Practical Tool to Understand Structure of Machine Learning Training Databases with Missing Data

Arthi Iyer 17arthii@students.harker.org

Junior, The Harker School, San Jose

Professor Dragutin Petkovic petkovic@sfsu.edu

Professor, SFSU CS Department

Technical report: San Francisco State University Computer Science TR-16.01

April 2016

ABSTRACT

A common problem in the field of machine learning (ML) today is that training databases often have missing data. ML training databases are most often organized as matrices, where rows constitute training samples, each with set of features or measures organized as columns, with last column being class label (ML class). In case of missing data, some values for features (e.g. feature values) might not be available due to various reasons. This substantially lowers the data's usability and accuracy in using most ML approaches. The problem how best to chose values to fill in for missing data (so called *imputing*) is well studied and is not the subject of this report. We instead address the problem that very little has been done to first provide a user with a tool to help him/her understand and asses the degree of missing values in the given training database and the impact to availability of data before he/she makes a decision what to do in terms of imputing. Specifically, we address the following questions: a) what features (measures) have what amount of missing data; b) what subsets of features (measures) yields what subset of training samples with *no missing* data. We believe that such knowledge is important as a starting step in deciding imputing strategy. The Missing Data Analyzer (MDA) SW tool we developed addresses this problem in two steps: a) it helps ML researchers understand statistics and impact of missing data and tradeoffs between using subset of features vs. available subset of training samples with no missing data; and b) it creates a *filtered training DB* with chosen subset of feature values yielding corresponding training samples with no missing data. In this report we present MDA algorithm and SW, including test sample and sample of output. MDA SW is posted on github repository

1. INTRODUCTION

In all Machine learning (ML) applications, the critical element is the high quality (e.g. accuracy, number of samples) of the training database that is used to train ML algorithms. ML training databases are most often organized as matrices, where rows constitute training samples, each with set of features or measures organized as columns, with last column being class label (ML class). In case of missing data, some values for features (e.g. feature values) might not be available due to various reasons. Some common issues related to quality of training database include not only data accuracy, noise in the data, adequate number of training samples, proper mix of positive and negative samples, but also missing data, which is the focus of this work. These missing data are often a result of data simply not being available due to many reasons and their distribution is often random. Such missing values pose severe problems in ML analysis:

a) A large majority of commonly used ML algorithms do not deal well or not at all with missing data;
b) Most very large training databases with randomly (or unknown to the user) missing feature value distribution pose a challenge for ML practitioner in deciding whether to recreate or impute some values, what features to drop, or what subset of feature values to use to get desired number (subset) of training samples. In addition, checking class mix of each chosen subset of training samples is important to ensure proper ML training.

The issue of best algorithms to impute missing values (e.g. how to compute them) has been extensively studied and is not the subject of this work (see for example [1, 2]). Most work on imputing focuses on various methods on replacing missing values based on statistics of remaining samples in the particular feature value, and is evaluated with respect to specific ML methods and applications. Methods have been proposed ranging from complex ones (based on ML and statistical analysis) to simpler ones (e.g. using mean, median of existing feature values). For practical purposes, it is assumed that number of missing values is at most several percents of the number of values in that feature. However, to the best of our knowledge, there is no simple tool to let users get an idea in the first place of a) the structure and distribution of missing values

in the training database and b) tradeoff information about choosing subset of features vs. available subset of training samples which need no imputing for the chosen feature subset.

When faced with large training DB with missing values users will likely ask questions like:

- What features have how much missing data?
- Can I drop some incomplete features and still get enough training samples with right class mix to do my ML analysis?
- My ML approach needs only original measures and cannot deal with missing data or imputed data (for whatever reason), so what are my choices.
- I really want to explore power of a specific features, but they have some missing data, so if I chose to keep them what is the subset of training samples with complete (no missing) data I have available?
- How many complete training samples (no missing data) are there if I use all features?
- I need to create a filtered DB with my choice of feature subset and only training samples with no missing data

Our work helps user answer these questions.

We do not make any assumptions of the distribution of missing values in the training database, importance of specific features, nor the influence of noise in existing feature values. We also do not address the issue of actual missing value replacement (imputing) since this is a well-covered subject in the literature is highly application/domain dependent.

The main contribution of our work is a Missing Data Analyzer (MDA) SW tool that accomplishes two steps: a) it analyses the training database and first produces data (e.g. lists, tables) which helps user understand the structure of training database with respect to missing values and the tradeoffs between feature subsets and complete training samples as described above; and b) Upon user selection of best option (e.g. feature subsets vs. available subset of training samples with no missing data) the tool extracts desired subset of complete training samples (with related FVi subset) into a new *filtered training DB* which does not need imputing and is ready for further ML processing.

We tested our MDA tool on a set of synthetic examples (one is presented in this paper).

The rest of the report is organized as follows. In section 2, we formalize the problem and show a typical structure of training database with missing data and MDA functionality, using a simple example. We then outline the tradeoffs the user faces and data used to help in this decision. Section 3 contains a description the MDA algorithm in high level pseudo code, synthetic example and its related MDA output as well as usage flow, explaining how the tool and its software implementation can be used to better understand the training database with respect to missing values. Section 4 outlines status and future work.

2. PROBLEM DEFINITION

We first overview the typical structure of a training database with missing data and then, based on a simple example, we outline our approach and the choices the user faces. General ML training database contains N_s rows, each row representing a training sample. Each training sample has a unique identifier. Associated with this identifier, in the given row/sample, is a set of N_F features or measures, FV_i , i = 1 … N_F (e.g. each FV_i is some test or measurement) arranged as a feature vector, finally followed by class label or identification CL (usually there are two class labels like positive/negative, 1/0, or event present or not). FV_i can be *numerical* (representing a continuum of values like from measuring level of some variable) or *categorical* (values from a finite set like gender, city, some category). Table I presents this structure. The fact that some feature vectors may be noisy is not the subject of this analysis. Lastly, in most cases N_s is far greater than N_F.

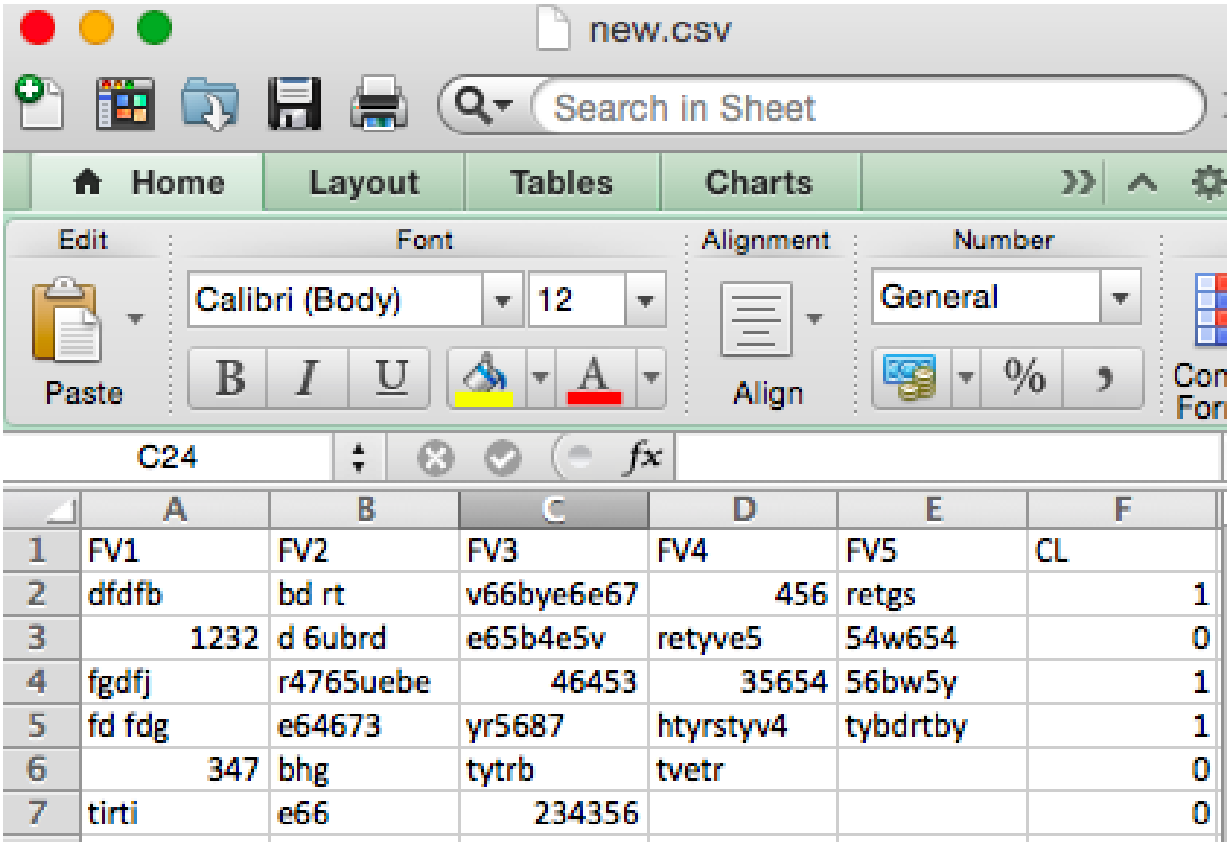
Specifically, our work and the MDA tool we built is answering the following questions:

a) How are missing values distributed vs. features in training database? To help here we rank the features FVi by the amount of missing data and show number and % of missing data for each feature FVi.
b) How many *complete* training samples (no missing feature values) are available for various subsets of FVi? What is resulting class label mix for this subset of training samples (to ensure proper mix for ML training)? Note that this problem has many choices - for example one could chose smaller number of complete (no missing data) training samples but with larger set of feature available, or vice versa. To help user make

these choices and tradeoffs we provide a chart listing various subsets of FVis (later we outlines specific ways we create these subsets) and for each provide number of complete (no missing data) training samples and related class mix (e.g. number of classes labeled 1).

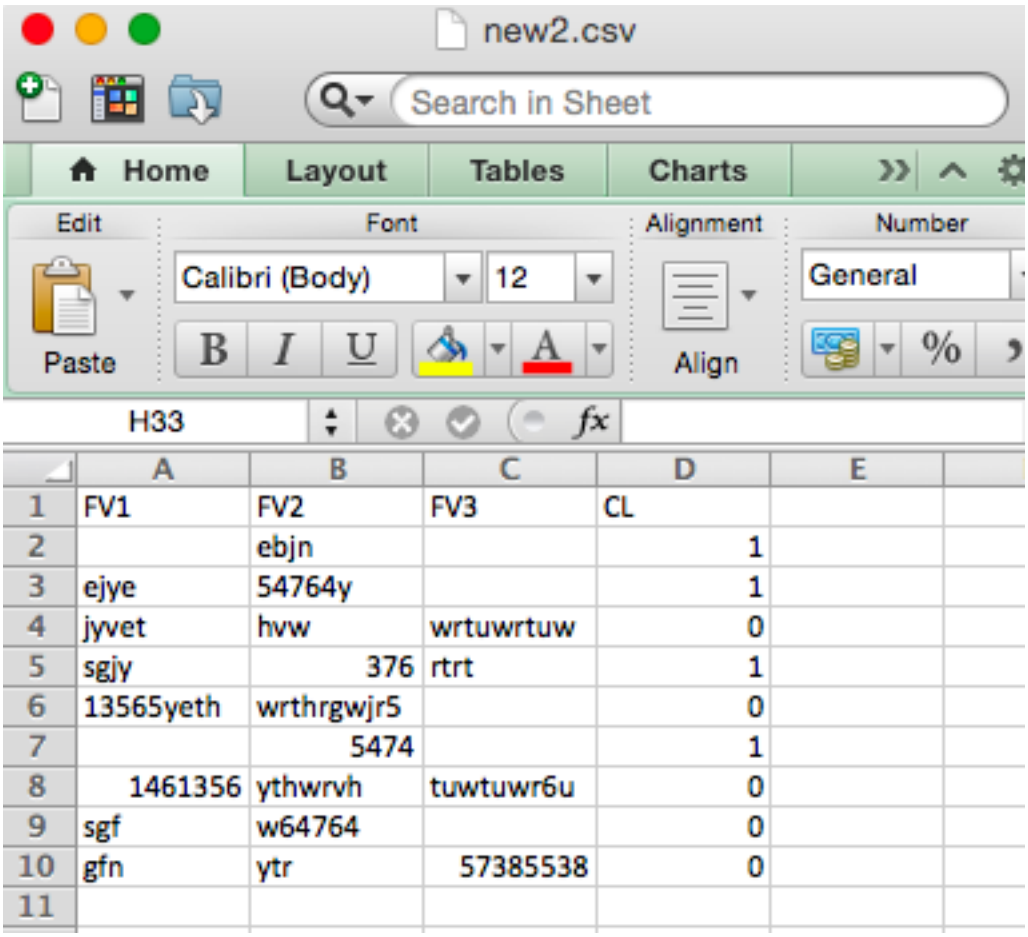
c) Upon user selection of given feature subset FVi, MDA tool generated filtered DB with only that FVi subset and only samples that are complete (no missing data)

Below, in Tables I and II we show two synthetic examples to illustrate how MDA works.



	A	B	C	D	E	F
1	FV1	FV2	FV3	FV4	FV5	CL
2	dfdfb	bd rt	v66bye6e67	456	retgs	1
3	1232	d 6ubrd	e65b4e5v	retyve5	54w654	0
4	fgdfj	r4765uebe	46453	35654	56bw5y	1
5	fd fdg	e64673	yr5687	htyrstyv4	tybdrtby	1
6	347	bhg	tytrb	tvetr		0
7	tirti	e66	234356			0

Table I – Example 1: typical structure of training ML database with missing values (synthetic example). Missing values are denoted by empty cells. In this case Ns=6 and Nf=5



	A	B	C	D	E
1	FV1	FV2	FV3	CL	
2		ebjn		1	
3	ejye	54764y		1	
4	jyvet	hvw	wrtuwrtuw	0	
5	sgjy	376	rtrt	1	
6	13565yeth	wrthrgwjr5		0	
7		5474		1	
8	1461356	ythwrvh	tuwtuwr6u	0	
9	sgf	w64764		0	
10	gfn	ytr	57385538	0	
11					

Table II – Example 2: Typical structure of training ML database with missing values (synthetic example). Missing values are denoted by empty cells. In this case Ns=9 and Nf=3

One can immediately note that several choices and trade-offs in dealing with missing values are available (the observations below are based on Example 1 in Table I):

- The user should first get an idea of the level of missing data for each feature value. From the example in Table I, the user can conclude that FV1, FV_2, FV3 are complete (no missing values) followed by FV4 one missing value, then FV5 with 2 missing values.
- The user could decide to not use any imputation and use only training samples which have all the data (this can happen in many combinations of subsets of training samples and subsets of FVi). The user would also want to know how many training samples and what class mix is available with each such choice of FVi subset to ensure proper ML training. In our example from Table I, user can observe that using three features with no missing data (FV1, FV2, FV3) produces total of 6 training complete samples with a class

mix of 3 ones and 3 zero class labels. Problem with this choice is that all other FVis are discarded which may not be what the user wants.

- To increase number of FVi for the training database, the user can explore further tradeoffs between set of FV_i, the available number of samples, and the class mix. In the examples below, we can keep adding FV_i to the subset, starting with those FVis with smallest missing values and so on:
    - Adding FV4 to previous subset FV1, FV2, FV3 yields 5 complete samples with class mix of 3 ones and 2 zero class labels
    - Adding FV5 (next most complete FVi) to the above subset produces 4 samples with class mix of 3 ones and 1
- MDA SW provides the above data in easy to understand form e.g. list of subsets of FVi (constructed as above by adding FVi with least missing values, on by one) with number of complete training samples and class mix for each (see Table III). Note that this data has to be provided by actual analysis of the actual training database being used, since we make no assumptions on the distribution of missing values. MDA will then allow user to extract chosen subset of training data into a new training database with no missing data but with subset of training samples for a chosen subset of FVis (see Image I for user dialog screen). The filtered database for Example 1 is shown in Table IV. This filtered DB, due to MDA algorithm implementation, will have FVi rearranged from the one with least missing data to ones with more missing data and so on.
- Finally, the question of how many complete training samples are there when ALL features are used is answered by the last entry in the list in Table III (it is 4)

Results and filtered DB for Example 2 in Table II are given in Image II, Table V and Table VI respectively.

3. ALGORITHM AND SOFTWARE

We outline high level pseudo code of MDA and then we discus some issues related to algorithm complexity and efficiency. We close this section with the MDA flow from users' point of view as well as with an example of program user dialog.

3.1 Pseudo code of MDA algorithm:

INPUT:

A ML training database (*Input_DB*) containing N_s rows, each row representing a training sample. Each training sample has a unique identifier. Associated with this identifier, in the given row/sample, is a set of N_F feature values or measures, FV_i , i = 1 … N_F (e.g. each FV_i is some test or measurement) arranged as a feature vector, finally followed by class label or identification CL (usually there are two class labels like positive/negative, 1/0, or event present or not). In sujmary, this Input_DB is arranged as rows and columns, where each row is a training sample, each column is representing FVi, and last column is CL. First (top) row contains FVi names.

OUTPUT:

MDA Results table showing a) number of complete (no missing data) for selected subset of FVi, and b) class mix (number of training samples labeled with CL=1) for the given set in a).

Upon user selection of FVi subset based on choices/tradeoffs between subset of FVi and number of available training samples with no missing data for that subset, MDA generates "*filtered DB*" which contains only chosen subset FVi with only complete (no missing data) training samples and their original CL. This filtered DB is then ready to be used in ML analysis where user does not want any missing data.

Step1: find basic info on mssing data for each FVi

Sort FVis by amount of missing data, in ascending order, in a list *SORTEDFVI* (first item has least missing data)

Step 2: rearrange original Input_DB for simplicity in coding

Rearrange original Inpout_DB such that FVis (columns) are now ordered from the fist in the above sorted set (the one with least missing data and so on – e.g. they are arranged from left to right as per ascending order of missing data.

Step 3: Compute MDA Results table

Set *SUBSET_FV* = empty; /* denotes subset of FVis to be used, which will be done by adding successive FVis from SORTEDFVI list*/

Repeat for all FVis in SORTEDFVI list in ascending order

Add next FVi from SORTEDFVI list to SUBSET_FV

For all training samples in Input_DB AND only for FVis in SUBSET_FV compute

NUM_Complete _Samples = number of training samples which are complete (no missing values);

Class_Mix = number of above complete training samples with CL=1;

OUPUT MDA Results Table: current list of FVis in SUBSET_FV with related *NUM_Complete _Samples*; Class_Mix

Step 4: Compute Filtered DB for chosen subset of FVi

User selects subset of FVi denoted *Selected_FVi*

Set *Filtered_DB* to empty;

For all FVis in user selected set Selected_FVi

For all training samples in Input_DB

IF training data sample has no missing elements for all Selecetd_ FVi

Add this training sample and its CL to FilteredDB

3.2 MDA algorithm and software implementation details:

In our algorithm, as outline above, in testing what training samples are complete for subset of FVis, we add FVis one by one to the subset, from those with least missing data. This is empirical approach and a practical one, with low complexity. To get best possible combinations of FVi subset vs. available complete training samples, one would have to explore all FVi subsets of size K (K=1…Nf) which has exponential complexity (2**Nf) which is computationally prohibitive.

In terms of SW implementation, there are 4 classes in the program, with one main class, called FVSelection for which the pseudo code of the main portion is available below. Two of the

classes are used to help read files and write to them. These classes average at around 70 lines of code. There is one class that is dedicated to FeatureVector objects that uses about 100 lines of code. FVSelection results in 786 lines of code. All code in this program is written in Java.

In terms of SW engineering methodology we applied three modern methods: a) test driven development where we would first create test data like examples above and use them to design algorithms and SW and test it; b) User Centered Design where we incorporated ML user needs in terms of program output and flow; and finally c) optimization toward code simplicity and maintenance and not code efficiency (e.g. rearranging of Input DB in Step 2).

3.3 MDA flow from users' point of view.

First Input

The user inputs the file path name to the training database that they would like the product to work on.

First Output

Output will be written to the same directory as the training database with the describing words appended to it. The output will always be a .csv file. For example, if the inputted file path name for the training database is "/Users/User/Documents/TB.csv", then the output file will be found at "/Users/User/Documents/TB_results.csv". This output will provide numerous statistics on the missing data of the database, including amount, distribution, and concentrations of it.

Second Input

There will be a second input that the user can put in that allows the program to create another .csv file that contains a subset of the data that the machine learner deems optimal based on the statistics provided in the first output.

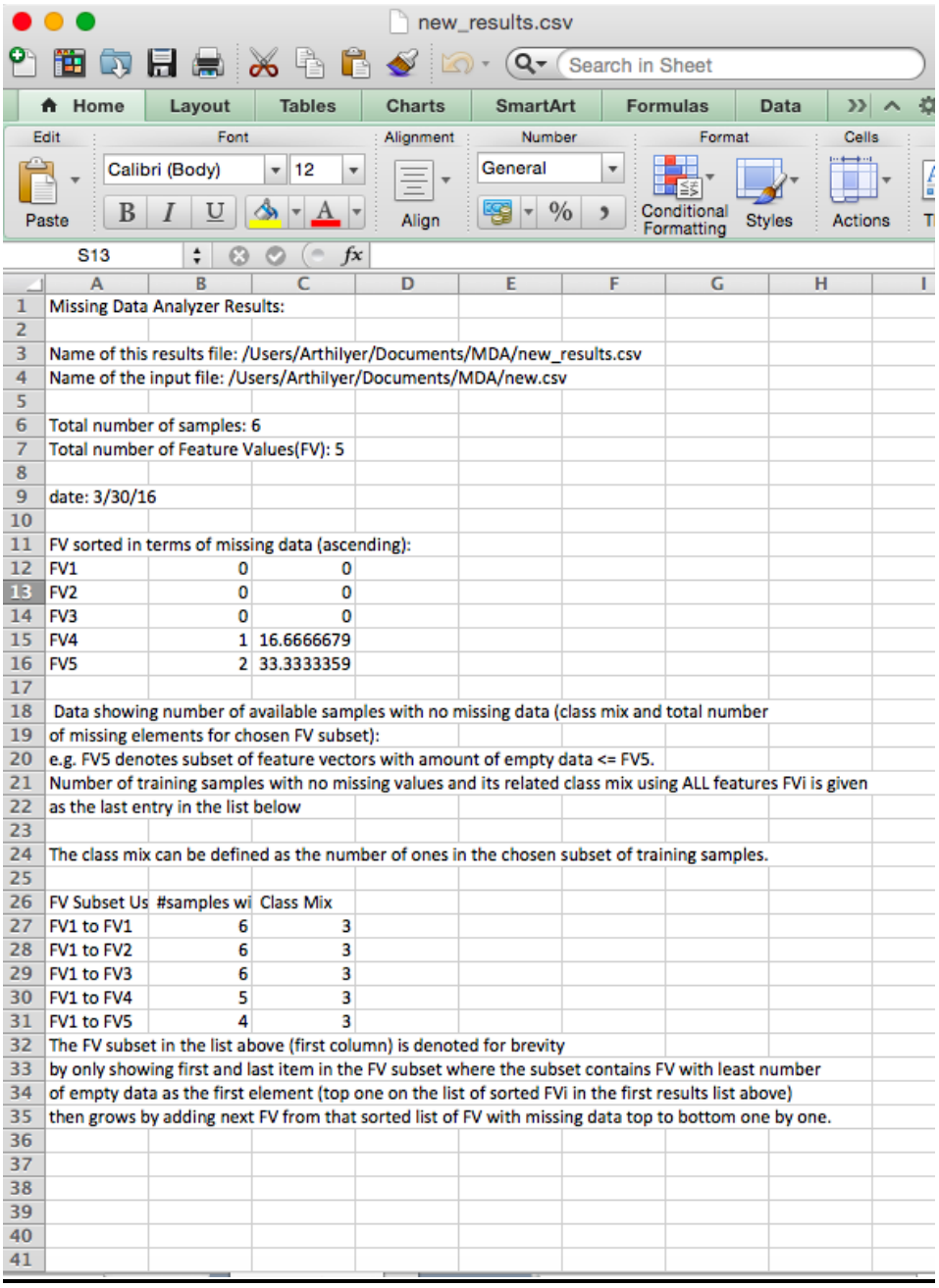
Second Output

Similarly, the second file output that contains the subset of the data will be able to be found at "/Users/User/Documents/TB_subset_matrix.csv".

3.4 MDA Program Output Example 1

```
What is the date that you would like to be reflected in the program's output?
3/29/16
Would you like to input the .csv file path name of your training database?
If not, type no now. Otherwise, input the path name now.
/Users/ArthiIyer/Documents/MDA/new.csv
A results file containing statistics on your TB can now be found at:

/Users/ArthiIyer/Documents/MDA/new_results.csv


If you would like to quit now, type 'quit'.

Otherwise, type the name of the FV that your subset should go up to.
To figure out which FV to use, please refer to results file.
After you input the FV, the program will create a .csv file containing
a subset of the training database. The subset will involve no imputing.

FV4

A database containing only the subset of data you requested can now be found

/Users/ArthiIyer/Documents/MDA/new_subset_database_FV4.csv
```

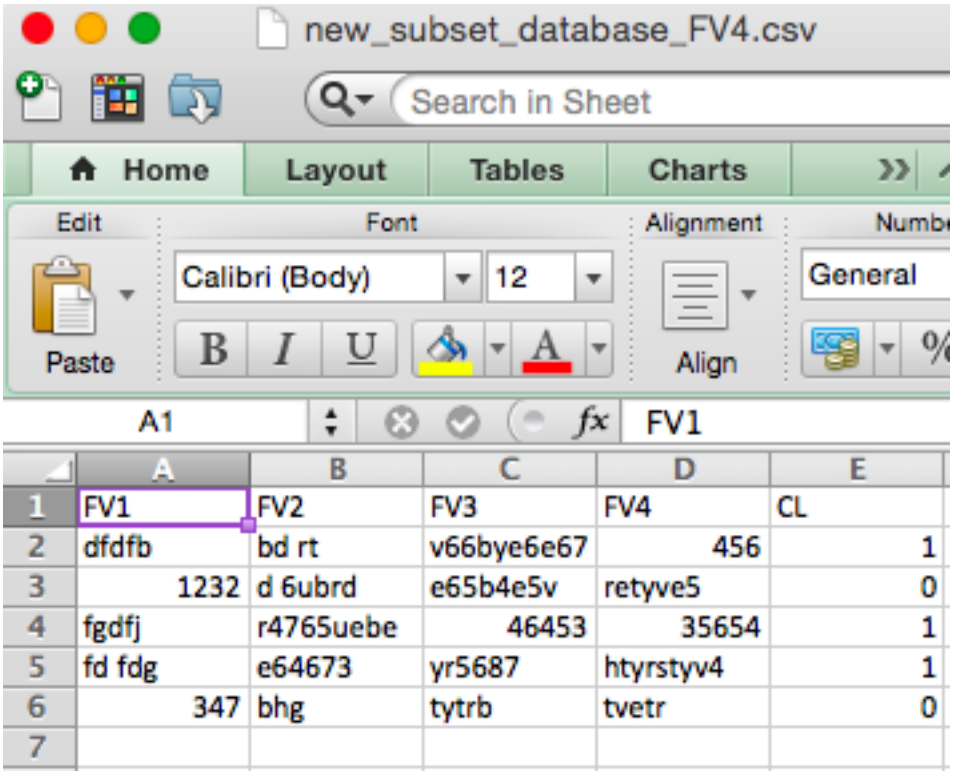
Image I – Typical terminal window view using test sample from Table I (synthetic example 1). Related results file is shown in Table III



	A	B	C	D	E	F	G	H	I
1	Missing Data Analyzer Results:								
2									
3	Name of this results file: /Users/Arthilyer/Documents/MDA/new_results.csv								
4	Name of the input file: /Users/Arthilyer/Documents/MDA/new.csv								
5									
6	Total number of samples: 6								
7	Total number of Feature Values(FV): 5								
8									
9	date: 3/30/16								
10									
11	FV sorted in terms of missing data (ascending):								
12	FV1	0	0						
13	FV2	0	0						
14	FV3	0	0						
15	FV4	1	16.6666679						
16	FV5	2	33.3333359						
17									
18	Data showing number of available samples with no missing data (class mix and total number								
19	of missing elements for chosen FV subset):								
20	e.g. FV5 denotes subset of feature vectors with amount of empty data <= FV5.								
21	Number of training samples with no missing values and its related class mix using ALL features FVi is given								
22	as the last entry in the list below								
23									
24	The class mix can be defined as the number of ones in the chosen subset of training samples.								
25									
26	FV Subset Us	#samples wi	Class Mix						
27	FV1 to FV1	6	3						
28	FV1 to FV2	6	3						
29	FV1 to FV3	6	3						
30	FV1 to FV4	5	3						
31	FV1 to FV5	4	3						
32	The FV subset in the list above (first column) is denoted for brevity								
33	by only showing first and last item in the FV subset where the subset contains FV with least number								
34	of empty data as the first element (top one on the list of sorted FVi in the first results list above)								
35	then grows by adding next FV from that sorted list of FV with missing data top to bottom one by one.								
36									
37									
38									
39									
40									
41									

Table III – Typical MDA output for the training data in Table I (synthetic example 1).

Note on interpreting the output as it related to feature subsets (Table III, rows 27-31, column A) Due to formatting we only show first and last FVi in each subset. Hence "FV1 to FV4" denotes "FV1, FV2, FV3, FV4" subset, namely all features in the sorted list starting from FV1 up to and including FV4.



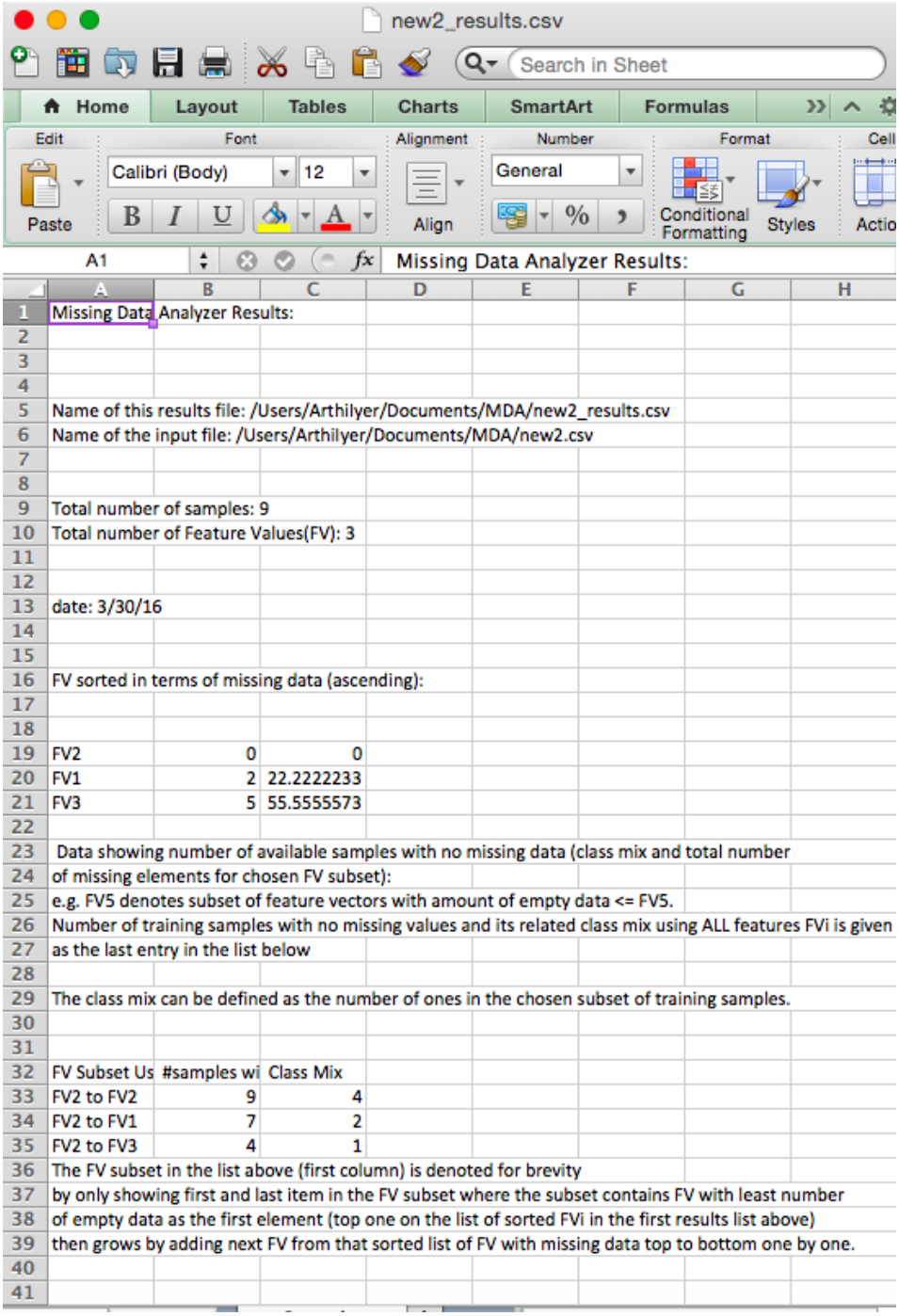
	A	B	C	D	E
1	FV1	FV2	FV3	FV4	CL
2	dfdfb	bd rt	v66bye6e67	456	1
3	1232	d 6ubrd	e65b4e5v	retyve5	0
4	fgdfj	r4765uebe	46453	35654	1
5	fd fdg	e64673	yr5687	htyrstyv4	1
6	347	bhg	tytrb	tvetr	0
7					

Table IV – Output training database (filtered) for example 1 in Table I, where user chose all FVi up to FV4

3.5 MDA Program Output Example 2

```
What is the date that you would like to be reflected in the program's output?
3/29/16
Would you like to input the .csv file path name of your training database?
If not, type no now. Otherwise, input the path name now.
/Users/ArthiIyer/Documents/MDA/new2.csv
A results file containing statistics on your TB can now be found at:

/Users/ArthiIyer/Documents/MDA/new2_results.csv


If you would like to quit now, type 'quit'.

Otherwise, type the name of the FV that your subset should go up to.
To figure out which FV to use, please refer to results file.
After you input the FV, the program will create a .csv file containing
a subset of the training database. The subset will involve no imputing.

FV3

A database containing only the subset of data you requested can now be found

/Users/ArthiIyer/Documents/MDA/new2_subset_database_FV3.csv
```

Image II - Typical terminal window view using test sample for example 2 from Table II (synthetic example). Related results file with statists is in Table V.

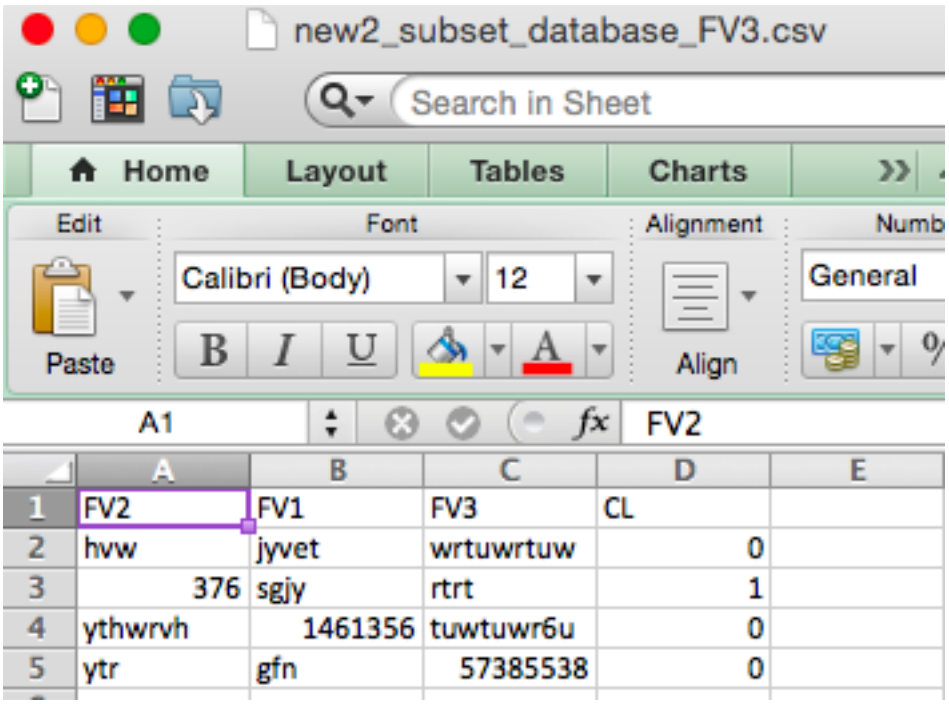


	A	B	C	D	E	F	G	H
1	Missing Data Analyzer Results:							
2								
3								
4								
5	Name of this results file: /Users/Arthilyer/Documents/MDA/new2_results.csv							
6	Name of the input file: /Users/Arthilyer/Documents/MDA/new2.csv							
7								
8								
9	Total number of samples: 9							
10	Total number of Feature Values(FV): 3							
11								
12								
13	date: 3/30/16							
14								
15								
16	FV sorted in terms of missing data (ascending):							
17								
18								
19	FV2	0	0					
20	FV1	2	22.2222233					
21	FV3	5	55.5555573					
22								
23	Data showing number of available samples with no missing data (class mix and total number							
24	of missing elements for chosen FV subset):							
25	e.g. FV5 denotes subset of feature vectors with amount of empty data <= FV5.							
26	Number of training samples with no missing values and its related class mix using ALL features FVi is given							
27	as the last entry in the list below							
28								
29	The class mix can be defined as the number of ones in the chosen subset of training samples.							
30								
31								
32	FV Subset Us	#samples wi	Class Mix					
33	FV2 to FV2	9	4					
34	FV2 to FV1	7	2					
35	FV2 to FV3	4	1					
36	The FV subset in the list above (first column) is denoted for brevity							
37	by only showing first and last item in the FV subset where the subset contains FV with least number							
38	of empty data as the first element (top one on the list of sorted FVi in the first results list above)							
39	then grows by adding next FV from that sorted list of FV with missing data top to bottom one by one.							
40								
41								

Table V – Typical MDA output for the training data for example 2 in Table II (synthetic example).

Note on interpreting the output as it related to feature subsets (Table V, rows 25-27, column A) Due to formatting we only show first and last FVi in each subset. Hence “FV2 to FV3” denotes

"FV2, FV1, FV3" subset, namely all features in the sorted list starting from FV2 up to and including FV3.



	A	B	C	D	E
1	FV2	FV1	FV3	CL	
2	hvw	jyvet	wrtuwrtuw	0	
3	376	sgjy	rtrt	1	
4	ythwrvh	1461356	tuwtuwr6u	0	
5	ytr	gfn	57385538	0	

Table VI – Output training database (filtered) for example 2 in Table II , where user chose all FVis up to FV3. Note that filtered DB has FVi rearranged from most complete t less complete.

4. STATUS AND FUTURE WORK

Our MDA tool is posted on github https://github.com/arthiyer/Missing-Data-Analyzer [3] and has open source license.

Future work includes:

-Improving user interface and providing graphical user interface for making tradeoffs

-Adding a simple "pin-down" function where user can chose a specific set of FVi and get information of available training subset with class label statistics only for that specific feature

subset [4]. This might be useful in case user wants to explore predictive power of specific features, where they might have missing values.

5. REFERENCES

1. T. D. Pigott: “A Review of Methods for Missing Data”, Educational Research and Evaluation , 2001, Vol. 7, No. 4, pp. 353±383
2. J. Kaiser:” Dealing with Missing Values in Data”, JOURNAL OF SYSTEMS Integration, 2014/1
3. Github repository link https://github.com/arthiyer/Missing-Data-Analyzer
4. Personal communication with Dr, Les Kobzik, MD, Stanford, 03-28-16