DATA TREATMENT
PART 1

UNIVERSITY of HOUSTON
DIVISION OF RESEARCH

DATA REVIEWING/EXPLORING

MEANING:
Initial look-up on the data by the analyst with end goal is to have a sense of the data.

METHODS:
1) Data Visualization 2) Statistical techniques

EXPLORE:
For example: Data variables, Data columns, Possible target feature ,Data shape

Pandas methods for EDA(Exploratory Data Analysis)

- df.head(10): It displays the data frame's first ten rows
- df.tail(5): It displays the data frame's last ten rows
- df.shape: It displays the data dimensions as (num_of_rows,num_of_columns)

DATA CLEANING

Why:

Large Organizations collect the data in various ways and there can be several merging of the data and result in a piling up different issues which can affect the final reporting of the data

> *80 percent of a data scientist's valuable time is spent simply finding, cleansing, and organizing data, leaving only 20 percent to actually perform analysis...*
>
> IBM Data Analytics

How to identify data problems:

1) *Reviewing the meta data*:

Data sometimes is self-explanatory thus looks clean but after reading the data document, data problems show-up

2) *Reviewing explored data*:

Data column contain Nan's ; Duplicate entries; Mis-matching data types etc

3) *Sanity Checks:*

Does data matches data documentation?

For example:

If you're working on the real time hospital data Does, the number patients visit the providers matches with actual recorded patient visits

Disclaimer:
NEVER "**MANIPULATE**" THE DATA-SET TO GET THE DESIRED RESULTS

CLEANING IS DIFFERENT THAN DATA MANIPULATION

Common Data Problems

- Datatype problems
- Data Range Constraints
- Duplicate Entries
- Missing Data/Entries

Datatype Problem

Python uses the following build in data types:

DATA TYPE	DATA TYPES IN A DATASET
Int ;Float	numeric
bool	True - False
date	Datetime
category	Finite set of values for a feature
str	String
object	Mixed Datatype

It is recommended data value should have a correct data type

Pandas Methods for interrogating data type of the data:

- df.info(verbose = True)

This method prints information about a DataFrame including the index dtype ,columns, non-null values, memory usage

- df.dtypes()

This method returns the series with the data type of each column
The result's index is the original Data Frame's columns

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10450 entries, 0 to 10449
Data columns (total 16 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   id                    10450 non-null  int64
 1   name                  10450 non-null  object
 2   host_id               10450 non-null  int64
 3   host_name             10447 non-null  object
 4   neighbourhood_group   0 non-null      float64
 5   neighbourhood         10450 non-null  int64
 6   latitude              10450 non-null  float64
 7   longitude             10450 non-null  float64
 8   room_type             10450 non-null  object
```

data2.info()

Datatype Problem

For example: Let's have a look at the data type of room_type feature

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10450 entries, 0 to 10449
Data columns (total 16 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   id                              10450 non-null  int64
 1   name                            10450 non-null  object
 2   host_id                         10450 non-null  int64
 3   host_name                       10447 non-null  object
 4   neighbourhood_group             0 non-null      float64
 5   neighbourhood                   10450 non-null  int64
 6   latitude                        10450 non-null  float64
 7   longitude                       10450 non-null  float64
 8   room_type                       10450 non-null  object
```

→ Object Data-type

Question: How to know what is the correct data-type?

- df[”column_name”] or df.column_name
- df.column_name.unique()

This method will print the series of unique data values of the column

```
array(['Entire home/apt', 'Private room', 'Hotel room', 'Shared room'],
      dtype=object)
```

The above series confirms category data-type of the column

Method of data-type conversion: df.column_name.astype(”data_type”)

Documentation: Pandas astype Method

- For example: data2.room_type.astype(“category”)

Datatype – Datetime datatype**

- Method of datatype conversion:
  - df.column_name.astype(”data_type”)

- Method of datetime datatype conversion:
  - df.column_name.pd.to_datetime(”data_type”)

Documentation: pandas.to_datetime

For example:

```
data2['last_review'] = pd.to_datetime(data2['last_review’])
```

Data Range Constraints

Such a problem can be found:
- **Numeric data**
- **Dates**

➢ *Meaning:*

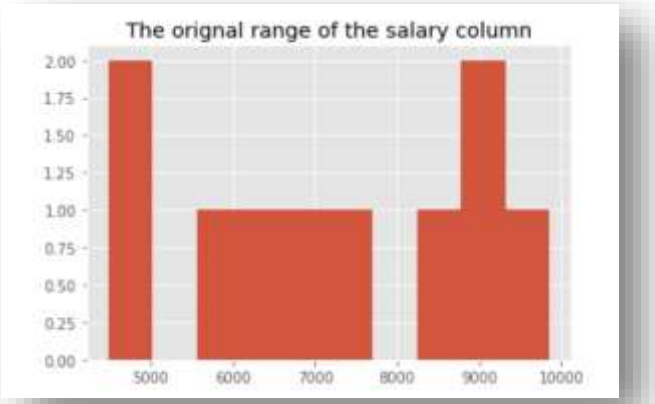
The range of the data is beyond the actual limit

For Example:

- ***Numeric data***

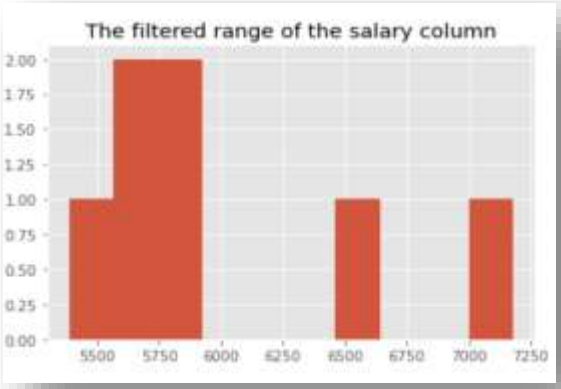
A Company has a salary column in a database

The range of the salary they are looking for is between $5000 and $8000



Filtering Method

```
salary_sample[
(salary_sample['Salary']< 8000)
&
 (salary_sample['Salary'] > 5000)
]
```



- ***Dates***

Let's look at a Medical Insurance dataset. They have recorded date-wise patients visiting the Emergency Room

	samp_date
2853	2000-03-01
1400	2007-08-12
1779	2015-07-02
4710	2015-10-22
2978	2012-05-16
93	2000-12-16
1771	2021-04-29

Date record

"2021-04-29" is incorrect

Solution: **Dropping Method**

```
sample_date.drop(sample_date.loc[sample_date['samp_date'] > pd.to_datetime(today)].index,
inplace = True)
```

Duplicate Entries

- *Meaning: Some entries may be more than once.*

For Example:

	Brand	Price	Color
0	Honda Civic	22000	Grey
1	Toyota Corolla	25000	Grey
2	Ford Focus	27000	White
3	Audi A4	35000	Black
4	Lexus	20000	Blue
5	Audi A4	35000	Black

Car-List

Duplicate Values implies uninformative data

How to detect duplicates in a large data-set?

By using pandas duplicated method

Documentation: *pandas.DataFrame.duplicated*

- *Column-wise :*
  - *col_duplicate = df.duplicated()*
    This will return the series of Boolean values
- *Row_wise:*
  row_duplicate = df.duplicated()
  df[row_duplicate]
  This will return the duplicated row

By default, the first entry is always false, and the second same entry is considered duplicated(or true)

Duplicate Entries

- *Special cases:*

  For Example:

  There can be cases such as:

  - No duplicates

id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_nights	number_of_reviews	last_review

  - Subset of data has duplicates

Brand	Price	Color	Model_year
Honda Civic	22000	Grey	2018
Toyota Corolla	25000	Grey	2018
Ford Focus	27000	White	2017
Audi A4	35000	Black	2020
Lexus	20000	Blue	2019
Audi A4	35000	Black	2020

Using the subset parameter:

```
Df_duplicates = df.duplicated( subset = ['Color', 'Model_year'],  keep = False)
```

```
Brand
Honda Civic        False
Toyota Corolla      True
Ford Focus         False
Audi A4            False
Lexus              False
Audi A4             True
dtype: bool
```

Duplicate values based on subset

Duplicate Entries

- *How to remove the duplicate entries?*

  - drop_duplicate( ) pandas' method
    Documentation: *pandas.DataFrame.drop_duplicates*

    - Drop duplicates in a subset

Brand	Price	Color	Model_year
Honda Civic	22000	Grey	2018
Ford Focus	27000	White	2017
Audi A4	35000	Black	2020
Lexus	20000	Blue	2019

df.drop_duplicate(subset = [‘Color’, ‘Model_year’])

***Note*:
Use df.shape() to see the change in number of rows and columns in a dataset after dropping duplicates

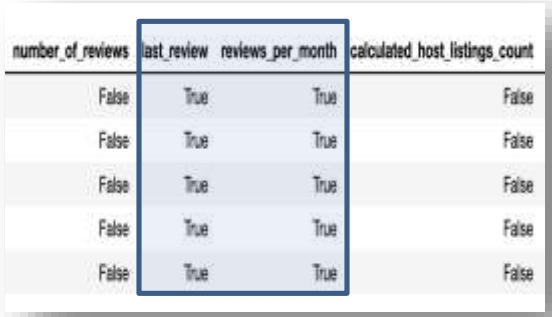
Missing Data Entries

- *Meaning:*
  - ***According to Wikipedia:***
    missing data *occur when no data value is stored for the variable in an observation*

- *Sources of missing data:*
  - *Data collected was not gathered for some entities*
  - *Programming error*
  - *Data recorded for a product does not have last year's record*

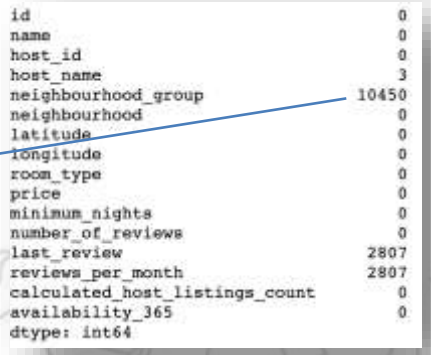
- *Methods of dealing missing data:*
  - *Detecting missing values*
    - *df.isna()*
      - *Documentation: pandas.DataFrame.isna()*
    - *df.isnull()*
      - *Documentation: pandas.DataFrame.isnull()*

    By default: Returns true for the missing values
  - *Summarizing missing values*
    - *df.isna().sum(axis = 0)*
      - *axis = 0 means sum along the rows*
      - *axis = 1 means sum along the columns*
  - *Summarizing based on the columns*
    - *df[df['column_name'].isnull()]*



number_of_reviews	last_review	reviews_per_month	calculated_host_listings_count
False	True	True	False
False	True	True	False
False	True	True	False
False	True	True	False
False	True	True	False

Missing values



```
id                                0
name                              0
host_id                           0
host_name                         3
neighbourhood_group           10450
neighbourhood                     0
latitude                          0
longitude                         0
room_type                         0
price                             0
minimum_nights                    0
number_of_reviews                 0
last_review                    2807
reviews_per_month              2807
calculated_host_listings_count    0
availability_365                  0
dtype: int64
```

Summarize missing values

Missing Data Entries

- *Methods of treating missing data:*
  - *Drop the missing values:*
    - *df.dropna(how = 'any').shape*
      - *Documentation: pandas.DataFrame.dropna*

        Parameter: (how =)

        row or column is removed from Data Frame, when we have at least one NA or all NA
        - 'any': If any NA values are present, drop that row or column
        - 'all' : If all values are NA, drop that row or column
  - *Fill **NA value** method:*
    - *We normally fill the NAN values with '0'*
      - df.fillna(*value=0, method=None, axis=None, inplace=False)*
      - *Documentation:pandas.DataFrame.fillna*
      - *For example:*

```
data2.reviews_per_month
0        0.17
1        0.06
2        3.78
3        0.14
4        2.00
        ...
10445     NaN
10446     NaN
10447     NaN
10448     NaN
10449     NaN
```

data2.reviews_per_month.fillna(0, inplace = True)

```
data2.reviews_per_month
0        0.17
1        0.06
2        3.78
3        0.14
4        2.00
        ...
10445    0.00
10446    0.00
10447    0.00
10448    0.00
10449    0.00
```

Next Steps: Assignment 1

- After selecting your data and preparing some initial research problems
- Step 1: Import your data
- Step 2: Explore your data
- Step 3:Try cleaning your data
  - Data type problem
  - Data range problem
  - Duplicate data
  - Missing data

Please upload your assignments : Moodle's Page