

An Enhancement of Deep Feature Synthesis Algorithm Using Mean, Median, and Mode Imputation

Josefa Ysabelle J. Maliwat, Princess A. Ylade,

Richard C. Regala, Dan Michael A. Cortez, Antolin J. Alipio, Khatalyn E. Mata, Mark Christopher R. Blanco,
Computer Science Department Pamantasan ng Lungsod ng Maynila Manila, NCR, Philippines

Abstract:- The Deep Feature Synthesis (DFS) algorithm automates feature engineering and is capable of extracting and applying complicated features to a variety of processes. Due to the novelty of DFS as a method for feature engineering, critical ways for dealing with missing values and unwanted data in a dataset have yet to be established. This paper discusses the usage of mean, median, and mode imputation to preprocess data before analyzing it. However, it is only limited to displaying the differences between non-imputed and imputed datasets. This strategy enables users to obtain more precise results by eliminating biased estimations. This study demonstrates that there is a distinct difference between the two datasets. This paper is concluded by proving that imputing datasets will cause distinctness in the results compared to the results of the datasets with missing and unwanted values.

Keywords:- Deep Feature Synthesis, Auto Feature Engineering, Imputation.

I. INTRODUCTION

Automated feature engineering intends to aid data scientists through the automatic creation of features from a dataset in which, the best out of all the features, can be chosen and can be used for training [1]. Creating features, especially through the use of data from different tables, can be a tedious process for the reason that each new feature typically requires a number of steps to construct.

The Deep Feature Synthesis (DFS) is an algorithm that automatically allows users to generate features for relational datasets. Relational data is currently the most commonly used type of enterprise data, which is why this type of data is being focused on. DFS is auspicious when aggregating features in a relational database structure [2]. The algorithm analyzes and follows the relationships of the different data in a base field. Then, it applies mathematical functions over the features to create the final output, a base table that portrays a summary of the relational data.

By applying mathematical functions and calculations, the final features of the datasets can be defined. Its process has a set of related entities and tables. Although the natural language descriptions are entirely different, the fundamental math remains the same. The same operation on a list of numeric values to generate a new numeric feature unique to the dataset is used. These could be repeatedly stacked to establish complex features. Features can be created with many tables because DFS

applies primitives across different relationships between datasets. Several mathematical methods create new features, including getting the mean, median, or mode. However, DFS still does not have procedures to treat the null and unwanted values from each dataset.

The main focus of this study is to handle missing values by using mean, median, and mode imputation. The researchers recognize that some situations do not require imputation, so imputing is made optional for the users. After utilizing the imputation technique, this paper only shows the differences between the non-imputed and imputed datasets.

II. REVIEW OF RELATED LITERATURE

The automation of feature engineering opens the door to expedite the process of applying machine learning to the datasets that are collected by data science teams nowadays [3]. Data scientists can now address new problems swiftly as they arise. Additionally, automated feature engineering will make it easier for new learners of data science to develop the necessary skills needed and apply it to their own field of expertise.

Feature extraction is an essential method in computer vision and it is used for object recognition, image alignment, navigation for robots, etc [4]. Convolutional Neural Network (CNN) is one of the popular deep learning techniques that deal with knowledge representation. It divides the images into layers so that each layer is more carefully studied compared to traditional analysis procedures. CNN's strong capacity in extracting complex features makes it easier to learn task specific features making it more efficient.

While deep learning automates feature engineering for text, audio, and images wherein a huge training set is required, the target of deep feature synthesis is to automatically engineer features for structured transactional and relational datasets. DFS can already start producing features based only on the schema of a dataset. On the other hand, many training examples are vital in order to train the complex architectures of deep learning for it to work.

III. PROPOSED METHOD

To achieve the objective of having the option to preprocess the data before extracting its features, this paper proposes the use of imputation, particularly the mean, median, and mode imputation methods.

A. Adding the mean, median, and mode imputation into the DFS Algorithm

Mean and median imputation works by filling in the missing values by the mean or median for quantitative attributes in a dataset. Mode imputation uses the most common value to replace the missing values for qualitative attributes. After loading the datasets and containing them in their respective variables, a temporary list holding the datasets to be imputed is made. This is done to loop through the datasets and pass them into the imputation function. The function will return the imputed datasets and pass them back to the temporary list to overwrite non-imputed datasets. After this process, the datasets are now ready for feature extraction.

B. Imputing the dataset using the mean, median, and mode

The imputation Module function will receive the data frame and the choice of imputation method to be used for numerical data types. The data frame to be imputed will enter the function. Then, the column Name and column Data will loop through the contents. After identifying the column's data type, the missing values will be imputed according to the choice of imputation method passed in the parameter. For string data types, mode imputation will automatically be used.

C. Pseudocode

```
function IMPUTATION(dataframe, method):
for columnName, columnData in dataframe.iteritems():
if datafram[columnName].dtype == 'object':
    datafram[columnName] =
        datafram[columnName].fillna(datafram[columnName].mod
        e()[0])
elif datafram[columnName].dtype == 'float64' or 'int64':
    if method == 'mean':
        datafram[columnName]
        =datafram[columnName].fillna(datafram[columnName].me
        an())
    elif method == 'median':
        datafram[columnName]
        =datafram[columnName].fillna(datafram[columnName].me
        dian())
    elif method == 'mode':
        datafram[columnName]
        =datafram[columnName].fillna(datafram[columnName].mo
        de())
return datafram
```

IV. RESULTS AND DISCUSSION

Fig. 1 shows the result of data aggregation with the use of the raw dataset. Fig. 2 shows the results of data aggregation with the use of the imputed dataset. The highlighted cells in Figures 1 and 2 are the values that have a difference between the results. As shown in Fig. 2, there is a difference between the values in the feature matrix of the non-imputed datasets and the imputed ones.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	00001	6	desktop	2	70	140.07	5	0.000000	0.000000	7.50	1	0.00057	0.00047	48.0000	1.47000	3017.71	200	28
2	00001	8	mobile	3	100	140.95	5	28.0000	0.000000	5.75	1	0.13300	0.13247	0.000000	1.01000	205.53	299	15
3	00001	8	mobile	3	100	139.43	5	70.0000	1.10900	5.51	1	0.00086	0.24000	39.0000	1.40000	771.42	274	47
4	12344	6	desktop	1	90	140.15	5	68.5000	2.877777	5.80	1	0.43213	0.00018	42.7220	1.40000	374.63	239	21
5	12344	7	desktop	3	80	140.81	5	75.0000	3.14600	8.73	1	0.34000	0.00000	36.6739	1.30000	420.00	303	18

Fig. 1. (a) The image shows the first 19 columns of the feature matrix using the non-imputed datasets.

1	U	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK	AL
17	7	7	5	5	1000	2010	17	96.4475	1.000000	20.05	1	0.00200	0.67200	51.8200	1.65000	150.16	50.22	66.667
8	8	8	1	4	2000	2011	8	10.45	5.45	4.81	2	0.45000	1.17900	55.9800	1.61000	137.11	63	12.875
17	7	4	1	6	1000	2011	23	96.5510	4.75	10.13	1	0.45052	0.34074	40.0152	1.80700	188.63	64	14.75
13	12	8	4	5	2000	2011	17	82.8375	3.737407	20.06	1	0.72000	0.04820	30.0755	1.70900	132.00	54	15
39	8	4	0	0	1000	2012	17	96.4475	3.01700	16.46	1	0.07100	0.34000	41.4111	1.30000	300.00	56	13.740

Fig. 1. (b) The image shows the second 19 columns of the feature matrix using the non-imputed datasets.

AA	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK	AL	AM	AN	AO	AP	AQ	AR	AS
139.96	-5	79.9795	30.0000	14.415	1	-0.07238	-0.00000	42.70329	14.01112	979.587	30.3333	8	128.51	0.05100	2.58825	0.04000	0.70100	
144.7008	4.675	79.9949	2.88211	16.41075	1.122	0.00052	0.30000	42.70329	14.01112	979.587	30.3333	8	139.2	0.60	30	2.7174	-0.40000	
121.82	3	72.0020	1.23026	13.49102	1.03495	0.00409	0.20000	28.55304	14.01703	974.286	46.75	11	118.9	5.504232	1.37449	0.98000	1.10000	
149.927	3	97.0212	2.880043	11.200	1	0.00000	0.00000	42.71207	15.00000	954.000	41.1000	11	126.74	5.553794	2.4010	0.17100	0.60700	
123.207	3	77.00440	1.310056	25.30754	1	0.00482	0.17121	35.10002	13.00728	990.307	43	7	105.04	1.00	3.00	2.70000	0.10000	

Fig. 1. (c) The image shows the third 19 columns of the feature matrix using the non-imputed datasets.

AT	BT	CT	DT	ET	FT	GT	HT	IT	KT	LT	MT	NT	OT	PT	RT	ST	WT	XT
33.37198	1.3225	543.18	76	1	1	7	2014	1	1	1	1	1	1	1	1	1	1	1
29.02042	1	667.58	22	1	1	2	2014	1	1	1	1	1	1	1	1	1	1	1
30.45056	1.181216	701.82	30	1	1	1	2014	1	1	1	1	1	1	1	1	1	1	1
34.41234	1.214970	764.78	32	1	1	2	2014	1	1	1	1	1	1	1	1	1	1	1
27.00152	1.227748	561.61	33	1	1	7	2014	1	1	1	1	1	1	1	1	1	1	1

Fig. 1. (d) The image shows the fourth 19 columns of the feature matrix using the non-imputed datasets.

AT	BT	CT	DT	ET	FT	GT	HT	IT	KT	LT	MT	NT	OT	PT	RT	ST	WT	XT
0.0001	0.0714	0.44048	0.1371	3.44400	0.11100	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	
-5.50000	-5.51774	1.30764	1.30764	1.304421	1.303953	13.0511	6.11100	14.04007	6.15305	5.530005	6.161573	204.0400	13.0000	157.99	36.451559	31.07000		
-0.00002	0.371207	0.000073	0.140506	1.421263	7.43000	0.132705	0.142200	0.137900	0.139900	0.142000	0.142100	0.142000	0.142000	0.142000	0.142000	0.142000	0.142000	
-0.1446	-0.15687	1.040719	0.00000	2.000000	0.200000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	

Fig. 1. (e) The image shows the fifth 19 columns of the feature matrix using the non-imputed datasets.

CR	CS	CT	CU	CV	CW	CX	CY	CZ	DA	DB	DC
SUM(sess1)SUM(sess1)SUM(sess1)SUM(sess1)SUM(sess1)SUM(sess1)MODE(tr NUM_UNIQUE(transactions.sessions.device)	86.49	6	-0.43368	-0.54531	256.2497	8.766791	mobile	2			
131.51	9	0.04521	1.553304	360.46448	10.81159	mobile	3				
107.93	8	0.195147	-2.47842	308.4267	10.35277	mobile	3				
66.21	6	2.303185	0.405896	252.616	9.023197	desktop	1				
179.18	7	-0.10022	-0.81987	246.1775	9.735302	tablet	3				

Fig. 1. (f) The image shows the last 8 columns of the feature matrix using the non-imputed datasets.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
5	00001	6	desktop	1	70	140.02	5	16.3140	0.107975	7.55	1	0.00000	0.00000	44.0000	1.47000	3017.71	70	28
4	00001	8	mobile	1	100	140.95	5	28.0000	0.000000	5.75	1	0.13300	0.13247	0.000000	1.01000	205.53	299	15
1	00002	8	mobile	3	120	139.43	5	67.3091	1.10902	5.81	1	0.10583	0.27220	40.0000	1.40000	771.42	274	47
3	12344	6	desktop	1	90	140.15	5	68.5000	2.877777	5.80	1	0.43213	0.00000	42.7220	1.40000	374.63	239	21
2	12344	7	desktop	3	80	140.81	5	75.0000	3.14600	8.73	1	0.34000	0.00000	36.6739	1.30000	420.00	303	18

Fig. 2. (a) The image shows the first 19 columns of the feature matrix using the imputed datasets.

T	U	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK	AL
17	7	5	5	1864	2092	18 88.16770	3.7	20.85	1 0.052329	0.050031	53.18723	1.849721	1587.02	50	13.10667			
8	9	4	1	4	2006	1011	18 110.41	3.5	14.81	2 0.540436	1.127907	55.10507	1.549013	1311.15	63	13.0175		
12	1	4	0	6	1994	2011	21 81.40612	4.75	20.04	1 0.04075	0.032644	45.4751	1.681261	1190.4	77	15.75		
13	11	8	0	5	2002	2011	18 76.09489	2.222223	20.06	1 0.0021	0.042612	32.72034	1.721301	130.09	58	13.2		
35	9	4	0	6	1996	2012	18 96.541	1.332230	56.46	2 0.93037	0.140018	52.1704	1.538018	1157.22	60	13.20372		

Fig. 2. (d) The image shows the fourth 19 columns of the feature matrix using the imputed datasets.

BY	RF	CA	CR	CC	CD	CF	CY	CG	CH	CI	CI	CC	CI	CM	CN	CD	CP	CO
0.3979	0.07181	0.50640	-0.71307	1.60056	7.41001	0 11.6507	0.040412	4.161414	0 0.07900	0.01364	108.313	0.04000	63.76	30 452.749	16.45130			
-0.54113	-0.57321	-0.53000	0.05194	3.357418	3.534421	0 16.0276	0.051553	16.1259	0.35351	0.09000	0.596279	18.9721	12.7034	157.99	39 468.5488	22.8956		
-0.78829	0.13791	1.22359	1.21203	4.96203	7.51580	0 11.8001	0.396153	6.254586	0 0.33536	0.53043	218.0423	13.1195	155.56	40 546.020	25.67581			
0.447388	0.556887	1.40536	2.420992	20.52584	0 8.0040	0.269439	5.444007	0 0.35326	0.324661	229.0424	9.50964	818.21	30 385.352	17.4541				
0.342938	0.238906	1.1200	0.444848	3.450129	17.21210	0 11.892	0.220479	15.8881	0 0.35021	0.302129	244.0546	11.4642	93.48	32 341.0897	21.8884			

Fig. 2. (e) The image shows the fifth 19 columns of the feature matrix using the imputed datasets.

CR	CS	CT	CU	CV	CW	CX	CY	CZ	DA	DB	DC
SUM(sess)	SUM(sess)	SUM(sess)	SUM(sess)	SUM(sess)	SUM(sess)	MODE(transactions.sessions.device)					
86.49	6	0.612029	-0.71509	260.8685	8.578982	desktop	3				
124.55	9	1.240846	1.807723	361.4346	10.61407	mobile	3				
83.61	8	0.894674	-2.76082	320.4617	11.39719	mobile	3				
66.21	6	2.993542	0.181136	253.8736	8.944419	desktop	1				
153.75	7	0.77951	-0.75144	265.9458	1.956618	desktop	3				

Fig. 2. (f) The image shows the last 8 columns of the feature matrix using the imputed datasets.

V. CONCLUSION AND RECOMMENDATION

The researchers implemented the mean, median, and mode imputation into DFS to handle missing and unnecessary data. It showed that imputing the missing and unnecessary values in a dataset will establish that there is a distinction between using imputation and not. Moreover, the imputation outcome is more evident if large datasets are utilized. Because the researchers identify further gains in this study, they recommend these essential directions for future research. First, the imputation method was performed on a separate file. Future work could concentrate on incorporating the code into the *AggregationPrimitive* file of the *featuretools* package. The present work still has no imputation methods for boolean data type. Use statistical analysis techniques if future researchers want to check the accuracy of the datasets. Finally, the imputation methods for float and int data types should be separated.

ACKNOWLEDGMENT

The researchers are thankful to the Almighty God for providing enough will and strength to make this study possible, to the professors of Computer Science Department at Pamantasan ng Lungsod ng Maynila for their help and guidance, and to their family and friends for their love, support, and company throughout the days of making this research paper.

REFERENCES

- [1]. Koerhsen, W. (2018, June 2). Automated Feature Engineering in Python. Retrieved from <https://towardsdatascience.com/automated-feature-engineering-in-python-99baf11cc219>
- [2]. Kanter, J. M. and Veeramachaneni, K., "Deep feature synthesis: Towards automating data science endeavors," *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, 2015, pp. 1-10, doi: 10.1109/DSAA.2015.7344858.

- [3]. Kanter, M. (2018, January 16). Deep Feature Synthesis: How Automated Feature Engineering Works. Retrieved from <https://www.linkedin.com/pulse/deep-feature-synthesis-how-engineering-automation-works-max-kanter>

- [4]. Patel, K. (2020, September 9). Image Feature Extraction: Traditional and Deep Learning Techniques. Retrieved from <https://towardsdatascience.com/image-feature-extraction-traditional-and-deep-learning-techniques-ccc059195d04>