Data Profiling and Cleaning
Handling Missing Values

UBCO Master of Data Science – DATA 542

Fatemeh Fard

Lecture 2 - review

DataFrames and Series objects in Pandas

Functions on DataFrames and Series to index and select data

Functions on DataFrames and Series to sub-set data (slicing)

Example data

	department	age	height	food	color
Jane	biology	32	160.0	steak	blue
Sara	chemistry	40	158.0	lamb	red
Nicole	NaN	35	170.0	apple	orange
Kaden	computer	50	NaN	NaN	yellow
Jeff	statistics	35	175.0	steak	blue
Reza	NaN	45	165.0	lamb	red
John	statistics	45	NaN	NaN	orange
Ramon	computer	40	175.0	cheese	yellow
Bryce	engineering	28	180.0	steak	blue

Discussion

What do you need to know about your data, before starting the analysis?

Discussion

What do you need to know about your data, before starting the analysis?

1. value types, ranges,
2. errors → negative age
3. outliers →
4. missing values →
   1. columns or rows have missing values
   2. percentage of null values
   3. why they are there
   4. Can we replace them
5. edge cases
6. how data is collected
7. row count & data size
8. what are your features and column names
9. some basic statistics about data

Data Profiling

Reviewing your data source, content, and identifying key quality assessments (e.g. outliers, null values)

1- Read your data

```
pd.read_csv() #Use header=None to include first row in the data
pd.read_sql()
pd.read_excel()
```

Data Profiling cont.

2- Take a look

```
df.head(integer)
df.tail(integer)
```

Optional number, indicating how many rows

	department	age	height	food	color	sport
Jane	biology	32	160.0	steak	blue	NaN
Sara	chemistry	40	158.0	lamb	red	NaN

	department	age	height	food	color	sport
Ramon	computer	40	175.0	cheese	yellow	NaN
Bryce	engineering	28	180.0	steak	blue	NaN

Data Profiling cont.

3- Gain some information about values

Null values

Data types

Number of entries

`df.info()`

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 9 entries, Jane to Bryce
Data columns (total 6 columns):
department    7 non-null object
age           9 non-null int64
height        7 non-null float64
food          7 non-null object
color         9 non-null object
sport         0 non-null float64
dtypes: float64(2), int64(1), object(3)
memory usage: 504.0+ bytes
```

Data Profiling cont.

4- Explore your data using functions

Explore each column

```
df.age.sum() #count(), add(), agg
df.department.size()
```

Or explore the dataset

```
df.sample()
```

```
df.count()
```

```
department    7
age           9
height        7
food          7
color         9
sport         0
dtype: int64
```

Each of the functions can be applied on a column or the dataset

Data Profiling cont.

5- Examine statistics of your data

```
df.describe()
```

```
df.describe()
```

	age	height	sport
count	9.000000	7.000000	0.0
mean	38.888889	169.000000	NaN
std	7.043516	8.286535	NaN
min	28.000000	158.000000	NaN
25%	35.000000	162.500000	NaN
50%	40.000000	170.000000	NaN
75%	45.000000	175.000000	NaN
max	50.000000	180.000000	NaN

Data Profiling cont.

6- Run some query

`df.query()`

```
df.query('age > 35 and height <160')
```

	department	age	height	food	color	sport
Sara	chemistry	40	158.0	lamb	red	NaN

Note: `df.query('age < 40')` is equivalent to `df[df.age < 40]`

Try IT

Import the sample-data for lecture 3 and assess the data and column values

Exploring Missing Values

Discussion

1) Discuss some reasons for having null values in the data.

2) Why should we care about the null values, at all?!

Discussion

1) Discuss some reasons for having null values in the data.

2) Why should we care about the null values, at all?!

Q1 :

- Real null , False null : Reasons of not having a value for a record, feature
- Survey data : people refuse to answer
- Type of data & how the data is collected
- storage or corruption of data
- Collection errors / computation (technical) issues : e.g. sensor not working, server is down
- Mis-information / ambiguity / joining data from various sources
-

Data Profiling cont.

7- Examine null values

```
df.isnull()
df.notnull()
df.column_name.isnull()
```

```
df.notnull()
```

	department	age	height	food	color	sport
Jane	True	True	True	True	True	False
Sara	True	True	True	True	True	False
Nicole	False	True	True	True	True	False
Kaden	True	True	False	False	True	False
Jeff	True	True	True	True	True	False
Reza	False	True	True	True	True	False
John	True	True	False	False	True	False
Ramon	True	True	True	True	True	False
Bryce	True	True	True	True	True	False

Data Profiling cont.

7- Examine null values

```
df.isnull()
df.notnull()
df.column_name.isnull()
```

- impact statistics (e.g. min or count)
- emission or mistake (e.g. randomly?)
- how to handle them?
- effect on predictive models | performance of models
- reliability of analysis

```
df.notnull()
```

	department	age	height	food	color	sport
Jane	True	True	True	True	True	False
Sara	True	True	True	True	True	False
Nicole	False	True	True	True	True	False
Kaden	True	True	False	False	True	False
Jeff	True	True	True	True	True	False
Reza	False	True	True	True	True	False
John	True	True	False	False	True	False
Ramon	True	True	True	True	True	False
Bryce	True	True	True	True	True	False

Null values or missing values

Data can be missed because of:

- No existing value for an observation
  - Child name for families with no children
- Errors
- Manual mistakes
- Etc.

Missing values cont.

Missing Completely At Random (MCAR)

- No systematic way that makes some data missing
  - Example: In a survey, some data are lost or ignored to be filled

Missing At Random (MAR)

- There is a relationship between the observed data and the missing values
  - Example: men are more likely to reveal their weight, in a survey, specific professions are not willing to reveal their income

Missing Not At Random (MNAR)

- There is a systematic mechanism of missing data
- "propensity of a value to be missing and its values" or missing reason depends on the missing value itself
  - Example: In a survey, people with less than $500/month do not reveal their income, education filed is missing with the lowest educated ..

MNAR is Non-Ignorable

- chi-square test, classification

https://www.theanalysisfactor.com/missing-data-mechanism/

Missing values in Pandas

NAN(Not A Number) or Null

In `Numpy`: `np.nan`

```
pd.DataFrame([{'a': 1, 'b': 2},
              {'b': 3, 'c': 4}])
```

	a	b	c
0	1.0	2	NaN
1	NaN	3	4.0

Exploring the missing value patterns with visualizations

Visualization helps have an understanding of most missing values

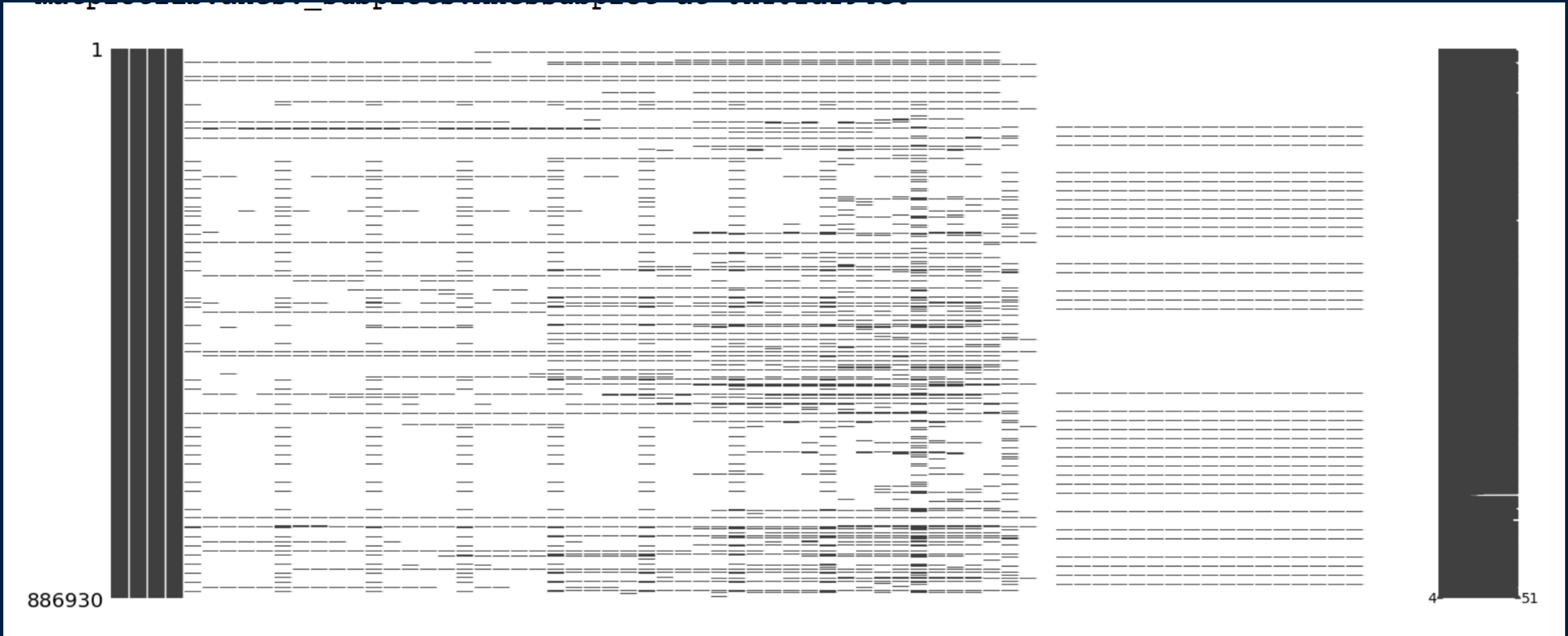
Heatmaps help in understanding the correlation between variables (columns)

Useful in big datasets

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 886930 entries, 0 to 886929
Data columns (total 70 columns):
Country Name      886930 non-null object
Country Code      886930 non-null object
Indicator Name    886930 non-null object
Indicator Code    886930 non-null object
1970              72288 non-null float64
1971              35537 non-null float64
1972              35619 non-null float64
1973              35545 non-null float64
1974              35730 non-null float64
1975              87306 non-null float64
1976              37483 non-null float64
1977              37574 non-null float64
1978              37576 non-null float64
1979              36809 non-null float64
1980              89122 non-null float64
1981              38777 non-null float64
1982              37511 non-null float64
1983              38460 non-null float64
1984              38606 non-null float64
1985              90296 non-null float64
1986              39372 non-null float64
1987              38641 non-null float64
1988              38552 non-null float64
1989              37540 non-null float64
1990              124405 non-null float64
```

Check sparsity of data in matrix

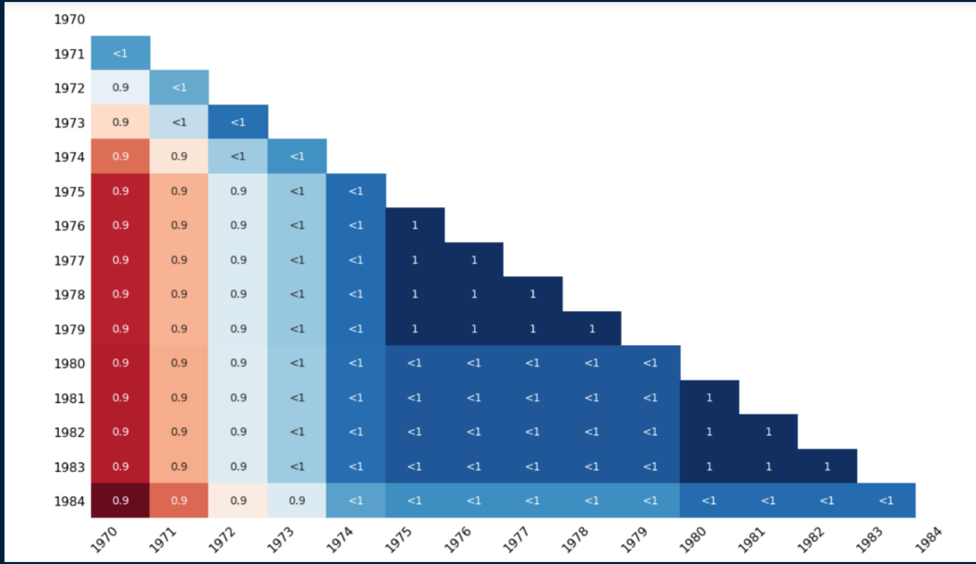
```
msno.matrix(df)
```



Heatmaps

"The missingno correlation heatmap measures nullity correlation: how strongly the presence or absence of one variable affects the presence of another"

https://github.com/ResidentMario/missingno




```
msno.heatmap(df)
```

Handling Missing Values

Handling missing values

Talk to the data collection source

1- Drop the missing value

- Drop the whole variable (column)
- Drop the data entry (suitable for less observations)

2- Replace missing values

- Average value of the entire variable
- Average value of the same group of data
- Most frequency in categorical data (e.g. color)
- Replace with the mode of the column in categorical data
- Use expert domain knowledge

3- Predict missing values with machine learning algorithms

4- Leave it as it is

Handling missing values in Pandas

```
isnull() # A boolean mask indicating missing values

notnull() # Opposite of isnull()

dropna() # Return a filtered version of the data

fillna() # Return a copy of the data with missing values filled or imputed
```

1- Drop missing values

Drop the values using `dropna()` to drop a row or a column containing `NAN`

```
data.dropna(column_list, axis, inplace)
data.dropna(axis = 0) #Drop entire row
data.dropna(axis = 1) #Drop entire column
```

2- Replace missing values using fillna()

Replace the missing values using

`fillna(new_value)`	Fills the NAN with the new_value
`fillna(method, axis)`	Fills the NAN with a forward_filling or backward_filling, along the specified axis

fillna() returns a copy of the array with the null values replaced.

Use isnull() as a mask to work with the array inplace.

2- Replace missing values using fillna() cont.

Fill_forward, carries the previous value forward to fill the NAN

Fill_backward, carries the next value backward to fill the NAN

Fill Forward

Fill Backward

	department	age	height	food	color	sport
Jane	biology	32	160.0	steak	blue	NaN
Sara	chemistry	40	158.0	lamb	red	NaN
Nicole	NaN	35	170.0	apple	orange	NaN
Kaden	computer	50	NaN	NaN	yellow	NaN
Jeff	statistics	35	175.0	steak	blue	NaN
Reza	NaN	45	165.0	lamb	red	NaN
John	statistics	45	NaN	NaN	orange	NaN
Ramon	computer	40	175.0	cheese	yellow	NaN
Bryce	engineering	28	180.0	steak	blue	NaN

2- Replace missing values using replace()

Replace the missing values using

```
replace(missing_value, new_value)
mean = data.height.mean()
#alternation: data['height'].mean()
data['height'].replace(np.nan, mean)
```

Remember: The actual DataFrame does not change, until you assign the new column to it

```
Jane       160.0
Sara       158.0
Nicole     170.0
Kaden      169.0
Jeff       175.0
Reza       165.0
John       169.0
Ramon      175.0
Bryce      180.0
Name: height, dtype: float64
```

2- Replace missing values cont.

Replace with Mean, Median, or Mode

More accurate values for replacement:

- Average of the group (e.g. Male and Female, ethnicity, department, etc.)

Disadvantages:

Imputation with Mean reduces the variance in the data

3- Predict missing values with machine learning algorithms

Estimate missing values with:

- Regression
- ANOVA
- Logistic regression

Rewrite SQL queries in Pandas

https://medium.com/jbennetcodes/how-to-rewrite-your-sql-queries-in-pandas-and-more-149d341fc53e

Try it

Download the dataset

https://www.kaggle.com/dansbecker/handling-missing-values

Explore the dataset (data profiling)

Explore the missing values

Visualize missing values

Use heatmaps and interpret the results, whether there is a correlation between various missing data features

Try it - Learn more

Try working with the experience from Kaggle:

https://www.kaggle.com/dansbecker/handling-missing-values/notebook

In Practice

Example data

	department	age	height	food	color
Jane	biology	32	160.0	steak	blue
Sara	chemistry	40	158.0	lamb	red
Nicole	NaN	35	170.0	apple	orange
Kaden	computer	50	NaN	NaN	yellow
Jeff	statistics	35	175.0	steak	blue
Reza	NaN	45	165.0	lamb	red
John	statistics	45	NaN	NaN	orange
Ramon	computer	40	175.0	cheese	yellow
Bryce	engineering	28	180.0	steak	blue

Data Profiling

Reviewing your data source, content, and identifying key quality assessments (e.g. outliers, null values)

1- Read your data

```
pd.read_csv() #Use header=None to include first row in the data

pd.read_sql()

pd.read_excel()
```

Data Profiling cont.

2- Take a look

```
df.head(integer)
df.tail(integer)
```

Optional number, indicating how number of rows

	department	age	height	food	color	sport
Jane	biology	32	160.0	steak	blue	NaN
Sara	chemistry	40	158.0	lamb	red	NaN

	department	age	height	food	color	sport
Ramon	computer	40	175.0	cheese	yellow	NaN
Bryce	engineering	28	180.0	steak	blue	NaN

Data Profiling cont.

3- Gain some information about values

Null values

Data types

Number of entries

`df.info()`

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 9 entries, Jane to Bryce
Data columns (total 6 columns):
department    7 non-null object
age           9 non-null int64
height        7 non-null float64
food          7 non-null object
color         9 non-null object
sport         0 non-null float64
dtypes: float64(2), int64(1), object(3)
memory usage: 504.0+ bytes
```

Data Profiling cont.

4- Explore your data using functions

Explore each column

```
df.age.sum() #count(), add(), agg
df.department.size()
```

Or explore the dataset

```
df.sample()
```

```
df.count()
```

```
department    7
age           9
height        7
food          7
color         9
sport         0
dtype: int64
```

Each of the functions can be applied on a column or the dataset

Data Profiling cont.

5- Examine statistics of your data

```
df.describe()
```

```
df.describe()
```

	age	height	sport
count	9.000000	7.000000	0.0
mean	38.888889	169.000000	NaN
std	7.043516	8.286535	NaN
min	28.000000	158.000000	NaN
25%	35.000000	162.500000	NaN
50%	40.000000	170.000000	NaN
75%	45.000000	175.000000	NaN
max	50.000000	180.000000	NaN

Data Profiling cont.

6- Run some query

`df.query()`

```
df.query('age > 35 and height <160')
```

	department	age	height	food	color	sport
Sara	chemistry	40	158.0	lamb	red	NaN

Note: `df.query('age < 40')` is equivalent to `df[df.age < 40]`

Data Profiling cont.

7- Examine null values

```
df.isnull()
df.notnull()
df.column_name.isnull()
```

```
df.notnull()
```

	department	age	height	food	color	sport
Jane	True	True	True	True	True	False
Sara	True	True	True	True	True	False
Nicole	False	True	True	True	True	False
Kaden	True	True	False	False	True	False
Jeff	True	True	True	True	True	False
Reza	False	True	True	True	True	False
John	True	True	False	False	True	False
Ramon	True	True	True	True	True	False
Bryce	True	True	True	True	True	False

Null values or missing values

Data can be missed because of:

- No existing value for an observation
  - Child name for families with no children
- Errors
- Manual mistakes
- Etc.

Missing values cont.

Missing Completely At Random (MCAR)

- No systematic way that makes some data missing
  - Example: In a survey, some data are lost or ignored to be filled

Missing At Random (MAR)

- There is a relationship between the observed data and the missing values
  - Example: men are more likely to reveal their weight, in a survey, specific professions are not willing to reveal their income

Missing Not At Random (MNAR)

- There is a systematic mechanism of missing data
- “propensity of a value to be missing and its values” or missing reason depends on the missing value itself
  - Example: In a survey, people with less than $500/month do not reveal their income, education filed is missing with the lowest educated ..

MNAR is Non-Ignorable

- chi-square test, classification

https://www.theanalysisfactor.com/missing-data-mechanism/

Missing values in Pandas

NAN(Not A Number) or Null

In `Numpy`: `np.nan`

```
pd.DataFrame([{'a': 1, 'b': 2},
              {'b': 3, 'c': 4}])
```

	a	b	c
0	1.0	2	NaN
1	NaN	3	4.0

Exploring the missing value patterns with visualizations

Visualization helps have an understanding of most missing values

Heatmaps help in understanding the correlation between variables (columns)

Visualization packages in Python

```
seaborn
pip install seaborn
missingno
pip install missingno
```

https://seaborn.pydata.org

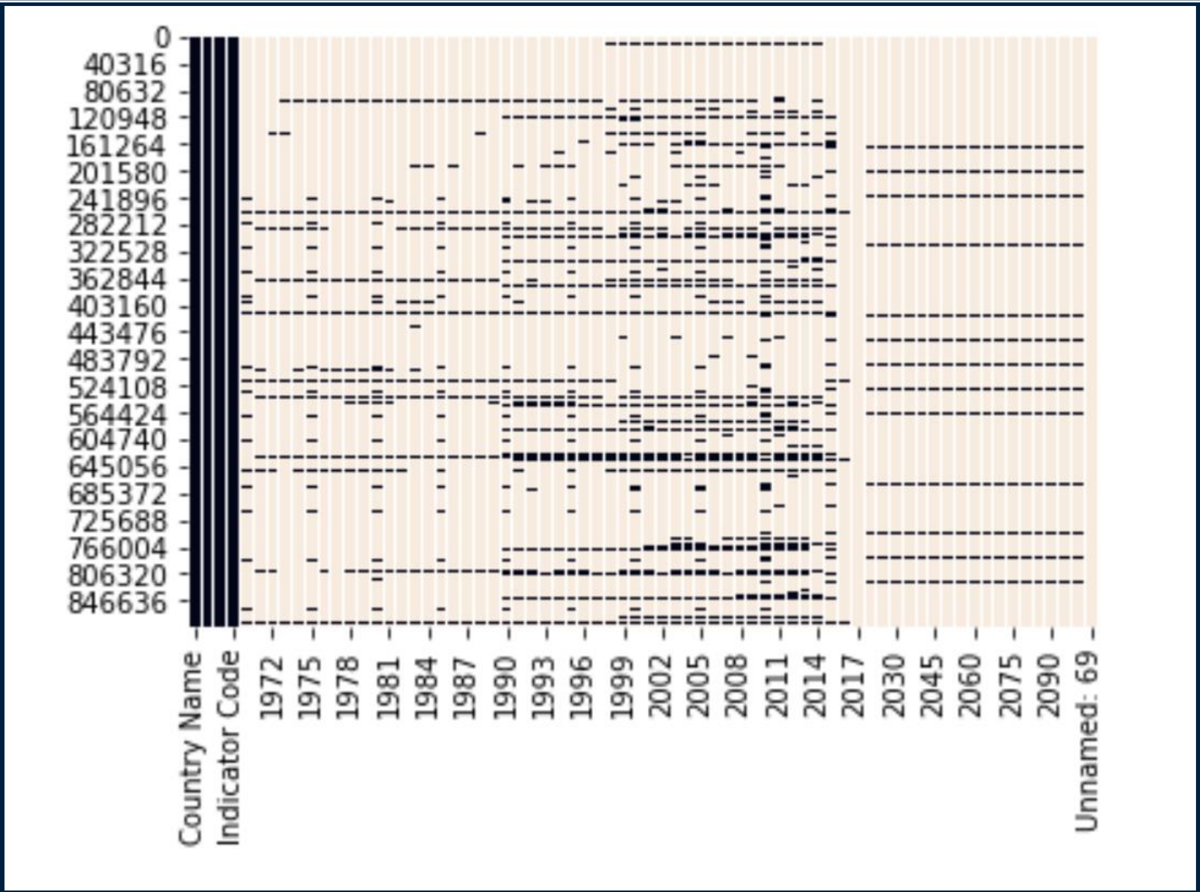
https://github.com/ResidentMario/missingno

```
import missingno as msno
```

Useful in big datasets

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 886930 entries, 0 to 886929
Data columns (total 70 columns):
Country Name      886930 non-null object
Country Code      886930 non-null object
Indicator Name    886930 non-null object
Indicator Code    886930 non-null object
1970              72288 non-null float64
1971              35537 non-null float64
1972              35619 non-null float64
1973              35545 non-null float64
1974              35730 non-null float64
1975              87306 non-null float64
1976              37483 non-null float64
1977              37574 non-null float64
1978              37576 non-null float64
1979              36809 non-null float64
1980              89122 non-null float64
1981              38777 non-null float64
1982              37511 non-null float64
1983              38460 non-null float64
1984              38606 non-null float64
1985              90296 non-null float64
1986              39372 non-null float64
1987              38641 non-null float64
1988              38552 non-null float64
1989              37540 non-null float64
1990              124405 non-null float64
```

Check sparsity with matrix



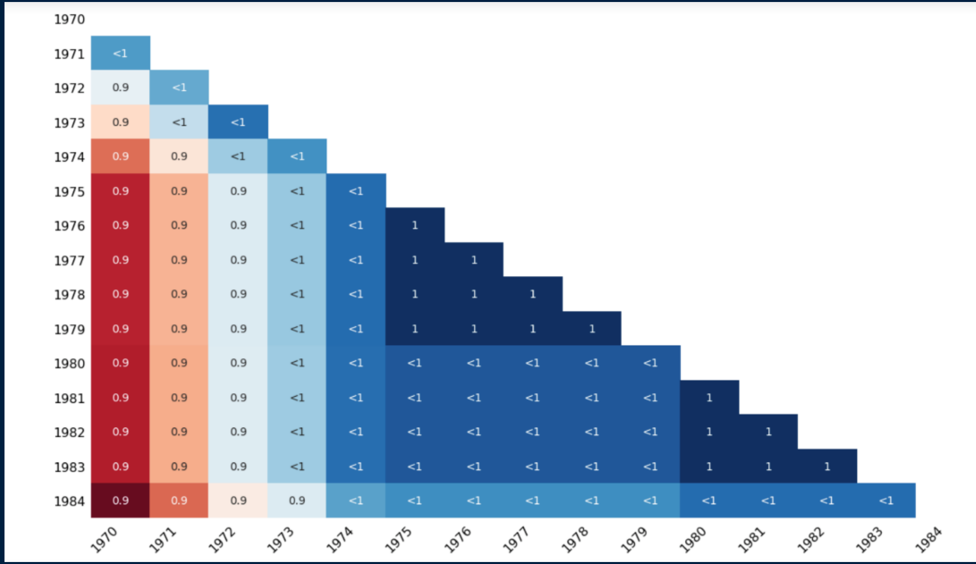
Check sparsity of data in matrix

```
msno.matrix(df)
```

Heatmaps

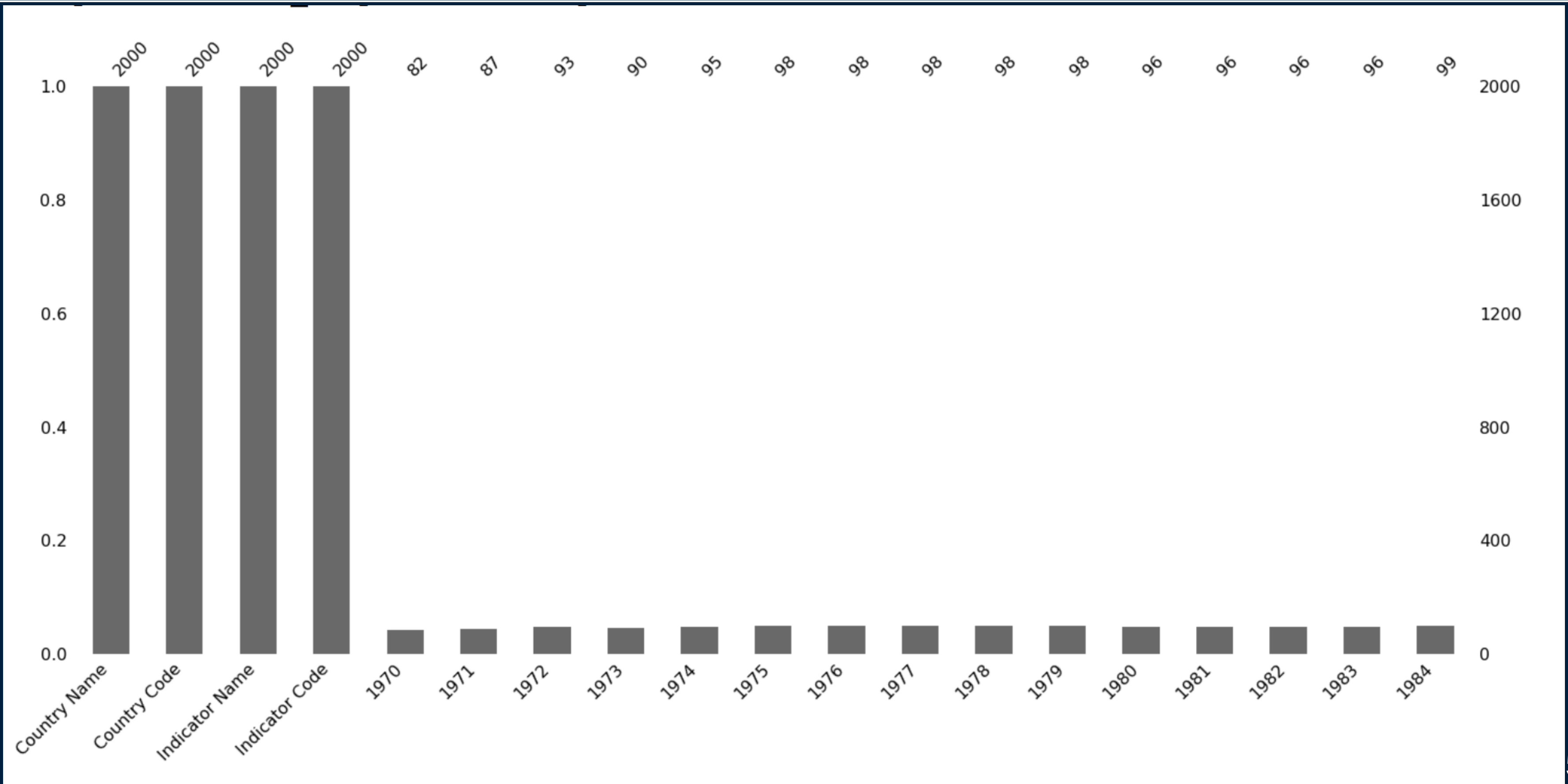
"The missingno correlation heatmap measures nullity correlation: how strongly the presence or absence of one variable affects the presence of another"

https://github.com/ResidentMario/missingno




```
msno.heatmap(df)
```

Barcharts


2000
2000
2000
2000
82
87
93
90
95
98
98
98
98
98
96
96
96
96
99
1.0
0.8
0.6
0.4
0.2
0.0
2000
1600
1200
800
400
0
Country Name
Country Code
Indicator Name
Indicator Code
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984

Other types of charts

Dendrogram

Geoplots

Usage of visualization functions in Python

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import missingno as msno
msno.matrix(df)
msno.heatmap(df)
msno.bar(df)
sns.heatmap(df.isnull(), cbar=False)
```

Handling missing values

Talk to the data collection source

1- Drop the missing value

- Drop the whole variable (column)
- Drop the data entry (suitable for less observations)

2- Replace missing values

- Average value of the entire variable
- Average value of the same group of data
- Most frequency in categorical data (e.g. color)
- Replace with the mode of the column in categorical data
- Use expert domain knowledge

3- Predict missing values with machine learning algorithms

4- Leave it as it is

Handling missing values in Pandas

```
isnull() # A boolean mask indicating missing values

notnull() # Opposite of isnull()

dropna() # Return a filtered version of the data

fillna() # Return a copy of the data with missing values filled or imputed
```

1- Drop missing values

Drop the values using `dropna()` to drop a row or a column containing `NAN`

```
data.dropna(column_list, axis, inplace)
data.dropna(axis = 0) #Drop entire row
data.dropna(axis = 1) #Drop entire column
```

1- Drop missing values with dropna

```
data.dropna(axis = 0)
```

Drops all rows that have a missing value

```
data.dropna(axis = 'columns')
```

Drops all columns that have a missing value

```
data.dropna(subset=['height'], axis = 0)
```

1- Drop missing values with dropna cont.

```
data.dropna(axis = 0)
```

	department	age	height	food	color
Jane	biology	32	160.0	steak	blue
Sara	chemistry	40	158.0	lamb	red
Jeff	statistics	35	175.0	steak	blue
Ramon	computer	40	175.0	cheese	yellow
Bryce	engineering	28	180.0	steak	blue

```
data.dropna(subset=['height'],
axis = 0)
```

	department	age	height	food	color
Jane	biology	32	160.0	steak	blue
Sara	chemistry	40	158.0	lamb	red
Nicole	NaN	35	170.0	apple	orange
Jeff	statistics	35	175.0	steak	blue
Reza	NaN	45	165.0	lamb	red
Ramon	computer	40	175.0	cheese	yellow
Bryce	engineering	28	180.0	steak	blue

1- Drop missing values with dropna cont.

```
data.dropna(axis =
'columns')
```

	Unnamed: 0	age	color
0	Jane	32	blue
1	Sara	40	red
2	Nicole	35	orange
3	Kaden	50	yellow
4	Jeff	35	blue
5	Reza	45	red
6	John	45	orange
7	Ramon	40	yellow
8	Bryce	28	blue

1- Drop missing values with dropna

Control the amount of drops by `thresh` and `how`

Default is: `how = 'any'` Drops specified axis containing a null value

Alternate: `how = 'all'` Drops specified axis that are ALL null values

Discuss the results

```
data.dropna(axis=1
, how ='all')
```

	Unnamed: 0	department	age	height	food	color	sport
0	Jane	biology	32	160.0	steak	blue	NaN
1	Sara	chemistry	40	158.0	lamb	red	NaN
2	Nicole	NaN	35	170.0	apple	orange	NaN
3	Kaden	computer	50	NaN	NaN	yellow	NaN
4	Jeff	statistics	35	175.0	steak	blue	NaN
5	Reza	NaN	45	165.0	lamb	red	NaN
6	John	statistics	45	NaN	NaN	orange	NaN
7	Ramon	computer	40	175.0	cheese	yellow	NaN
8	Bryce	engineering	28	180.0	steak	blue	NaN

1- Drop missing values with dropna

Control the amount of drops by `thresh` and `how`

`thresh = integer` Requires that many non-NAN values

Discuss other alternations

```
data.dropna(axis=0
, thresh = 4)
```

	Unnamed: 0	department	age	height	food	color	sport
0	Jane	biology	32	160.0	steak	blue	NaN
1	Sara	chemistry	40	158.0	lamb	red	NaN
2	Nicole	NaN	35	170.0	apple	orange	NaN
3	Kaden	computer	50	NaN	NaN	yellow	NaN
4	Jeff	statistics	35	175.0	steak	blue	NaN
5	Reza	NaN	45	165.0	lamb	red	NaN
6	John	statistics	45	NaN	NaN	orange	NaN
7	Ramon	computer	40	175.0	cheese	yellow	NaN
8	Bryce	engineering	28	180.0	steak	blue	NaN

Important note

```
data.dropna(axis = 0)

data.dropna(subset=['height'], axis = 0)
```

None of the above modifies the actual DataFrame.

```
data.dropna(subset=['height'], axis = 0,
inplace = True)
```

Or

```
data = data.dropna(subset=['height'], axis = 0)
```

2- Replace missing values using fillna()

Replace the missing values using

`fillna(new_value)`	Fills the NAN with the new_value
`fillna(method, axis)`	Fills the NAN with a forward_filling or backward_filling, along the specified axis

fillna() returns a copy of the array with the null values replaced.

Use isnull() as a mask to work with the array inplace.

2- Replace missing values using fillna() cont.

Fill_forward, carries the previous value forward to fill the NAN

Fill_backward, carries the next value backward to fill the NAN

Fill Forward

Fill Backward

	department	age	height	food	color	sport
Jane	biology	32	160.0	steak	blue	NaN
Sara	chemistry	40	158.0	lamb	red	NaN
Nicole	NaN	35	170.0	apple	orange	NaN
Kaden	computer	50	NaN	NaN	yellow	NaN
Jeff	statistics	35	175.0	steak	blue	NaN
Reza	NaN	45	165.0	lamb	red	NaN
John	statistics	45	NaN	NaN	orange	NaN
Ramon	computer	40	175.0	cheese	yellow	NaN
Bryce	engineering	28	180.0	steak	blue	NaN

2- Replace missing values using fillna() cont.

Fill_forward, carries the previous value forward to fill the NAN

Fill_backward, carries the next value backward to fill the NAN

```
df.fillna(method='ffill', axis =0)
```

	department	age	height	food	color	sp
Jane	biology	32	160.0	steak	blue	N
Sara	chemistry	40	158.0	lamb	red	N
Nicole	chemistry	35	170.0	apple	orange	N
Kaden	computer	50	170.0	apple	yellow	N
Jeff	statistics	35	175.0	steak	blue	N

```
df.fillna(method='bfill', axis=0)
```

	department	age	height	food	color
Jane	biology	32	160.0	steak	blue
Sara	chemistry	40	158.0	lamb	red
Nicole	computer	35	170.0	apple	orange
Kaden	computer	50	175.0	steak	yellow
Jeff	statistics	35	175.0	steak	blue

2- Replace missing values using replace()

Replace the missing values using

```
replace(missing_value, new_value)
mean = data.height.mean()
#alternation: data['height'].mean()
data['height'].replace(np.nan, mean)
```

Remember: The actual DataFrame does not change, until you assign the new column to it

```
Jane      160.0
Sara      158.0
Nicole    170.0
Kaden     169.0
Jeff      175.0
Reza      165.0
John      169.0
Ramon     175.0
Bryce     180.0
Name: height, dtype: float64
```

Question

Question 1: Which of the following applies the modifications to the DataFrame `data`

```
mean = data.height.mean()
```

A) `data['height'] = data['height'].replace(np.nan, mean)`

B) `data.height = data.height.replace(np.nan, mean)`

C) `A and B`

D) `None`

2- Replace missing values cont.

Replace with Mean, Median, or Mode

More accurate values for replacement:

- Average of the group (e.g. Male and Female, ethnicity, department, etc.)

Discuss other alternations

Disadvantages:

Imputation with Mean reduces the variance in the data

2- Replace missing values cont.

Interpolate the numerical values with `Pandas Interpolate`

Nicole	NaN	35	170.0	apple
Kaden	computer	50	172.5	NaN
Jeff	statistics	35	175.0	steak
Reza	NaN	45	165.0	lamb
John	statistics	45	170.0	NaN
Ramon	computer	40	175.0	cheese

Install SciPy

```
data.interpolate()
```

Arguments:

`method:` polynomial, cubic, etc.

`limit:` will determine how many values after a non-NaN value will be filled out

`limit_direction:` fill forward or backward. Default is forward

More information at: https://pandas-docs.github.io/pandas-docs-travis/missing_data.html

2- Replace missing values cont.

Imputation package from Scikit Learn

```
 from sklearn.preprocessing import Imputer
values = mydata.values
imputer = Imputer(missing_values='NaN',
strategy='mean')
transformed_values =
imputer.fit_transform(values)

 # strategy can be changed to "median" and
"most_frequent"
```

2- Replace missing values cont.

Imputation package from Scikit Learn

```
from sklearn.impute import SimpleImputer
my_imputer = SimpleImputer()
data_with_imputed_values =
my_imputer.fit_transform(original_data)
```

Default value is Mean

3- Predict missing values with machine learning algorithms

Estimate missing values with:

- Regression
- ANOVA
- Logistic regression

Rewrite SQL queries in Pandas

https://medium.com/jbennetcodes/how-to-rewrite-your-sql-queries-in-pandas-and-more-149d341fc53e

Try it

Download the flight_dalay_2017 dataset

https://www.kaggle.com/dansbecker/handling-missing-values

Explore the dataset (data profiling)

Explore the missing values

Visualize missing values

Use heatmaps and interpret the results, whether there is a correlation between various missing data features

Try it - Learn more

Try working with the experience from Kaggle:

https://www.kaggle.com/dansbecker/handling-missing-values/notebook


UBC
THE UNIVERSITY OF BRITISH COLUMBIA