Assignment 3: Missing Data Tutorial

Low, Daniel Mark (S3120155)
Petre, Bogdan (S3480941)
Xu, Teng Andrea (S3548120)

Group 7

September 24, 2017

Introduction and Motivation

Missing data is everywhere. You are likely to find it in any dataset you want to analyze. Why is missing data a problem? One of the main issues is obtaining biased results from statistical tests such as correlation. For instance, perhaps two variables would not correlate as much if the missing data was available. This is a tutorial covering the basic and more advanced methods for dealing with missing data. We provide all code in a jupyter notebook available at this link:

https://github.com/bogdanp05/IDS_Assignment2

First, we will explain the different types of missingness you might encounter with examples of each. Then, we will provide basic methods for handling missing data, including discarding data and single value imputation with implementations in Python. Finally, we will review more advanced methods, such as multiple value imputation with implementations in R so we can end by comparing both languages in handling missing data. These methods are dealing with a problem, which is that somehow data was lost or never obtained. Therefore, the first rule of dealing with missing data is to avoid missing data in the first place. Extra care should be taken during data collection to avoid missingness as much as possible.

1 Relevant bibliography

1. Gelman and Hill (2007) [2]. Comment: We used Chapter 25 of this book which covers most of what we included in our tutorial.
2. Azur et al. (2011) [1]. Comment: This paper reviews the theory behind Multiple Imputation.
3. Therese D. Pigott (2001) [4]. Comment: Deepening the differences between Multiple Imputation and Single Imputation.
4. Marina Soley-Bori (2013) [3]. Comment: Chapter 4 of the paper explains the most used techniques in order to deal with missing data.

2 Types of missing data

There are four main types of missing data that occur in data sets [2].

Missingness completely at random is when there is the same chance of having missing values for all units. You would end up with the same statistical results with the true data as with this type of missing data.
Example 1: If there is a survey where the participants have to regularly submit some data, we could have missing values because sometimes people simply forget to submit.
Example 2: We could have an old data set (like the Titanic data set) in which some of the values got damaged beyond recognition (e.g., ink wearing off).

Missingness at random is when the chance of a variable to be missing depends only on other, fully recorded variables.

Example: In a survey where age and education are fully recorded, we could say that the *income* variable is missing at random if the chance of missingness depends only on these other variables, which have no missing values.

Missingness that depends on unobserved predictors is when the chance of a recorded variable to have missing values depends on variables that are not recorded. This case is similar to the *missingness at random* case, with the exception that the variables indicative of the variable of interest are not recorded.
Example: Imagine a survey where a variable is represented by the answer to a complicated question. Non native speakers might have trouble understanding the question and may therefore not respond to it. If the country of origin is not recorded, then the missingness depends on unobserved predictors.

Missingness that depends on the missing value itself is when the chance of a variable to be missing depends on the value of that variable.
Example 1: A classic example is that people with higher incomes tend to not state their income.
Example 2: A more recent example of this phenomena was during the US presidential election of 2016, when, during the polls before the election, a lot of Donald Trump supporters declined to answer out of social embarrassment.

Question: Which of these types of missing data is the most desirable?
Answer: The ideal case is that of Missingness Completely at Random (the types of missing data are actually ordered from best to worst case scenario) because the variable has the same probability to be missing for all the records variables. A good way to deal with it would be to simply remove the rows entirely, since this will not bias the results, which would lead to what is called Complete-Case Analysis. However, in most cases it is impossible to prove the complete random nature of a missing variable. What data scientists try to do instead is to create a model for this variable and to include as many observed predictors as possible (e.g., create a model for income that takes into account age, gender, education, region).

3 Visualizing your (missing) data

For this tutorial, we will use the Titanic dataset, a widely used dataset in Data Science. It has information on 1300 passengers of the Titanic, including their names, age, if they survived, where they were heading, and other relevant information. We will use Pandas, which is a Python library for easily manipulating dataframes. We will also use matplotlib, which is for plotting visualizations. The first step is to import the libraries and the Titanic dataset. Then we are going to print the first 20 rows using the DataFrame.head(n) method. Pandas prints the last columns below.

Preprocess missing data using Pandas for Python

```
import pandas as pd
import matplotlib.pyplot as plt
```

```
df = pd.read_csv("titanic_original.csv")
print(df.head(20))
```

```
    pclass  survived                                             name  \
0      1.0       1.0                    Allen, Miss. Elisabeth Walton
1      1.0       1.0                   Allison, Master. Hudson Trevor
2      1.0       0.0                     Allison, Miss. Helen Loraine
3      1.0       0.0             Allison, Mr. Hudson Joshua Creighton
4      1.0       0.0  Allison, Mrs. Hudson J C (Bessie Waldo Daniels)
5      1.0       1.0                          Anderson, Mr. Harry
6      1.0       1.0                Andrews, Miss. Kornelia Theodosia
7      1.0       0.0                           Andrews, Mr. Thomas Jr
8      1.0       1.0    Appleton, Mrs. Edward Dale (Charlotte Lamson)
9      1.0       0.0                         Artagaveytia, Mr. Ramon
10     1.0       0.0                          Astor, Col. John Jacob
11     1.0       1.0  Astor, Mrs. John Jacob (Madeleine Talmadge Force)
12     1.0       1.0                   Aubart, Mme. Leontine Pauline
13     1.0       1.0                    Barber, Miss. Ellen "Nellie"
14     1.0       1.0             Barkworth, Mr. Algernon Henry Wilson
15     1.0       0.0                           Baumann, Mr. John D
16     1.0       0.0                        Baxter, Mr. Quigg Edmond
17     1.0       1.0  Baxter, Mrs. James (Helene DeLaudeniere Chaput)
18     1.0       1.0                      Bazzani, Miss. Albina
19     1.0       0.0                       Beattie, Mr. Thomson

       sex      age  sibsp  parch    ticket      fare    cabin embarked boat  \
0   female  29.0000    0.0    0.0     24160  211.3375       B5        S    2
1     male   0.9167    1.0    2.0    113781  151.5500  C22 C26        S   11
2   female   2.0000    1.0    2.0    113781  151.5500  C22 C26        S  NaN
3     male  30.0000    1.0    2.0    113781  151.5500  C22 C26        S  NaN
4   female  25.0000    1.0    2.0    113781  151.5500  C22 C26        S  NaN
5     male  48.0000    0.0    0.0     19952   26.5500      E12        S    3
6   female  63.0000    1.0    0.0     13502   77.9583       D7        S   10
7     male  39.0000    0.0    0.0    112050    0.0000      A36        S  NaN
8   female  53.0000    2.0    0.0     11769   51.4792     C101        S    D
9     male  71.0000    0.0    0.0  PC 17609   49.5042      NaN        C  NaN
10    male  47.0000    1.0    0.0  PC 17757  227.5250  C62 C64        C  NaN
11  female  18.0000    1.0    0.0  PC 17757  227.5250  C62 C64        C    4
12  female  24.0000    0.0    0.0  PC 17477   69.3000       B35        C    9
13  female  26.0000    0.0    0.0     19877   78.8500       NaN        S    6
14    male  80.0000    0.0    0.0     27042   30.0000       A23        S    B
15    male      NaN    0.0    0.0  PC 17318   25.9250       NaN        S  NaN
16    male  24.0000    0.0    1.0  PC 17558  247.5208  B58 B60        C  NaN
17  female  50.0000    0.0    1.0  PC 17558  247.5208  B58 B60        C    6
18  female  32.0000    0.0    0.0     11813   76.2917       D15        C    8
19    male  36.0000    0.0    0.0     13050   75.2417        C6        C    A
```

Figure 1: Importing the libraries and the Titanic dataset

Then we use a Python dictionary to make a bar plot. As we can see, variable Age has over 200 missing data.

Visualize missing data

```
# Dictionary with amount of NaN per column

columns = df.columns
D = {}
for i in columns:
    D[str(i)] = df[i].isnull().sum()
print(D)
```

```
{'pclass': 1, 'survived': 1, 'name': 1, 'sex': 1, 'age': 264, 'sibsp': 1, 'parch': 1, 'ticket': 1, 'fare': 2, 'cabi
n': 1015, 'embarked': 3, 'boat': 824, 'body': 1189, 'home.dest': 565}
```

Figure 2: Python dictionary

```
# Plot missing values per column

plt.bar(range(len(D)), D.values(), align='center')
plt.xticks(range(len(D)), D.keys(),rotation='vertical')
plt.title('Missing values per column (out of 1310 rows)')
plt.show()
```

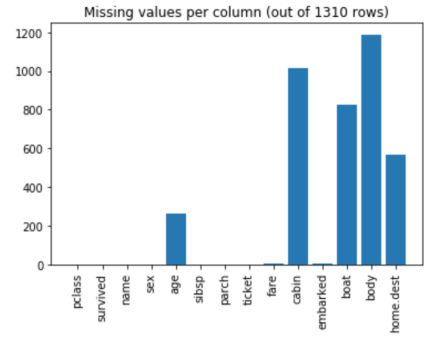

Figure 3: Bar plot of missing data per variable

Now we can start looking at possible solutions.

4 Discarding missing data

One solution is to discard every row where there is at least one missing value. This is the complete-case analysis mentioned before.

Basic solutions

```
# Complete-case analysis: Drop rows with a NaN value

df2 = df.dropna(axis=0, how='any')
print(df2.head(10))
```

```
Empty DataFrame
Columns: [pclass, survived, name, sex, age, sibsp, parch, ticket, fare, cabin, embarked, boat, body, home.dest]
Index: []
```

Figure 4: Complete-case analysis: discard rows with missing data

The result is an empty dataframe! All rows in this dataset have at least one missing value in a given variable.
Question: When can I use complete-case analysis work?
Answer: When the records with missing values do not differ dramatically from the ones with recorded values and when the number of rows to be dropped is relatively small (e.g., 5%).

Single value imputation

Another solution is to perform an imputation or fill-in a missing value with another value. Which value should be used? The mean of the variable is commonly used. Row 15 was NaN before and now it is filled.
Just to show that it worked, we can plot the amount of missing data per variable again and see that Age dropped to zero missing values.
The problem with imputing the mean is that the mean is not always a good approximation. It does not really make sense to replace the age of a passenger with 29.8. It also has the drawback of reducing the correlations between variables. Furthermore, it underestimates the true variance, which is used to compute statistical tests. This will result in biased data. We can compute the standard deviation to demonstrate this.

Question: When can I use single imputation with the mean?
Answer: If there is a variable with low variance then the mean will be a good approximation. For example, if there is a reaction time experiment the require a button press, and some instances

Single value imputation/filling

```
# Fill NaNs with mean of the column (only numerical data)
mean_age = df['age'].mean()
print(mean_age)

df_mean = df.fillna({'age': mean_age})
print(df_mean[['age']].head(20))
```

```
29.8811345124
         age
0   29.000000
1    0.916700
2    2.000000
3   30.000000
4   25.000000
5   48.000000
6   63.000000
7   39.000000
8   53.000000
9   71.000000
10  47.000000
11  18.000000
12  24.000000
13  26.000000
14  80.000000
15  29.881135
16  24.000000
17  50.000000
18  32.000000
19  36.000000
```

Figure 5: Replacing missing data with the mean

```
# Dictionary with amount of NaN per column
columns = df_mean.columns
D1 = {}
for i in columns:
    D1[str(i)] = df_mean[i].isnull().sum()

# Plot missing values per column
plt.bar(range(len(D1)), D1.values(), align='center')
plt.xticks(range(len(D1)), D1.keys(),rotation='vertical')
plt.title('Missing values per column (out of 1310 rows)')
plt.show()
```

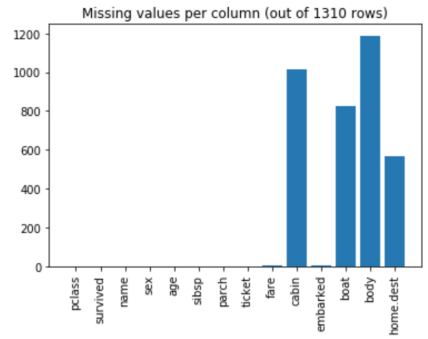

Figure 6: Bar plot of missing data per variable

```
print("SD before mean imputation: "+str(df['age'].std()))
print("SD after mean imputation: "+str(df_mean['age'].std()))

# This is underestimating the true SD
```

```
SD before mean imputation: 14.4134996999
SD after mean imputation: 12.8782770952
```

Figure 7: Standard deviation before and after imputing the mean

were not recorded and there is low variance, then the mean may be a good substitute.

Another type of single value imputation is filling in the value with the value after the missing value –backward filling– or with the value before this missing value –forward filling–. We can see that row 15 is now filled-in.

```
# Backward filling
df_back = df.fillna(method="bfill")

print(df_back[['age']].head(20))
```

```
        age
0   29.0000
1    0.9167
2    2.0000
3   30.0000
4   25.0000
5   48.0000
6   63.0000
7   39.0000
8   53.0000
9   71.0000
10  47.0000
11  18.0000
12  24.0000
13  26.0000
14  80.0000
15  24.0000
16  24.0000
17  50.0000
18  32.0000
19  36.0000
```

```
# Forward filling
df_forward = df.fillna(method="ffill")

print(df_forward[['age']].head(20))
```

```
        age
0   29.0000
1    0.9167
2    2.0000
3   30.0000
4   25.0000
5   48.0000
6   63.0000
7   39.0000
8   53.0000
9   71.0000
10  47.0000
11  18.0000
12  24.0000
13  26.0000
14  80.0000
15  80.0000
16  24.0000
17  50.0000
18  32.0000
19  36.0000
```

Figure 8: Backward and forward filling

The issue with forward and backward filling is that suppose that value around a NaN is an outlier (e.g., Age = 91). Then we would be duplicating the outlier, which could affect our ability to detect outliers and would also affect the mean. Perhaps this method might be useful if data is ordered along a certain variable, for instance, "Position in race". If one knows that subject in 4th place arrived almost at the same time as subjects in 3 or 5th place (which is common in short track competitions), then one could impute using this method. But be careful and analyze your specific case.

5 Multiple Value Imputation

Another approach used in order to deal with missing data is the Multiple Imputation technique. This approach as opposed to single imputation, accounts for the statistical uncertainty in the imputations.[1]
This is done by creating several imputed values –not just one– for each missing value, each of which is predicted for a slightly different model and each of which also reflects sampling variability.[2]
At the end of the process we will have multiple "complete" datasets in which we are able to calculate variance and infer which of the new data values is better suited to replace the missing value.

The goal of Multiple Imputation is not to re-create the individual missing values as close as possible to the true ones, but to handle missing data to achieve valid statistical inference.[3]
In the following example, we will show how multiple imputation works using R as a programming language with the well documented library MICE. The MICE library is a set of functions that deal with missing data and are able to deal with numerical values with different algorithms (e.g., "pmm"(predictive mean matching), "rd"(Random Forest), "norm" (Bayesian Linear Regression), among others).
Furthermore, categorical types of data can also be predicted using algorithms like "logreg"(Logistic Regression) or "polyreg" (Polytomous Logistic Regression).
N.B: In order to use these methods, the data has to be factorized in order to enable machine learning on each category.

Demonstration

We worked with the Titanic dataset and we will fill in a specific type of entry that is currently missing, in this case, Age.

Multi Imputation with R

```
library(mice)

initial_file_path = "./titanic_original.csv"
table <- read.csv(file=initial_file_path, header=TRUE, sep=",", stringsAsFactors = FALSE)
```

```
table
```

pclass	survived	name	sex	age	sibsp	parch	ticket	fare	cabin	embarked	boat	body	home.dest
1	1	Allen, Miss. Elisabeth Walton	female	29.0000	0	0	24160	211.3375	B5	S	2	NA	St Louis, MO
1	1	Allison, Master. Hudson Trevor	male	0.9167	1	2	113781	151.5500	C22 C26	S	11	NA	Montreal, PQ / Chesterville, ON
1	0	Allison, Miss. Helen Loraine	female	2.0000	1	2	113781	151.5500	C22 C26	S		NA	Montreal, PQ / Chesterville, ON
1	0	Allison, Mr. Hudson Joshua Creighton	male	30.0000	1	2	113781	151.5500	C22 C26	S		135	Montreal, PQ / Chesterville, ON
1	0	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)	female	25.0000	1	2	113781	151.5500	C22 C26	S		NA	Montreal, PQ / Chesterville, ON
1	1	Anderson, Mr. Harry	male	48.0000	0	0	19952	26.5500	E12	S	3	NA	New York, NY
1	1	Andrews, Miss. Kornelia Theodosia	female	63.0000	1	0	13502	77.9583	D7	S	10	NA	Hudson, NY
1	0	Andrews, Mr. Thomas Jr	male	39.0000	0	0	112050	0.0000	A36	S		NA	Belfast, NI
1		Appleton, Mrs. Edward Dale											

Figure 9: Importing the titanic dataset

After we imported our table to work on, we can analyze it, as always, to see how many entries are missing for each column.

Percentage of entries missing per column

```
pMiss <- function(x){sum(is.na(x))/length(x)*100}
#2 indicates the columns
apply(table,2,pMiss)
```

pclass	0.0763358778625954
survived	0.0763358778625954
name	0
sex	0
age	20.1526717557252
sibsp	0.0763358778625954
parch	0.0763358778625954
ticket	0
fare	0.152671755725191
cabin	0
embarked	0
boat	0
body	90.763358778626
home.dest	0

Figure 10: High Percentage of Missing Data

Then we can apply the MICE function with the right parameter in order to obtain multiple suggested values for each entry (i.e. row) where the value is missing.

Thanks to MICE I have a table with 5 new columns , each entry of the column is a possible value of the missing value in the row

```
tmp <- tempData$imp$age
#Renaiming The Columns Name
colnames(tmp) <- c("age1","age2","age3","age4","age5")
head(tmp,n=15)
#I'll pick the median for each entry in order to reduce at minimum the error,
#to avoid very large or very
```

	age1	age2	age3	age4	age5
16	57	27	38	39	58
38	27	16	38	32	35
41	36	51	47	47	54
47	19	27	51	56	51
60	45	22	30	33	38
70	35	32	42	45	52
71	30	58	51	56	62
75	47	33	54	58	37
81	22	51	58	47	42
107	55	64	71	46	45
108	51	22	33	55	43
109	27	23	49	52	64
119	47	30	37	62	45
122	43	23	44	34	44
126	46	51	51	46	37

Figure 11: Creating Multiple Datasets

We can easily see that we could fill the 16th entry with 5 different values. There are different techniques to choose from in order to fill the incomplete dataset, like taking into account standard error, variance, and mean of each table. We choose to fill the incomplete dataset with the median of each row, since the median is more robust than other technique against outliers.

```
table
```

pclass	survived	name	sex	age	sibsp	parch	ticket	fare	cabin	embarked	boat	body	home.dest
1	1	Allen, Miss. Elisabeth Walton	female	29.0000	0	0	24160	211.3375	B5	S	2	NA	St Louis, MO
1	1	Allison, Master. Hudson Trevor	male	0.9167	1	2	113781	151.5500	C22 C26	S	11	NA	Montreal, PQ / Chesterville, ON
1	0	Allison, Miss. Helen Loraine	female	2.0000	1	2	113781	151.5500	C22 C26	S		NA	Montreal, PQ / Chesterville, ON
1	0	Allison, Mr. Hudson Joshua Creighton	male	30.0000	1	2	113781	151.5500	C22 C26	S		135	Montreal, PQ / Chesterville, ON
1	0	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)	female	25.0000	1	2	113781	151.5500	C22 C26	S		NA	Montreal, PQ / Chesterville, ON
1	1	Anderson, Mr. Harry	male	48.0000	0	0	19952	26.5500	E12	S	3	NA	New York, NY
1	1	Andrews, Miss. Kornelia Theodosia	female	63.0000	1	0	13502	77.9583	D7	S	10	NA	Hudson, NY
1	0	Andrews, Mr. Thomas Jr	male	39.0000	0	0	112050	0.0000	A36	S		NA	Belfast, NI
1	1	Appleton, Mrs. Edward Dale (Charlotte Lamson)	female	53.0000	2	0	11769	51.4792	C101	S	D	NA	Bayside, Queens, NY

We can easily see that all entries of age is filled

```
apply(table,2,pMiss)
```

pclass	0.0763358778625954
survived	0.0763358778625954
name	0
sex	0
age	0
sibsp	0.0763358778625954
parch	0.0763358778625954
ticket	0
fare	0.152671755725191
cabin	0
embarked	0
boat	0
body	90.763358778626
home.dest	0

Figure 12: Every entry of Age now is filled-in

Question: Why can Multiple Imputation be better than Single Imputation?
Answer: The problem with the Single Imputation technique is that is does not take into account randomness, it simply fills the data with already known data. For example, if in the dataset there are only "regular" values (values close to the mean), the missing data will be filled in by these "regular" value. Instead, in real cases where randomness occur, there will be some outliers that deviate a bit the variance and the standard distribution of the data. Multiple Imputation incorporates randomness, performing multiple iterations on the same entry and taking as an input a seed that helps in enhancing randomness of data.

6 6. Conclusions

We covered the theory behind missing data and some examples of how to deal with it including a Jupyter Notebook implementation available online. Discarding data (i.e. complete-case analysis) should only be used when there is very little missing data. We covered several of the single value methods including mean imputation and backward and forward filling. We reviewed the negative aspects of these methods but also when they might be used. Finally, we reviewed multiple value imputation, which takes into account the variance and uncertainty of the missing data and provided a more powerful solution worth trying.

When comparing Python and R, generally speaking, one could implement or find implementations for all methods in both languages. However, we found a more developed library for multiple value

implementation in R called “MICE”. A similar library has been implemented in Python called "fancyimpute" but there is not a well-developed documentation for multiple value imputation. Therefore, this is an example for "monolingual" Python users as to what's out there. R users have a long history of well-developed tutorials and a wide-array of documentation for statistical analysis and data science.

References

[1] Azur, M. J., Stuart, E. A., Frangakis, C., & Leaf, P. J. (2011). Multiple imputation by chained equations: what is it and how does it work?. International journal of methods in psychiatric research, 20(1), 40-49.

[2] Vaughn, B. K. (2008). Data analysis using regression and multilevel/hierarchical models, by Gelman, A., & Hill.

[3] Soley-Bori, M. (2013). Dealing with missing data: Key assumptions and methods for applied analysis. Boston University.

[4] Pigott, T. D. (2001). A review of methods for missing data. Educational research and evaluation, 7(4), 353-383.