Webology (ISSN: 1735-188X)
Volume 17, Number 2, 2020

Automated Data Pre-Processing Using Flask Framework

T.MAMATHA[1] , B. RAMA SUBBA REDDY[2] , C SHOBA BINDU[3]

[1]Research Scholar, Dept. of CSE, JNTUA University, Anantapuramu, AP, INDIA

[2]Professor & HOD, Dept. of CSE, SV College of Engineering, Tirupati, INDIA

[3]Professor , Dept. of CSE , JNTUA University, Anantapuramu, AP, INDIA

Abstract-Pre-processing of the data addresses the solutions for inaccurate data in two major fields in the current day computer science– Machine learning and Data mining. Data, in real world scenarios, that had taken from direct sources is often prone to errors, inaccuracies and inconsistencies and incomplete. These datasets are not considered to be used in data mining and Machine learning algorithms as inaccurate data often misleads the results of algorithms. Poor quality of the data leads to poor decision as a decision can be no better than the information upon which it's based. Critical decisions and important calculations that might serve a key purpose in the evaluation might get corrupted. Pre-processing of the data resolves the issues and ensure the data is much more suitable as input for the algorithms.

Key Words: Datasets, data pre-processing, web application, flask framework, data cleaning, data reduction, null values, categorical data, data encoding, splitting dataset, python

1. INTRODUCTION

Data Pre-processing is a technique that is used to convert the raw data into a clean data set[2]. In other words, whenever the data is gathered from different sources it is collected in raw format which is not feasible for the analysis. Therefore, certain steps are executed to convert the data into a small clean data set. This technique is performed before the execution of the iterative analysis. The set of steps is known as Data Pre-processing. Frameworks like Numpy, Pandas[8], Sklearn[9] and programming languages like R in environments like R studio, Jupyter notebook, Anaconda accomplish a great job in pre-processing the datasets which requires programming skills and frame work knowledge.

The idea proposed is to develop a web application for pre-processing the datasets by providing most important pre-processing operations with proper user interface developed using Flask[3][4], Pandas frameworks[8] in python programming language.

2. MODULES IN PREPROCESSING

2.1 Data Cleaning

The data cleaning operation is the primary step in pre-processing. Data cleaning refers to the handling of incomplete data and null values in given dataset. The steps taken to handle the missing data are
1. Ignoring the missing data
2. Filling the missing values manually
3. Filling the missing values automatically
4. Using global constant to fill the missing data

Data cleaning operation not only handles the missing data but also smoothens out the noisy data by handling the outliers, correct the inconsistent data. Data cleaning resolves the redundancy caused by data integration operations.

Ignoring the missing data is usually done when the class labels are missing. Filling of the missing data manually can be done by using the measures of central tendency. This can be achieved by using mean(average) or mode or median of the values in an attribute to fill the missing values in the attribute.

2.2Attribute Handling

Attribute handling refers to attribute scaling and data reduction in pre-processing. Attribute scaling can also be referred as Normalization where the values of the attribute can be scaled to new range from existing range. There are two major normalization techniques
1. Min max normalization
2. Decimal scaling

Min max normalization[3] is scaling the data from the existing range to newly mentioned range whereas decimal scaling is the scaling of the values in attribute exist range 0 and 1. Data reduction is the concept of reducing the attributes that are not important or unnecessary by deleting them. As the unimportant columns are being deleted, the size and the dimensions of the dataset are reduced.

2.3Handling Categorical Data

Using categorical data directly into the data mining methods[5] might be an efficient method but machine learning and deep learning algorithms does not understand the categorical data as what the data it is. Machine learning algorithms have a hard time processing categorical data. The categorical data can be transformed into numerical data assigning the numerical values to the class labels in the attribute based on the type of encoding being used. Few most commonly used encoding techniques are
1. One-hot encoding
2. Label encoding

In one-hot encoding, new columns are added to the dataset for each label in the class and binary values are assigned for all the newly generated classes by assigning 1 if the column is the previous label in the class for that tuple and 0 if not. Whereas in Label encoding, no new columns are generated but

2.4 Splitting The Dataset

Data splitting appear as a prerequisite to eliminate bias in training data in ML algorithms. While modifying parameters of an ML algorithm in order to best fit the training data, usually results in over fitting. For this reason, we split the dataset into multiple, discrete subsets on which we train different parameters. It usually consists of Train and Test sets.

i) Training set:

The Training set is used to evaluate the actual model your algorithm will employ when subjected to new data. This dataset is typically 60-80% of your available data.

ii) Test set:

The Test set is the final dataset you touch (generally 20% of your data). Your accuracy in concluding the test set is the accuracy of your ML algorithm.

3.Features

3.1 Description

Description refers to the properties of a dataset[7]. The properties include the number of entries the data set contain, the number of columns or attributes. It also consists of the data type of each column from the dataset, its count of values, and finally the memory used to store the dataset.info() method is used to get a concise summary of the data frame. It is very helpful when doing exploratory analysis[4] of the data. Desc () method is used to get deep analysis of the data frame used to get the information about the column.

desc=io.StringIO()dataframe.info(buf=desc)dataframe[column_name].describe()

3.2 Handling Null Values

Pandas dropna() function is used to remove all the missing values from the column. This is usually done when the class label is missing[2] (when your data mining goal is classification), or many attributes are missing from the row. The in place parameter of this function is used to make the changes in data frame itself if assigned True.

Data frame. Dropna (inplace=True)

Replacing missing values of an attribute with the statistical value is other method of handling missing data. Simple imputer in Sklearn is used to fill missing values using mean, mode, median, constant using strategy parameter in Sk learn. Pre processing. Simple Imputer()

si=Simple Imputer (missing_values=np.nan,
strategy)si=si.fit_transform(dataframe[[column_name1]])df[column_name1]=pd.DataFrame(si)

3.3 Feature handling

It is an important pre-processing step for the structured dataset in supervised learning[1].

Min-max normalization is one of the normalization techniques.It ranges data from 0-1.It is calculated as

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \quad \textbf{(1)}$$

a=Min Max Scaler(). fit_ transform (data frame [[column name]])
dataframe[column_name]=pd.DataFrame(data=a,columns=[column_name])

Standardization is used to scale the data such that mean is very close to 0 and standard deviation '1'.StandardScaler() is the method used to perform standardization to a column.

$$x_{\text{stand}} = \frac{x - \text{mean}(x)}{\text{standard deviation } (x)} \quad \textbf{(2)}$$

a=Standard Scaler().fit_trans form(data frame[[column name]]) data frame[column_name]=pd. Data Frame(data=a, columns=[column_name])

Deleting unnecessary columns is also a part of feature handling. Pandas drop() method is used to drop/delete the column from the data set.The parameters include the column name which you want to delete, and axis which specifies whether to drop labels from the index (0 or 'index') or columns (1 or 'columns').

dataframe = dataframe.drop(columns=column_name, axis=1)

3.4 Encoding

With one-hot encoding, a categorical feature becomes an array whose size is the number of labels available for that attribute.

Pandas concat () method joins the columns. For suppose in this case the existing column is dropped and joined with the new encoded column. The pandas.get_dummies() method does one hot encoding. The parameters include the column name, prefix(String to append Data Frame column names),and prefix_sep (appending prefix, separator/delimiter to use or pass a list or dictionary as with prefix.)

dataframe=pandas.concat([dataframe.drop(column_name,axis=1),
pandas.get_dummies(dataframe[column_name], prefix=column_name, prefix_sep='_')], axis=1)

Label encoding is done using LabelEncoder() and fit_transformmethods.LabelEncoder() object encodes the categorical column as machine understandable values using fit_transform method.

le=LabelEncoder() label=le.fit_trans form(df [column_name])
dataframe=dataframe.drop(column_name, axis='columns')
dataframe[column_name]=label

3.5 Splitting the dataset

Splitting the large dataset into test and train datasets is the best way for training and evaluating the accuracy and efficiency of the algorithm[2][3]. The whole dataset is divided into two sets. Train Dataset is Used to fit and train the machine learning model. And Test Dataset is Used to evaluate the fit machine learning model[2][6].

x: independent variables or attributes in the data set y: dependent variables or attributes in the dataset percent: percentage of no. of rows in test data x_train, x_test, y_train, y_test = train_test_split (data frame[x], data frame[y], test_size=percent)

4. Results

For the demonstration of this web application the Titanic dataset is used.

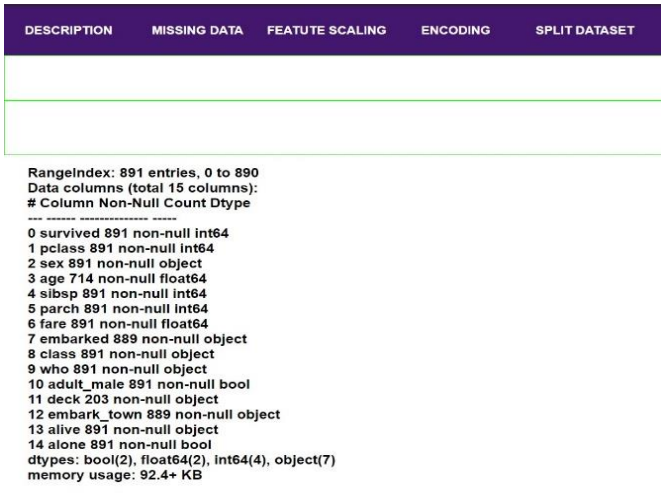


Fig 4.1 Properties of the titanic dataset

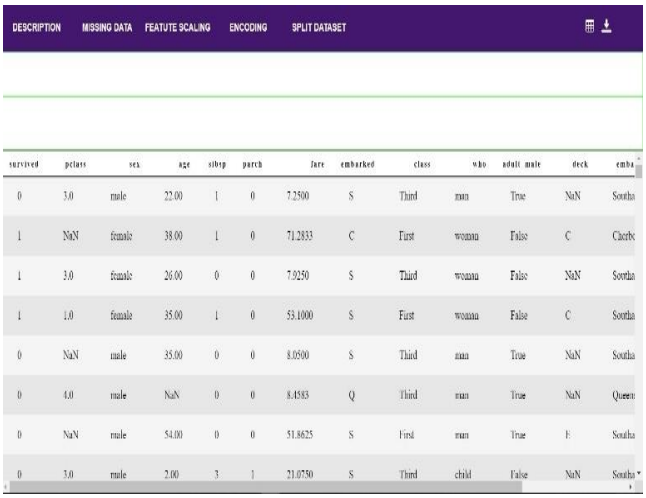
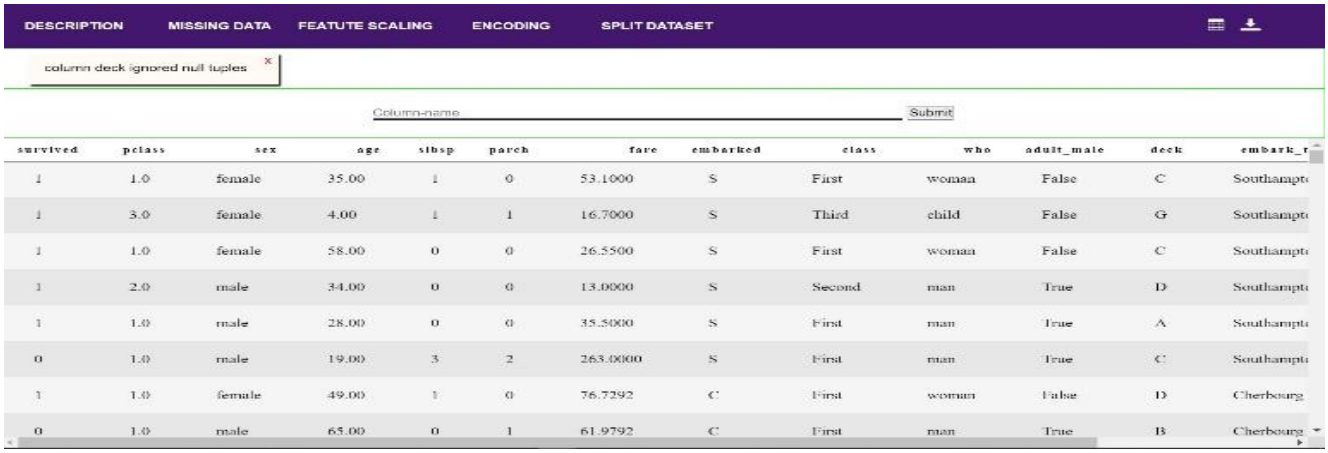
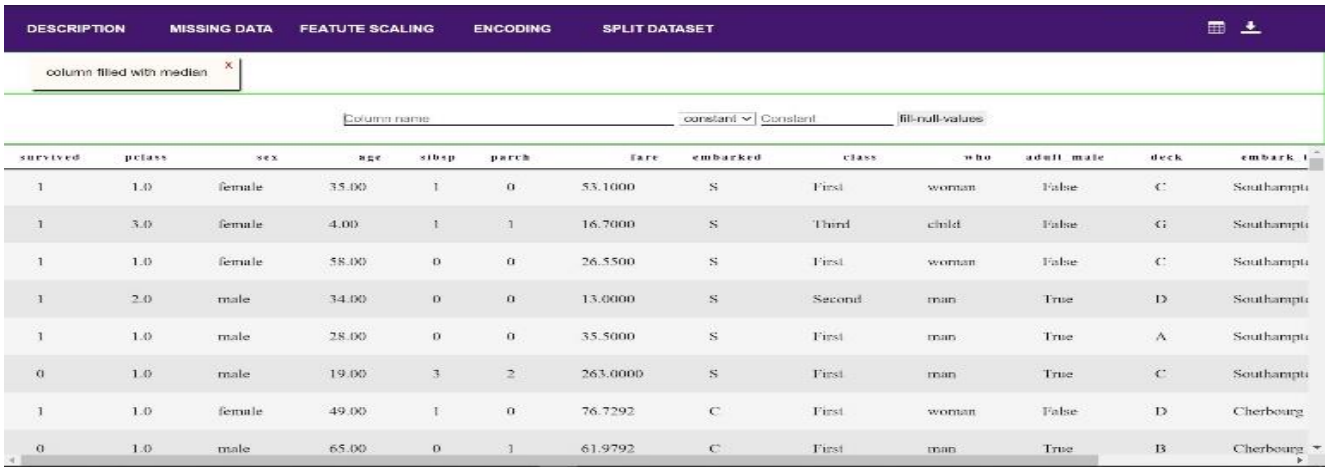


Fig 4.2 Titanic dataset before handling null values



survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_t
1	1.0	female	35.00	1	0	53.1000	S	First	woman	False	C	Southampt
1	3.0	female	4.00	1	1	16.7000	S	Third	child	False	G	Southampt
1	1.0	female	58.00	0	0	26.5500	S	First	woman	False	C	Southampt
1	2.0	male	34.00	0	0	13.0000	S	Second	man	True	D	Southampt
1	1.0	male	28.00	0	0	35.5000	S	First	man	True	A	Southampt
0	1.0	male	19.00	3	2	263.0000	S	First	man	True	C	Southampt
1	1.0	female	49.00	1	0	76.7292	C	First	woman	False	D	Cherbourg
0	1.0	male	65.00	0	1	61.9792	C	First	man	True	B	Cherbourg

Fig 4.3 Titanic dataset after ignoring null values in column ‘deck’



survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_t
1	1.0	female	35.00	1	0	53.1000	S	First	woman	False	C	Southampt
1	3.0	female	4.00	1	1	16.7000	S	Third	child	False	G	Southampt
1	1.0	female	58.00	0	0	26.5500	S	First	woman	False	C	Southampt
1	2.0	male	34.00	0	0	13.0000	S	Second	man	True	D	Southampt
1	1.0	male	28.00	0	0	35.5000	S	First	man	True	A	Southampt
0	1.0	male	19.00	3	2	263.0000	S	First	man	True	C	Southampt
1	1.0	female	49.00	1	0	76.7292	C	First	woman	False	D	Cherbourg
0	1.0	male	65.00	0	1	61.9792	C	First	man	True	B	Cherbourg

Fig 4.4 Titanic dataset after filling null values in column ‘p class’ with constant ‘3’

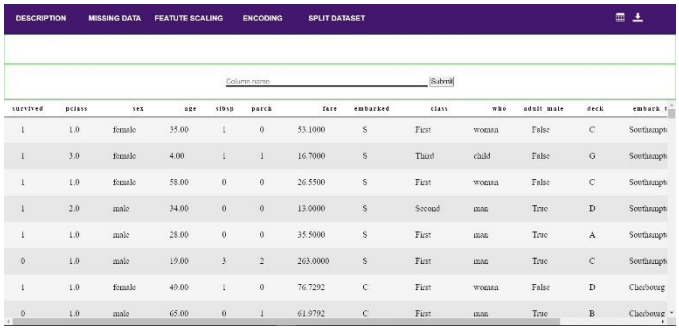


Fig 4.5 Titanic dataset before normalization

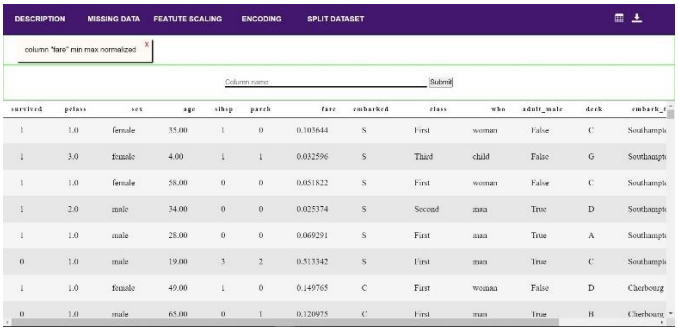
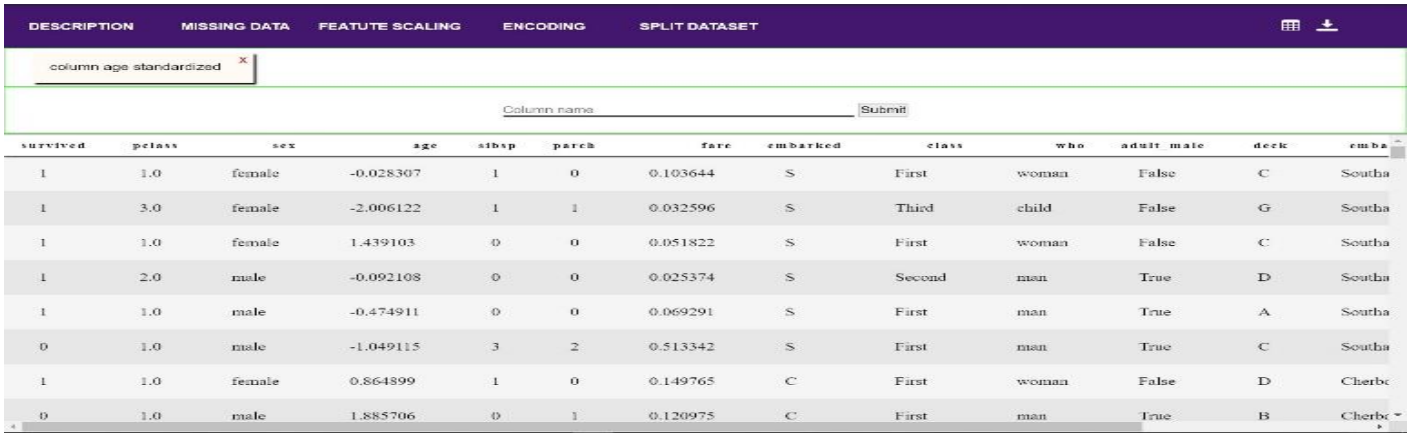
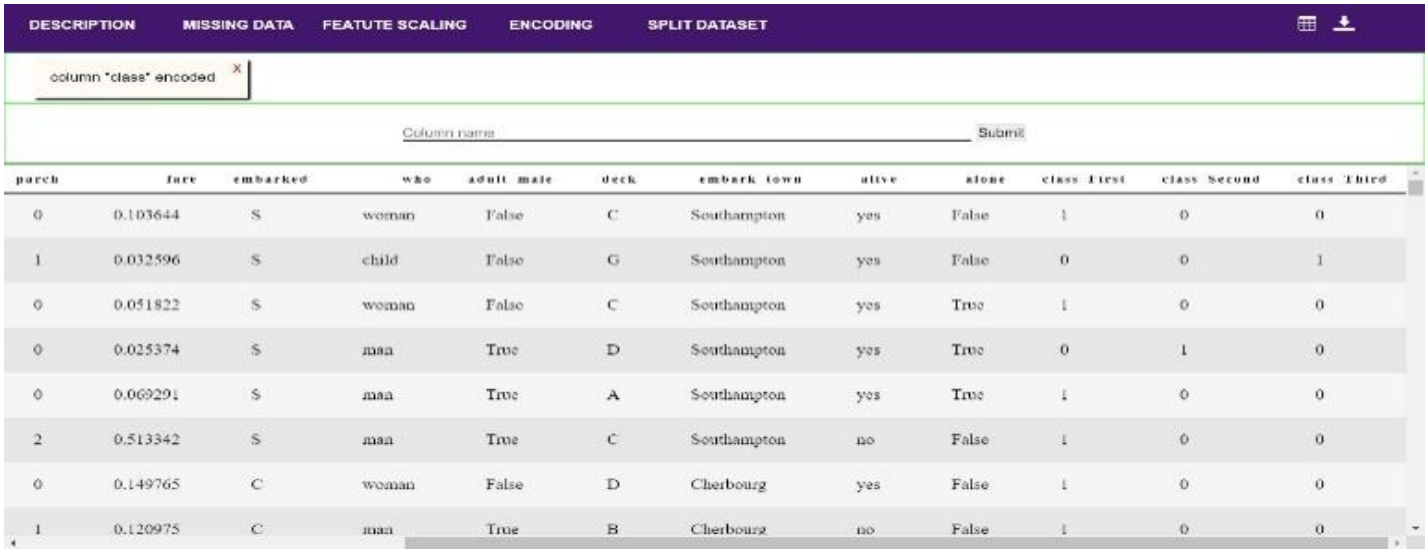


Fig 4.6Titanic dataset after minmax normalizing ‘fare’



survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	emba
1	1.0	female	-0.028307	1	0	0.103644	S	First	woman	False	C	Southa
1	3.0	female	-2.006122	1	1	0.032596	S	Third	child	False	G	Southa
1	1.0	female	1.439103	0	0	0.051822	S	First	woman	False	C	Southa
1	2.0	male	-0.092108	0	0	0.025374	S	Second	man	True	D	Southa
1	1.0	male	-0.474911	0	0	0.069291	S	First	man	True	A	Southa
0	1.0	male	-1.049115	3	2	0.513342	S	First	man	True	C	Southa
1	1.0	female	0.864899	1	0	0.149765	C	First	woman	False	D	Cherbo
0	1.0	male	1.885706	0	1	0.120975	C	First	man	True	B	Cherbo

Fig 4.7 Titanic dataset after standardizing column ‘age’



parch	fare	embarked	who	adult_male	deck	embark_town	alive	alone	class_First	class_Second	class_Third
0	0.103644	S	woman	False	C	Southampton	yes	False	1	0	0
1	0.032596	S	child	False	G	Southampton	yes	False	0	0	1
0	0.051822	S	woman	False	C	Southampton	yes	True	1	0	0
0	0.025374	S	man	True	D	Southampton	yes	True	0	1	0
0	0.069291	S	man	True	A	Southampton	yes	True	1	0	0
2	0.513342	S	man	True	C	Southampton	no	False	1	0	0
0	0.149765	C	woman	False	D	Cherbourg	yes	False	1	0	0
1	0.120975	C	man	True	B	Cherbourg	no	False	1	0	0

Fig 4.8 Titanic dataset after one-hot encoding column ‘class’ creating 3 columns ‘class_First’, ‘class_Second’, ‘class_Third’

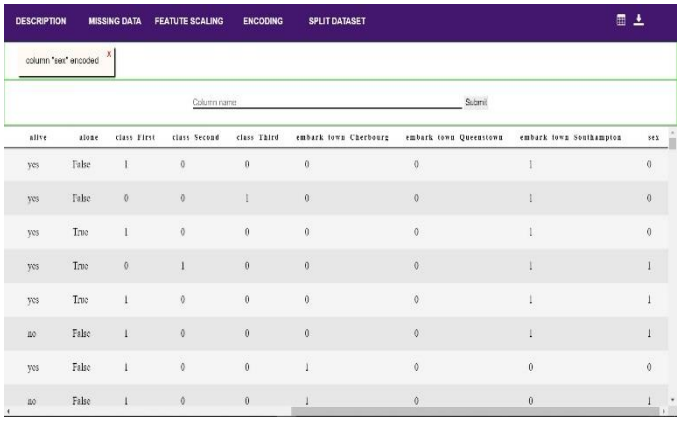


Fig 4.9 titanic dataset after label encoding column ‘sex’

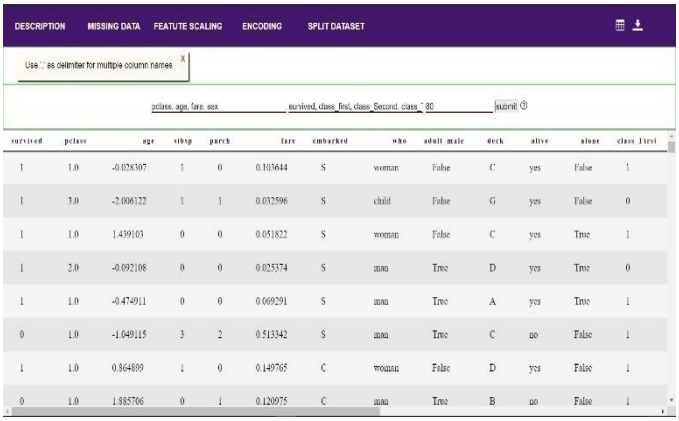
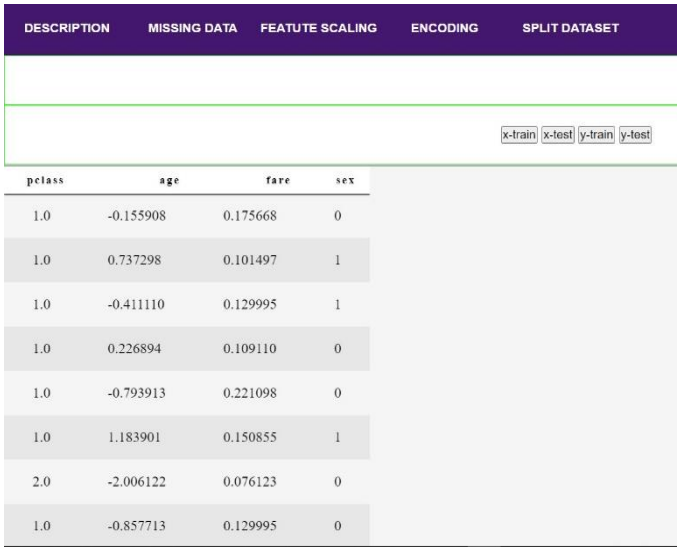
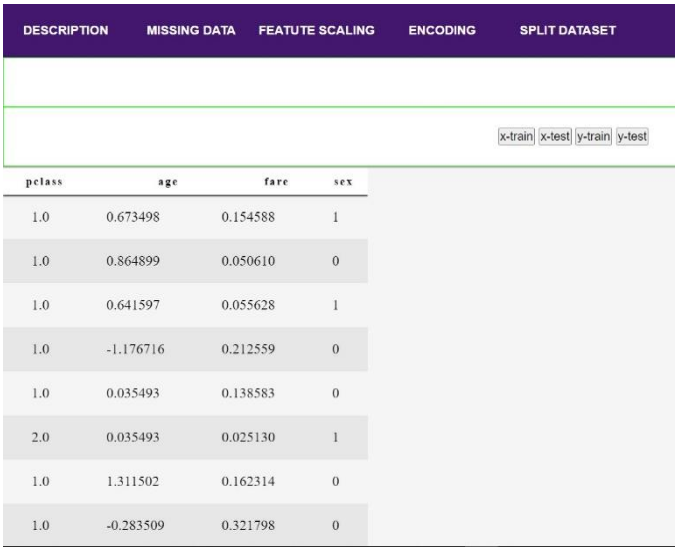


Fig 4.10 titanic dataset before splitting



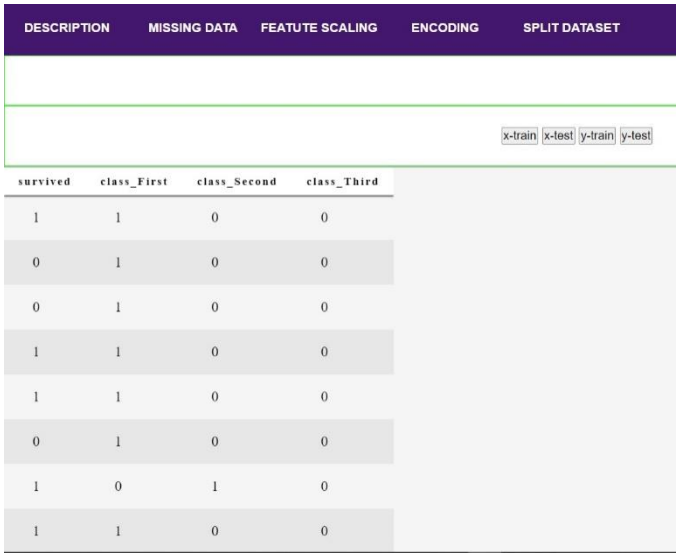
pclass	age	fare	sex
1.0	-0.155908	0.175668	0
1.0	0.737298	0.101497	1
1.0	-0.411110	0.129995	1
1.0	0.226894	0.109110	0
1.0	-0.793913	0.221098	0
1.0	1.183901	0.150855	1
2.0	-2.006122	0.076123	0
1.0	-0.857713	0.129995	0

Fig 4.11 x-train dataset after splitting of dataset



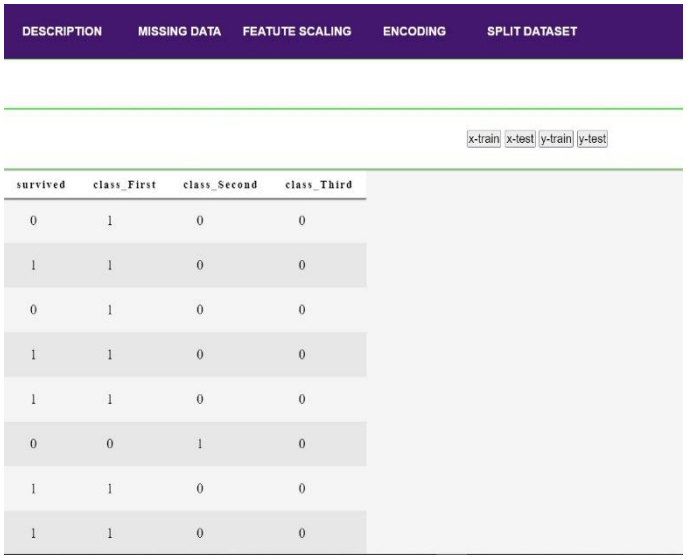
pclass	age	fare	sex
1.0	0.673498	0.154588	1
1.0	0.864899	0.050610	0
1.0	0.641597	0.055628	1
1.0	-1.176716	0.212559	0
1.0	0.035493	0.138583	0
2.0	0.035493	0.025130	1
1.0	1.311502	0.162314	0
1.0	-0.283509	0.321798	0

Fig 4.12 x-test dataset after splitting of titanic dataset



survived	class_First	class_Second	class_Third
1	1	0	0
0	1	0	0
0	1	0	0
1	1	0	0
1	1	0	0
0	1	0	0
1	0	1	0
1	1	0	0

fig 4. 13 y-train dataset after splitting of titanic dataset



survived	class_First	class_Second	class_Third
0	1	0	0
1	1	0	0
0	1	0	0
1	1	0	0
1	1	0	0
0	0	1	0
1	1	0	0
1	1	0	0

Fig 4.14. y-test dataset after splitting of titanic dataset

5. Conclusion and Future scope

Data pre-processing is used in both databased-driven and rules-based applications. In fields like Machine Learning, data pre-processing is the most crucial step to be performed to ensure that the dataset is good to fed to the algorithm so that the trained algorithm works at maximum accuracy. In the field of Data Science, data pre-processing is the important technique to transform the raw data into useful and efficient format.

Data pre-processing web application using flask developed to address the basic most frequently used features in data pre-processing techniques and can be further developed to support every aspect in data pre-processing with more sophisticated user interface in more descriptive, detailed manner.

6. References

[1] Sumian Peng, “Research on Data Pre processing Process in the Web Log Mining,” IEEE Conference. Information Science and Engineering (ICISE), 2009 1st International Conference, pp. 942 - 945, 26-28 Dec. 2009
[2] Sudheer Reddy, K,” An effective data pre processing method for Web Usage Mining”, IEEE Conference Information Communication and Embedded Systems (ICICES), 2013 International Conference. 7 - 10 Feb. 2013.
[3] G. Miguel, “Flask Web Development,” O’Reilly Media, Inc. 2014.
[4] https://flask.palletsprojects.com/en/2.0.x/tutorial/
[5] https://jinja.palletsprojects.com/en/3.0.x/
[6] P Hurtik, V Molek and J. Hula, "Data Pre processing Technique for Neural Networks Based on Image Represented by a Fuzzy Function[J]", Ieee Transactions on Fuzzy Systems, vol. 28, no. 7, pp. 1195-1204, 2020.
[7]https://www.w3schools.com/html/default.asp
[8]https://pandas.pydata.org/docs/user_guide/index.html#user-guide
[9]https://scikit- learn.org/stable/modules/pr eprocessing. html# preprocessing

Authors:

T Mamatha received her B.Tech Degree in Computer Science and Engineering from Jawaharlal Nehru Technological University, Hyderabad, TS, in 2003. She received her M.Tech in Software Engineering at JNTUA University in 2007. At present she is working as an Assistant professor in Computer Science and Engineering , Sreenidhi Institute of Science and Technology, Hyderabad,TS INDIA. Her research interest includes Software Engineering, Software Testing, Machine Learning and Networking

B Rama Subba Reddy received his B.Tech Degree in Computer Science and Engineering From Jawaharlal Nehru Technological University, Anantapur, India, in 1997. M.Tech (CSE)From Mysore university, Mysore in 2002. PhD (Data Mining) From SV University, Tirupati 2014. At present he is working as a Professor & Vice Principal in Computer Science and Engineering Department , S V Engineering College, Tirupati, AP, INDIA. His research interest includes Network Security , Wireless Communication system and Software Engineering

C Shoba Bindu received her B.Tech Degree in Electronics & Communication Engineering from Jawaharlal Nehru Technological University, Anantapur, India, in 1997. M.Tech. in Computer Science and Engineering from JNTUA University, Anantapuram in 2002. She received her Ph.D in Computer Science and Engineering at JNTUA University in 2010.. At present she is working as a professor in Computer Science and Engineering , JNTUA, Anantapuram, AP, INDIA. Her research interest includes Network Security, Wireless Communication system and Software Engineering