30%”testing dataset” randomly.
We create 3 datasets from the training dataset with different missing value percentage:

- The first dataset with 2% missing value.
- The second dataset with 6% missing value.
- The third dataset with 10% missing value.

We deleted the instances using ablebits excel tool, which select random instances and delete it manually.

The experiment has been done as following:

- We use jupyter notebook (Colab) for analysis and MidasPy, here taken a data from Kaggle/UCI and explore with Panda’s library.

Training dataset and testing dataset operations:

- Then Data have been preprocessed with Data Frame function in Panda’s library like null value dropping or replacing, converting the string to numeric value and label encoding.
- All the Data had been used for Visualization using EDA method (Exploring Data Analysis) with Matplotlib and Seaborn.
- Input and Output values have been selected for Analysis and stored in X & Y variables later it has been scaled using MinMaxScaler Function - the function is use to take minimum & maximum values in the matrix after Label Encoding.
- Data has been split in default manner for analysis with Machine Learning Model.
- MLP classifier and voting classifier has been applied for accuracy testing.

Training dataset with 2%,6%,and 10% missing values operations:

- Data have been imported through pandas attribute (read_csv) in the notebook.
- Using dataframe attribute checking the null values the dataset.
- Then label-encoding the un-null columns using Label Encoder.

- As the dataset has very little missingness, we randomly set 5,000 observed values as missing in each column.
- Next, we list categorical variables in a vector and one-hot encode them using MIDASpy's inbuilt preprocessing function cat_conv.
- Then returns both the encoded data and a nested list of categorical column names we can pass to the imputation algorithm.
- To construct the final, pre-processed data we append the one-hot encoded categorical data to the non-categorical data, and replace null values with np.nan values.

The data are now ready to be fed into the imputation algorithm, which involves three steps:

1. First, we specify the dimensions, input corruption proportion, and other hyperparameters of the MIDAS neural network.
2. Second, we build a MIDAS model based on the data. The vector of one-hot-encoded column names should be passed to the SoftMax columns argument, as MIDAS employs a SoftMax final-layer activation function for categorical variables.
3. Third, we train the model on the data, setting the number of training epochs as 20 in this example.

- Once training is complete, we can generate any number of imputed datasets (M) using the generate samples function (her we set M as 10), then either write these imputations to separate .CSV files or work with them directly.
- Using the list of generated imputations, we can estimate M separate regression models and combine the parameter and variance estimates (see Rubin 1987) using MIDASpy's combine function.
- Calling the first Imputed values into the variable and using the data for analysis with the Machine Learning Models.
- Appling MLP classifier and voting classifier for accuracy test.
- Finally comparing the values of accuracy for each and every data-sets of MLP & Voting M models in Bar chart and Line Chart with Iterations.

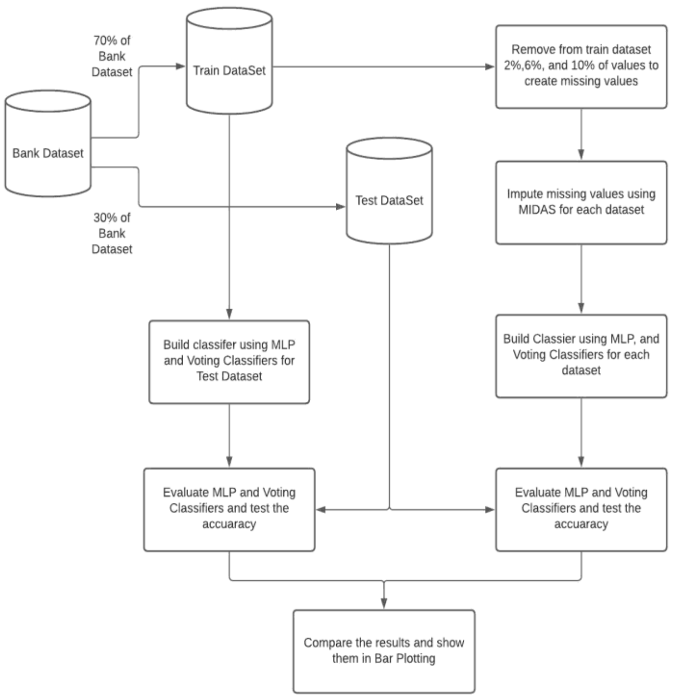
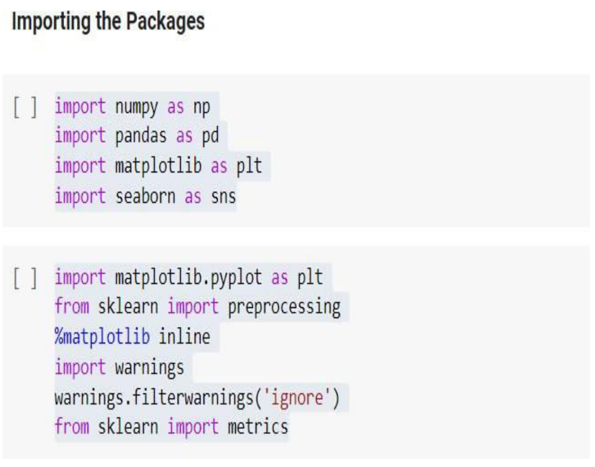

70% of Bank Dataset
Train DataSet
Remove from train dataset 2%,6%, and 10% of values to create missing values
Bank Dataset
Test DataSet
Impute missing values using MIDAS for each dataset
30% of Bank Dataset
Build classifer using MLP and Voting Classifiers for Test Dataset
Build Classier using MLP, and Voting Classifiers for each dataset
Evaluate MLP and Voting Classifiers and test the accuaracy
Evaluate MLP and Voting Classifiers and test the accuaracy
Compare the results and show them in Bar Plotting

Figure 4: Experiment Diagram

3. IMPLEMENTATION & RESULTS

In this section we will talk about the code which used for testing MIDAS model. We used Google Colab to execute the code.

- At the beginning we have to install MIDASpy using pip.
- Then we import packages which we will use in our code.


Importing the Packages

```
[ ] import numpy as np
    import pandas as pd
    import matplotlib as plt
    import seaborn as sns
```

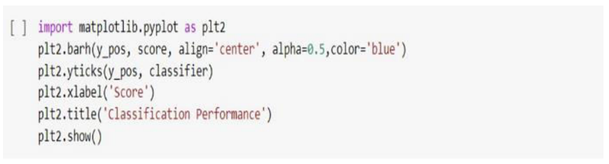
```
[ ] import matplotlib.pyplot as plt
    from sklearn import preprocessing
    %matplotlib inline
    import warnings
    warnings.filterwarnings('ignore')
    from sklearn import metrics
```

Figure 5: Importing Packages.

- Import the train and test datasets.

Comparing results:

- Present all results using Bar Plotting.



```
[ ] import matplotlib.pyplot as plt2
    plt2.barh(y_pos, score, align='center', alpha=0.5,color='blue')
    plt2.yticks(y_pos, classifier)
    plt2.xlabel('Score')
    plt2.title('Classification Performance')
    plt2.show()
```

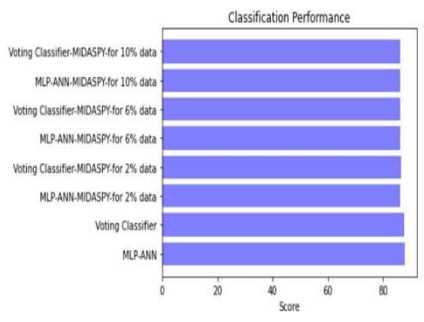
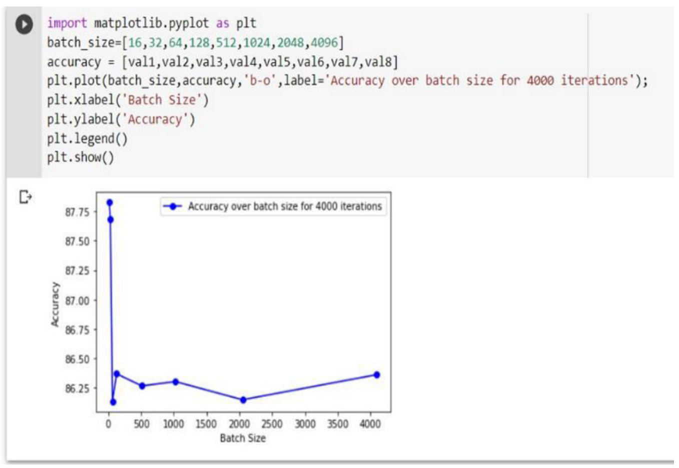


Figure 6:Bar Plotting for MLP And Voting Classifiers Results.

- Present all results using Line Plotting with Iterations.



```
import matplotlib.pyplot as plt
batch_size=[16,32,64,128,512,1024,2048,4096]
accuracy = [val1,val2,val3,val4,val5,val6,val7,val8]
plt.plot(batch_size,accuracy,'b-o',label='Accuracy over batch size for 4000 iterations');
plt.xlabel('Batch Size')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```

Figure 7: Line Plotting with Iterations for MLP and Voting Classifiers results.

4. CONCLUSION

There are many libraries and techniques in python used for missing data imputation like missforest, simpleimputer class, k-NN algorithm, and others.

In this paper we have introduced a new model for missing data imputation MIDASpy. Based on results we got from MLP and Voting Classifier we can say that MIDASpy perform well on big dataset.

The results of MLP and Voting Classifiers for test dataset almost the same compared to the results of MLP and Voting Classifiers results of train dataset with 2%,6%, and 10% missing value.

5. REFRENCES

1. Pei Chen. Optimization algorithms on subspaces: Revisiting missing data problem in low- rank matrix. International Journal of Computer Vision, 80(1):125{142,2008.
2. Donald B Rubin. Inference and missing data. Biometrika, pages 581{592, 1976.
3. Roderick JA Little. Missing-data adjustments in large surveys. Journal of Business & Economic Statistics, 6(3):287{296, 1988.
4. Lovedeep Gondara and Ke Wang “MIDA: Multiple Imputation using Denoising Autoencoders” ,Simon Fraser University,2018.
5. Vincent P, Larochelle H, Lajoie I, Bengio Y, Manzagol PA (2010) Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. J Mach Learn Res 11(12):3371–3408
6. Vincent P, Larochelle H, Bengio Y, Manzagol PA (2008) Extracting and composing robust features with denoising autoencoders. In: Proceedings of international conference on machine learning, pp 1096– 1103
7. Baştanlar, Yalin, and Mustafa Özuysal. "Introduction to machine learning." miRNomics: MicroRNA Biology and Computational Analysis (2014): 105-128.
8. Simeone, Osvaldo. "A brief introduction to machine learning for engineers." arXiv preprint arXiv:1709.02840 (2017).
9. Dongare, A. D., R. R. Kharde, and Amit D. Kachare. "Introduction to artificial neural network." International Journal of Engineering and Innovative Technology (IJEIT) 2.1 (2012): 189-194.
10. Maind, Sonali B., and Priyanka Wankar. "Research paper on basic of artificial neural network." International Journal on Recent and Innovation Trends in Computing and Communication 2.1 (2014): 96-100.
11. Schafer, Joseph L., and John W. Graham. "Missing data: our view of the state of the art." Psychological methods 7.2 (2002): 147.
12. T. Johnson and T. Dasu. Data quality and data cleaning: An overview. In SIGMOD, page 681, 2003.
13. E. Rahm and H. H. Do. Data cleaning: Problems and current approaches. IEEE Data Eng. Bull., 23(4), 2000
14. Rahm, Erhard, and Hong Hai Do. "Data cleaning: Problems and current approaches." IEEE Data Eng. Bull. 23.4 (2000): 3-13.
15. Saunders, Jeanne A., et al. "Imputing missing data: A comparison of methods for social work researchers." Social work research 30.1 (2006): 19-31.
16. Ma, Q., Lee, W.-C., Fu, T.-Y., Gu, Y., & Yu, G. (2020). MIDIA: exploring denoising autoencoders for missing data imputation. Data Mining and Knowledge Discovery.
17. Gondara, Lovedeep, and Ke Wang. "Mida: Multiple imputation using denoising autoencoders." Pacific-Asia conference on knowledge discovery and data mining. Springer, Cham, 2018
18. Lall, Ranjit, and Thomas Robinson. "The MIDAS touch: accurate and scalable missing- data imputation with deep learning." Political Analysis (2020): 1-18.[

19. Song, Q., & Shepperd, M. (2007). Missing Data Imputation Techniques. International Journal of Business Intelligence and Data Mining, 2(3), 261.
20. Costa, Adriana Fonseca, et al. "Missing data imputation via denoising autoencoders: the untold story." International symposium on intelligent data analysis. Springer, Cham, 2018.
21. Ruck, Dennis W., Steven K. Rogers, and Matthew Kabrisky. "Feature selection using a multilayer perceptron." Journal of Neural Network Computing 2.2 (1990): 40-48
22. Tsai, Chih-Fong, Yu-Feng Hsu, and David C. Yen. "A comparative study of classifier ensembles for bankruptcy prediction." Applied Soft Computing 24 (2014): 977-984.[
23. Zhang, Yong, et al. "A weighted voting classifier based on differential evolution." Abstract and Applied Analysis. Vol. 2014. Hindawi, 2014.
24. Kumar, U. Karthik, MB Sai Nikhil, and K. Sumangali. "Prediction of breast cancer using voting classifier technique." 2017 IEEE international conference on smart technologies and management for computing, communication, controls, energy and materials (ICSTM). IEEE, 2017.
25. Oliphant, Travis E. A guide to NumPy. Vol. 1. USA: Trelgol Publishing, 2006.
26. Aiken, John M., Chastity Aiken, and Fabrice Cotton. "A Python library for teaching computation to seismology students." Seismological Research Letters 89.3 (2018): 1165-1171.
27. Chen, Ying, et al. "Performance modeling for the Panda array I/O library." Proceedings of the 1996 ACM/IEEE Conference on Supercomputing. 1996.
28. [Moro et al., 2014] S. Moro, P. Cortez and P. Rita. A Data-Driven Approach to Predict the Success of Bank Telemarketing. Decision Support Systems, Elsevier, 62:22-31, June 2014

BIOGRAPHIES

IBRAHIM A I HAMEED
M. Tech Computer Science and Technology, AU College of Engineering (A) Andhra University.