Data Analysis Process

5 Main Steps

1. Asking questions
2. Data Wrangling
3. Exploratory Data Analysis
4. Drawing Conclusions
5. Communicating Results

`Note` - Data Analysis Process in iteractive and non linear process.

Step 1: Asking Questions

How can I ask better questions?

- Subject Matter Expertise
- Experience

Examples.

1. What features will contriubute to my analysis?

2. What features are not important for my analysis?

3. Which of the features have a strong correlation?

4. Do I need data preprocessing?

5. What kind of feature manipulation/engineering is required?

Step 2: Data Wrangling/Munging

Data Wrangling/Munging is the process of transforming and mapping the data from one raw data form into another format with the intent of making it more appropriate and valuable for a variety of purposes such as analytics.

Data Wrangling is consist of three steps:

2a. Gathering Data
2b. Accessing Data
2c. Cleaning Data

Step 3: Exploratory Data Analysis

Explore and Augment the Data

EDA is consist of two steps:

3a. Exploring Data
3b. Augmenting Data

3a. Exploring Data

1. Finding Correlation and Covariance
2. Doing Univariate and Multivariate analysis
3. Plotting Graphs(Data Viz.)

3b. Augmenting Data

1. Removing Outliers
2. Merging Dataframes
3. Feature Engineering

Step 4: Drawing Conclusions

Conclusions can be drawn using various techniques:

4a. Machine Learning
4b. Inferential Statics
4c. Descriptive Statics

Step 5: Communicating Results

Outcomes can be shared via

5a. Reports
5b. PPTs 5c. Blogs
5d. In person

2a. Data Gathering

```
import numpy as np
import pandas as pd
```

Local Files

Working with csv

Opening a local csv file

```
df = pd.read_csv('Datasets/aug_train.csv')
df
```

```
       enrollee_id      city  city_development_index gender  \
0             8949  city_103                   0.920   Male
1            29725   city_40                   0.776   Male
2            11561   city_21                   0.624    NaN
3            33241  city_115                   0.789    NaN
4              666  city_162                   0.767   Male
...            ...       ...                     ...    ...
19153         7386  city_173                   0.878   Male
19154        31398  city_103                   0.920   Male
19155        24576  city_103                   0.920   Male
19156         5756   city_65                   0.802   Male
19157        23834   city_67                   0.855    NaN

           relevent_experience enrolled_university education_level  \
0      Has relevent experience       no_enrollment        Graduate
1       No relevent experience       no_enrollment        Graduate
2       No relevent experience    Full time course        Graduate
3       No relevent experience                 NaN        Graduate
4      Has relevent experience       no_enrollment         Masters
...                        ...                 ...             ...
19153   No relevent experience       no_enrollment        Graduate
19154  Has relevent experience       no_enrollment        Graduate
19155  Has relevent experience       no_enrollment        Graduate
19156  Has relevent experience       no_enrollment     High School
19157   No relevent experience       no_enrollment  Primary School

      major_discipline experience company_size    company_type
last_new_job  \
0                 STEM        >20          NaN             NaN
1
1                 STEM         15        50-99         Pvt Ltd
>4
2                 STEM          5          NaN             NaN
never
3      Business Degree         <1          NaN         Pvt Ltd
never
4                 STEM        >20        50-99  Funded Startup
4
...                ...        ...          ...             ...
...
```

```
19153       Humanities          14           NaN              NaN
1
19154             STEM          14           NaN              NaN
4
19155             STEM         >20         50-99          Pvt Ltd
4
19156              NaN          <1       500-999          Pvt Ltd
2
19157              NaN           2           NaN              NaN
1

       training_hours  target
0                  36     1.0
1                  47     0.0
2                  83     0.0
3                  52     1.0
4                   8     0.0
...               ...     ...
19153              42     1.0
19154              52     1.0
19155              44     0.0
19156              97     0.0
19157             127     0.0

[19158 rows x 14 columns]
```

Opening a csv file from URL

```
import requests
from io import StringIO

url =
'https://raw.githubusercontent.com/cs109/2014_data/master/countries.cs
v'
headers = {"User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X
10.14; rv:66.0) Gecko/20100101 Firefox/66.0"}
response = requests.get(url, headers=headers)
data = StringIO(response.text)

pd.read_csv(data)

       Country         Region
0      Algeria         AFRICA
1       Angola         AFRICA
2        Benin         AFRICA
3     Botswana         AFRICA
4      Burkina         AFRICA
..         ...            ...
189   Paraguay  SOUTH AMERICA
190       Peru  SOUTH AMERICA
```

```
191   Suriname  SOUTH AMERICA
192    Uruguay  SOUTH AMERICA
193  Venezuela  SOUTH AMERICA

[194 rows x 2 columns]
```

Sep and Names Parameter

- sep is used if the values are separated by tabs or semicolons or other separators.
- names can be provided is we want the specific column names or if the column names are not included in the file

```
pd.read_csv('Datasets/movie_titles_metadata.tsv', sep='\t',
names=['id', 'name', 'year', 'rating', 'votes', 'genres'])

       id                        name  year  rating     votes  \
0      m0  10 things i hate about you  1999     6.9   62847.0
1      m1  1492: conquest of paradise  1992     6.2   10421.0
2      m2                  15 minutes  2001     6.1   25854.0
3      m3       2001: a space odyssey  1968     8.4  163227.0
4      m4                    48 hrs.  1982     6.9   22289.0
..    ...                         ...   ...     ...       ...
612  m612                    watchmen  2009     7.8  135229.0
613  m613                         xxx  2002     5.6   53505.0
614  m614                      x-men  2000     7.4  122149.0
615  m615          young frankenstein  1974     8.0   57618.0
616  m616                   zulu dawn  1979     6.4    1911.0

                                                genres
0                                  ['comedy' 'romance']
1           ['adventure' 'biography' 'drama' 'history']
2                  ['action' 'crime' 'drama' 'thriller']
3                       ['adventure' 'mystery' 'sci-fi']
4        ['action' 'comedy' 'crime' 'drama' 'thriller']
..                                                 ...
612  ['action' 'crime' 'fantasy' 'mystery' 'sci-fi'...
613                       ['action' 'adventure' 'crime']
614                                  ['action' 'sci-fi']
615                                  ['comedy' 'sci-fi']
616     ['action' 'adventure' 'drama' 'history' 'war']

[617 rows x 6 columns]
```

index_col parameter

- setting a column as index

```
pd.read_csv('Datasets/aug_train.csv', index_col='enrollee_id')
```

```
                  city  city_development_index gender
relevent_experience  \
enrollee_id

8949         city_103                   0.920   Male  Has relevent
experience
29725         city_40                   0.776   Male   No relevent
experience
11561         city_21                   0.624    NaN   No relevent
experience
33241        city_115                   0.789    NaN   No relevent
experience
666          city_162                   0.767   Male  Has relevent
experience
...               ...                     ...    ...
...
7386         city_173                   0.878   Male   No relevent
experience
31398        city_103                   0.920   Male  Has relevent
experience
24576        city_103                   0.920   Male  Has relevent
experience
5756          city_65                   0.802   Male  Has relevent
experience
23834         city_67                   0.855    NaN   No relevent
experience

            enrolled_university education_level major_discipline
experience  \
enrollee_id

8949                no_enrollment        Graduate             STEM
>20
29725               no_enrollment        Graduate             STEM
15
11561            Full time course        Graduate             STEM
5
33241                         NaN        Graduate  Business Degree
<1
666                 no_enrollment         Masters             STEM
>20
...                           ...             ...              ...
...
7386                no_enrollment        Graduate       Humanities
14
31398               no_enrollment        Graduate             STEM
14
24576               no_enrollment        Graduate             STEM
>20
5756                no_enrollment     High School              NaN
```

```
<1
23834             no_enrollment  Primary School              NaN
2

            company_size    company_type last_new_job  training_hours
target
enrollee_id

8949                 NaN             NaN            1              36
1.0
29725              50-99         Pvt Ltd           >4              47
0.0
11561                NaN             NaN        never              83
0.0
33241                NaN         Pvt Ltd        never              52
1.0
666                50-99  Funded Startup            4               8
0.0
...                  ...             ...          ...             ...
...
7386                 NaN             NaN            1              42
1.0
31398                NaN             NaN            4              52
1.0
24576              50-99         Pvt Ltd            4              44
0.0
5756             500-999         Pvt Ltd            2              97
0.0
23834                NaN             NaN            1             127
0.0

[19158 rows x 13 columns]
```

Header parameter

if the header row(column names) are misplaced due to some reason then specific row can be used as header

```
pd.read_csv('Datasets/test.csv', header=1)

   0  enrollee_id      city  city_development_index gender  \
0  1        29725   city_40                   0.776   Male
1  2        11561   city_21                   0.624    NaN
2  3        33241  city_115                   0.789    NaN
3  4          666  city_162                   0.767   Male

       relevent_experience enrolled_university education_level  \
0   No relevent experience       no_enrollment        Graduate
1   No relevent experience    Full time course        Graduate
```

```
2   No relevent experience                  NaN        Graduate
3  Has relevent experience        no_enrollment         Masters

  major_discipline experience company_size    company_type 
last_new_job  \
0             STEM         15        50-99         Pvt Ltd 
>4
1             STEM          5          NaN             NaN 
never
2  Business Degree         <1          NaN         Pvt Ltd 
never
3             STEM        >20        50-99  Funded Startup 
4

   training_hours  target
0              47       0
1              83       0
2              52       1
3               8       0
```

usecols parameter

used for fetching specific columns

```
pd.read_csv('Datasets/aug_train.csv', usecols=['enrollee_id', 
'gender', 'education_level'])
```

```
       enrollee_id gender education_level
0             8949   Male        Graduate
1            29725   Male        Graduate
2            11561    NaN        Graduate
3            33241    NaN        Graduate
4              666   Male         Masters
...            ...    ...             ...
19153         7386   Male        Graduate
19154        31398   Male        Graduate
19155        24576   Male        Graduate
19156         5756   Male     High School
19157        23834    NaN  Primary School

[19158 rows x 3 columns]
```

skiprows/nrows parameter

- skiprows use for skipping specific rows
- nrows used for fetching n rows only

```
pd.read_csv('Datasets/aug_train.csv', skiprows=[1,5], nrows=100)
```

```
    enrollee_id      city  city_development_index gender  \
0         29725   city_40                   0.776   Male
1         11561   city_21                   0.624    NaN
2         33241  city_115                   0.789    NaN
3         21651  city_176                   0.764    NaN
4         28806  city_160                   0.920   Male
..          ...       ...                     ...    ...
95        11184   city_74                   0.579    NaN
96         7016   city_65                   0.802   Male
97         8695   city_11                   0.550   Male
98         6172   city_11                   0.550   Male
99        14672  city_173                   0.878    NaN

        relevent_experience enrolled_university education_level  \
0    No relevent experience       no_enrollment        Graduate
1    No relevent experience    Full time course        Graduate
2    No relevent experience                 NaN        Graduate
3   Has relevent experience    Part time course        Graduate
4   Has relevent experience       no_enrollment     High School
..                      ...                 ...             ...
95   No relevent experience    Full time course        Graduate
96  Has relevent experience       no_enrollment        Graduate
97  Has relevent experience       no_enrollment        Graduate
98  Has relevent experience       no_enrollment        Graduate
99   No relevent experience       no_enrollment         Masters

   major_discipline experience company_size    company_type
last_new_job  \
0              STEM         15        50-99         Pvt Ltd
>4
1              STEM          5          NaN             NaN
never
2   Business Degree         <1          NaN         Pvt Ltd
never
3              STEM         11          NaN             NaN
1
4               NaN          5        50-99  Funded Startup
1
..              ...        ...          ...             ...          .
..
95             STEM          2      100-500         Pvt Ltd
1
96             STEM          6        50-99         Pvt Ltd
2
97             STEM          6        10/49         Pvt Ltd
2
98             STEM          8      100-500         Pvt Ltd
1
99             STEM        >20          NaN             NaN
1
```

```
    training_hours  target
0               47     0.0
1               83     0.0
2               52     1.0
3               24     1.0
4               24     0.0
..             ...     ...
95              34     0.0
96              14     1.0
97              27     1.0
98              24     1.0
99             150     0.0

[100 rows x 14 columns]
```

encoding parameter

- if datasets have specific encoding then we have to pass it

```
pd.read_csv('Datasets/zomato.csv', encoding='latin-1')
```

```
      Restaurant ID           Restaurant Name  Country Code
City  \
0           6317637          Le Petit Souffle           162
Makati City
1           6304287          Izakaya Kikufuji           162
Makati City
2           6300002    Heat - Edsa Shangri-La           162
Mandaluyong City
3           6318506                      Ooma           162
Mandaluyong City
4           6314302               Sambo Kojin           162
Mandaluyong City
...             ...                       ...           ...
...
9546        5915730             NamlÛ± Gurme           208
ÛÁstanbul
9547        5908749            Ceviz AÛôacÛ±           208
ÛÁstanbul
9548        5915807                     Huqqa           208
ÛÁstanbul
9549        5916112              A)ô)ôk Kahve           208
ÛÁstanbul
9550        5927402  Walter's Coffee Roastery           208
ÛÁstanbul

                                                Address  \
0     Third Floor, Century City Mall, Kalayaan Avenu...
1     Little Tokyo, 2277 Chino Roces Avenue, Legaspi...
```

```
2     Edsa Shangri-La, 1 Garden Way, Ortigas, Mandal...
3     Third Floor, Mega Fashion Hall, SM Megamall, O...
4     Third Floor, Mega Atrium, SM Megamall, Ortigas...
...                                                ...
9546  Kemanke)ô Karamustafa Pa)ôa Mahallesi, RÛ±htÛ±...
9547  Ko)ôuyolu Mahallesi, Muhittin íìstí_ndaÛô Cadd...
9548  Kuruí_e)ôme Mahallesi, Muallim Naci Caddesi, N...
9549  Kuruí_e)ôme Mahallesi, Muallim Naci Caddesi, N...
9550  CafeaÛôa Mahallesi, BademaltÛ± Sokak, No 21/B,...

                                        Locality  \
0      Century City Mall, Poblacion, Makati City
1     Little Tokyo, Legaspi Village, Makati City
2     Edsa Shangri-La, Ortigas, Mandaluyong City
3         SM Megamall, Ortigas, Mandaluyong City
4         SM Megamall, Ortigas, Mandaluyong City
...                                          ...
9546                                    Karakí_y
9547                                   Ko)ôuyolu
9548                                Kuruí_e)ôme
9549                                Kuruí_e)ôme
9550                                        Moda

                                       Locality Verbose   Longitude  \
0     Century City Mall, Poblacion, Makati City, Mak...  121.027535
1     Little Tokyo, Legaspi Village, Makati City, Ma...  121.014101
2     Edsa Shangri-La, Ortigas, Mandaluyong City, Ma...  121.056831
3     SM Megamall, Ortigas, Mandaluyong City, Mandal...  121.056475
4     SM Megamall, Ortigas, Mandaluyong City, Mandal...  121.057508
...                                                 ...         ...
9546                                Karakí_y, ÛÁstanbul   28.977392
9547                               Ko)ôuyolu, ÛÁstanbul   29.041297
9548                             Kuruí_e)ôme, ÛÁstanbul   29.034640
9549                             Kuruí_e)ôme, ÛÁstanbul   29.036019
9550                                    Moda, ÛÁstanbul   29.026016

       Latitude                          Cuisines  ...
Currency  \
0     14.565443        French, Japanese, Desserts  ...  Botswana
Pula(P)
1     14.553708                          Japanese  ...  Botswana
Pula(P)
2     14.581404  Seafood, Asian, Filipino, Indian  ...  Botswana
Pula(P)
3     14.585318                   Japanese, Sushi  ...  Botswana
Pula(P)
4     14.584450                  Japanese, Korean  ...  Botswana
Pula(P)
...         ...                               ...  ...                 .
..
```

```
9546  41.022793                              Turkish  ...  Turkish
Lira(TL)
9547  41.009847   World Cuisine, Patisserie, Cafe  ...  Turkish
Lira(TL)
9548  41.055817            Italian, World Cuisine  ...  Turkish
Lira(TL)
9549  41.057979                   Restaurant Cafe  ...  Turkish
Lira(TL)
9550  40.984776                              Cafe  ...  Turkish
Lira(TL)

     Has Table booking Has Online delivery Is delivering now  \
0                  Yes                  No                No
1                  Yes                  No                No
2                  Yes                  No                No
3                   No                  No                No
4                  Yes                  No                No
...                ...                 ...               ...
9546                No                  No                No
9547                No                  No                No
9548                No                  No                No
9549                No                  No                No
9550                No                  No                No

     Switch to order menu Price range  Aggregate rating  Rating color
\
0                      No           3               4.8    Dark Green

1                      No           3               4.5    Dark Green

2                      No           4               4.4         Green

3                      No           4               4.9    Dark Green

4                      No           4               4.8    Dark Green

...                   ...         ...               ...           ...

9546                   No           3               4.1         Green

9547                   No           3               4.2         Green

9548                   No           4               3.7        Yellow

9549                   No           4               4.0         Green

9550                   No           2               4.0         Green


     Rating text Votes
0      Excellent   314
```

```
1       Excellent   591
2       Very Good   270
3       Excellent   365
4       Excellent   229
...           ...   ...
9546    Very Good   788
9547    Very Good  1034
9548         Good   661
9549    Very Good   901
9550    Very Good   591

[9551 rows x 21 columns]
```

skip bad lines

- if some rows has issues like extra column value then such rows would be automatically skipped

```
pd.read_csv('Datasets/test1.csv', header=1, on_bad_lines='skip')

   0  enrollee_id      city  city_development_index gender  \
0  1        29725   city_40                   0.776   Male
1  3        33241  city_115                   0.789    NaN
2  4          666  city_162                   0.767   Male

       relevent_experience enrolled_university education_level  \
0   No relevent experience       no_enrollment        Graduate
1   No relevent experience                 NaN        Graduate
2  Has relevent experience       no_enrollment         Masters

  major_discipline experience company_size    company_type
last_new_job  \
0             STEM         15        50-99         Pvt Ltd
>4
1  Business Degree         <1          NaN         Pvt Ltd
never
2             STEM        >20        50-99  Funded Startup
4

   training_hours  target
0              47       0
1              52       1
2               8       0
```

dtype parameter

used in case some columns has different data type and we want to change it.

In below example target column has by default values in float dtype but it can be easily represneted in int dtype that will save memory

```
pd.read_csv('Datasets/aug_train.csv', dtype={'target':int})

       enrollee_id      city  city_development_index gender  \
0             8949  city_103                   0.920   Male   
1            29725   city_40                   0.776   Male   
2            11561   city_21                   0.624    NaN   
3            33241  city_115                   0.789    NaN   
4              666  city_162                   0.767   Male   
...            ...       ...                     ...    ...   
19153         7386  city_173                   0.878   Male   
19154        31398  city_103                   0.920   Male   
19155        24576  city_103                   0.920   Male   
19156         5756   city_65                   0.802   Male   
19157        23834   city_67                   0.855    NaN   

           relevent_experience enrolled_university education_level  \
0      Has relevent experience       no_enrollment        Graduate   
1       No relevent experience       no_enrollment        Graduate   
2       No relevent experience    Full time course        Graduate   
3       No relevent experience                 NaN        Graduate   
4      Has relevent experience       no_enrollment         Masters   
...                        ...                 ...             ...   
19153   No relevent experience       no_enrollment        Graduate   
19154  Has relevent experience       no_enrollment        Graduate   
19155  Has relevent experience       no_enrollment        Graduate   
19156  Has relevent experience       no_enrollment     High School   
19157   No relevent experience       no_enrollment  Primary School   

      major_discipline experience company_size    company_type
last_new_job  \
0                 STEM        >20          NaN             NaN
1   
1                 STEM         15        50-99         Pvt Ltd
>4   
2                 STEM          5          NaN             NaN
never   
3      Business Degree         <1          NaN         Pvt Ltd
never   
4                 STEM        >20        50-99  Funded Startup
4   
...                ...        ...          ...             ...
...   
19153       Humanities         14          NaN             NaN
1   
19154             STEM         14          NaN             NaN
4   
19155             STEM        >20        50-99         Pvt Ltd
4   
19156              NaN         <1      500-999         Pvt Ltd
2   
```

```
19157                  NaN             2           NaN                NaN
1

       training_hours  target
0                  36       1
1                  47       0
2                  83       0
3                  52       1
4                   8       0
...               ...     ...
19153              42       1
19154              52       1
19155              44       0
19156              97       0
19157             127       0

[19158 rows x 14 columns]
```

Handling Dates

- generally read_csv fetches the date column in object dtype which further need to explicitely convert to datetime64 dtype but parse_dates directly converts.

```
pd.read_csv('Datasets/ipl-matches.csv', parse_dates=['Date']).info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 950 entries, 0 to 949
Data columns (total 20 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   ID                950 non-null    int64
 1   City              899 non-null    object
 2   Date              950 non-null    datetime64[ns]
 3   Season            950 non-null    object
 4   MatchNumber       950 non-null    object
 5   Team1             950 non-null    object
 6   Team2             950 non-null    object
 7   Venue             950 non-null    object
 8   TossWinner        950 non-null    object
 9   TossDecision      950 non-null    object
 10  SuperOver         946 non-null    object
 11  WinningTeam       946 non-null    object
 12  WonBy             950 non-null    object
 13  Margin            932 non-null    float64
 14  method            19 non-null     object
 15  Player_of_Match   946 non-null    object
 16  Team1Players      950 non-null    object
 17  Team2Players      950 non-null    object
 18  Umpire1           950 non-null    object
 19  Umpire2           950 non-null    object
```

```
dtypes: datetime64[ns](1), float64(1), int64(1), object(17)
memory usage: 148.6+ KB
```

Covertors

Used to apply function to specific columns

In below example we are using rename function on Team1 and Team2 column it shorts the name

```python
def rename(name):
    l = name.split(' ')
    short_form = ''
    for i in l:
        short_form += i[0][0].upper()
    return short_form

rename('Dilkhush Singh')
```

```
'DS'
```

```python
pd.read_csv('Datasets/ipl-matches.csv', converters={'Team1':rename,
'Team2':rename})
```

```
          ID        City        Date   Season  MatchNumber Team1 Team2  \
0    1312200   Ahmedabad  2022-05-29     2022        Final    RR    GT

1    1312199   Ahmedabad  2022-05-27     2022  Qualifier 2   RCB    RR

2    1312198     Kolkata  2022-05-25     2022   Eliminator   RCB   LSG

3    1312197     Kolkata  2022-05-24     2022  Qualifier 1    RR    GT

4    1304116      Mumbai  2022-05-22     2022           70    SH    PK

..       ...         ...         ...      ...          ...   ...   ...

945   335986     Kolkata  2008-04-20  2007/08            4   KKR    DC

946   335985      Mumbai  2008-04-20  2007/08            5    MI   RCB

947   335984       Delhi  2008-04-19  2007/08            3    DD    RR

948   335983  Chandigarh  2008-04-19  2007/08            2   KXP   CSK

949   335982   Bangalore  2008-04-18  2007/08            1   RCB   KKR


                                        Venue
TossWinner  \
0              Narendra Modi Stadium, Ahmedabad             Rajasthan
```

```
Royals
1               Narendra Modi Stadium, Ahmedabad             Rajasthan
Royals
2                          Eden Gardens, Kolkata         Lucknow Super
Giants
3                          Eden Gardens, Kolkata               Gujarat
Titans
4                        Wankhede Stadium, Mumbai          Sunrisers
Hyderabad
..                                           ...
...
945                                 Eden Gardens              Deccan
Chargers
946                             Wankhede Stadium               Mumbai
Indians
947                             Feroz Shah Kotla             Rajasthan
Royals
948  Punjab Cricket Association Stadium, Mohali          Chennai Super
Kings
949                       M Chinnaswamy Stadium  Royal Challengers
Bangalore

    TossDecision SuperOver                  WinningTeam    WonBy
Margin  \
0            bat         N               Gujarat Titans  Wickets
7.0
1          field         N            Rajasthan Royals  Wickets
7.0
2          field         N  Royal Challengers Bangalore     Runs
14.0
3          field         N               Gujarat Titans  Wickets
7.0
4            bat         N                 Punjab Kings  Wickets
5.0
..           ...       ...                          ...      ...      .
..
945          bat         N        Kolkata Knight Riders  Wickets
5.0
946          bat         N  Royal Challengers Bangalore  Wickets
5.0
947          bat         N             Delhi Daredevils  Wickets
9.0
948          bat         N           Chennai Super Kings     Runs
33.0
949        field         N        Kolkata Knight Riders     Runs
140.0

    method Player_of_Match
Team1Players  \
```

```
0      NaN       HH Pandya  ['YBK Jaiswal', 'JC Buttler', 'SV Samson',
'D ...
1      NaN      JC Buttler  ['V Kohli', 'F du Plessis', 'RM Patidar',
'GJ ...
2      NaN      RM Patidar  ['V Kohli', 'F du Plessis', 'RM Patidar',
'GJ ...
3      NaN       DA Miller  ['YBK Jaiswal', 'JC Buttler', 'SV Samson',
'D ...
4      NaN   Harpreet Brar  ['PK Garg', 'Abhishek Sharma', 'RA
Tripathi', ...
..     ...             ...
...
945    NaN       DJ Hussey  ['WP Saha', 'BB McCullum', 'RT Ponting',
'SC G...
946    NaN      MV Boucher  ['L Ronchi', 'ST Jayasuriya', 'DJ
Thornely', '...
947    NaN     MF Maharoof  ['G Gambhir', 'V Sehwag', 'S Dhawan', 'MK
Tiwa...
948    NaN      MEK Hussey  ['K Goel', 'JR Hopes', 'KC Sangakkara',
'Yuvra...
949    NaN     BB McCullum  ['R Dravid', 'W Jaffer', 'V Kohli', 'JH
Kallis...

                                          Team2Players        Umpire1
\
0    ['WP Saha', 'Shubman Gill', 'MS Wade', 'HH Pan...    CB Gaffaney

1    ['YBK Jaiswal', 'JC Buttler', 'SV Samson', 'D ...    CB Gaffaney

2    ['Q de Kock', 'KL Rahul', 'M Vohra', 'DJ Hooda...  J Madanagopal

3    ['WP Saha', 'Shubman Gill', 'MS Wade', 'HH Pan...   BNJ Oxenford

4    ['JM Bairstow', 'S Dhawan', 'M Shahrukh Khan',...   AK Chaudhary

..                                                 ...            ...

945  ['AC Gilchrist', 'Y Venugopal Rao', 'VVS Laxma...      BF Bowden

946  ['S Chanderpaul', 'R Dravid', 'LRPL Taylor', '...       SJ Davis

947  ['T Kohli', 'YK Pathan', 'SR Watson', 'M Kaif'...      Aleem Dar

948  ['PA Patel', 'ML Hayden', 'MEK Hussey', 'MS Dh...      MR Benson

949  ['SC Ganguly', 'BB McCullum', 'RT Ponting', 'D...      Asad Rauf


            Umpire2
0       Nitin Menon
```

```
1        Nitin Menon
2           MA Gough
3          VK Sharma
4      NA Patwardhan
..               ...
945      K Hariharan
946        DJ Harper
947   GA Pratapkumar
948       SL Shastri
949      RE Koertzen

[950 rows x 20 columns]
```

na_values parameter

In some files if the missing values is represented in the form of ? or something like 00000 then to treat them as Nan values we use na_values parameter

```
pd.read_csv('Datasets/aug_train.csv', na_values=['Male'])

       enrollee_id      city  city_development_index gender  \
0             8949  city_103                   0.920    NaN
1            29725   city_40                   0.776    NaN
2            11561   city_21                   0.624    NaN
3            33241  city_115                   0.789    NaN
4              666  city_162                   0.767    NaN
...            ...       ...                     ...    ...
19153         7386  city_173                   0.878    NaN
19154        31398  city_103                   0.920    NaN
19155        24576  city_103                   0.920    NaN
19156         5756   city_65                   0.802    NaN
19157        23834   city_67                   0.855    NaN

           relevent_experience enrolled_university education_level  \
0      Has relevent experience       no_enrollment        Graduate
1       No relevent experience       no_enrollment        Graduate
2       No relevent experience    Full time course        Graduate
3       No relevent experience                 NaN        Graduate
4      Has relevent experience       no_enrollment         Masters
...                        ...                 ...             ...
19153   No relevent experience       no_enrollment        Graduate
19154  Has relevent experience       no_enrollment        Graduate
19155  Has relevent experience       no_enrollment        Graduate
19156  Has relevent experience       no_enrollment     High School
19157   No relevent experience       no_enrollment  Primary School

      major_discipline experience company_size    company_type
last_new_job  \
0                 STEM        >20          NaN             NaN
```

```
1
1                  STEM           15         50-99          Pvt Ltd
>4
2                  STEM            5           NaN              NaN
never
3       Business Degree           <1           NaN          Pvt Ltd
never
4                  STEM          >20         50-99   Funded Startup
4
...                 ...          ...           ...              ...
...
19153        Humanities           14           NaN              NaN
1
19154              STEM           14           NaN              NaN
4
19155              STEM          >20         50-99          Pvt Ltd
4
19156               NaN           <1       500-999          Pvt Ltd
2
19157               NaN            2           NaN              NaN
1

       training_hours  target
0                  36     1.0
1                  47     0.0
2                  83     0.0
3                  52     1.0
4                   8     0.0
...               ...     ...
19153              42     1.0
19154              52     1.0
19155              44     0.0
19156              97     0.0
19157             127     0.0

[19158 rows x 14 columns]
```

Loading big dataset in chunks

If a dataset is so huge that loading whole in memory is not possible then we can use chunks to break it and load into the memory.

```
chunks = pd.read_csv('Datasets/aug_train.csv', chunksize=5000)

for chunk in chunks:
    print(chunk.shape)
```

```
(5000, 14)
(5000, 14)
```

```
(5000, 14)
(4158, 14)
```

Working with Excel files

read_excel is very similar to read_csv

Opening a local excel file

```
pd.read_excel('output.xlsx')
```

Opening other sheets

```
pd.read_excel('output.xlsx', sheet_name='sheet_2')
```

Working with text files

```
pd.read_csv('https://storage.googleapis.com/kagglesdsdata/datasets/
2735/4525/S08_question_answer_pairs.txt?X-Goog-Algorithm=GOOG4-RSA-
SHA256&X-Goog-Credential=gcp-kaggle-com%40kaggle-
161607.iam.gserviceaccount.com%2F20240807%2Fauto%2Fstorage
%2Fgoog4_request&X-Goog-Date=20240807T071300Z&X-Goog-Expires=259200&X-
Goog-SignedHeaders=host&X-Goog-
Signature=7011b201b24d80b39cef4f7f69fac37ede519fd5ffe8827dea2025270778
8f5dfa0e074ea7617e922ec33e08eec4f8eb7ce87a5829053e1454aef57f8c4ce2466a
95f00ff2b0b797132c4e44812f4fb665c326c47d30de2f0497fdfd236d64cbb944fd5b
14738950191e911e0f1270d57371dbf02a46f13c9b096a40fe4540b5e726a60035c995
75e5278268ddd274691bebf868aa846e7827167331daed4f6a727b15f5e143575abfa6
ff632152e9c96220bdc7c0303fc8a3c6ccc01be3b991250737e5f9cf0b38c4c8692f90
0394384c539b54378e69e37a73b29f0faf404c2efab8cec6f54c5e65d0ef73b1bb34d8
58dabb4bafc4ba6f60a85850c4ef0307', sep='\t')
```

```
         ArticleTitle
Question  \
0     Abraham_Lincoln  Was Abraham Lincoln the sixteenth President
of...
1     Abraham_Lincoln  Was Abraham Lincoln the sixteenth President
of...
2     Abraham_Lincoln  Did Lincoln sign the National Banking Act of
1...
3     Abraham_Lincoln  Did Lincoln sign the National Banking Act of
1...
4     Abraham_Lincoln                      Did his mother die of
pneumonia?
...               ...                                             .
..
1710   Woodrow_Wilson  Was Wilson president of the American
Political...
1711   Woodrow_Wilson  Did he not cast his ballot for John M.
```

```
Palmer ...
1712   Woodrow_Wilson  Did Wilson not spend 1914 through the
beginnin...
1713   Woodrow_Wilson  Was Wilson , a staunch opponent of
antisemitis...
1714   Woodrow_Wilson                             What happened in
1917?

                                                   Answer  \
0                                                     yes
1                                                    Yes.
2                                                     yes
3                                                    Yes.
4                                                      no
...                                                   ...
1710                                                  Yes
1711                                                  Yes
1712                                                  Yes
1713                                                  Yes
1714  raised billions through Liberty loans, imposed...

     DifficultyFromQuestioner DifficultyFromAnswerer  ArticleFile
0                        easy                   easy  S08_set3_a4
1                        easy                   easy  S08_set3_a4
2                        easy                 medium  S08_set3_a4
3                        easy                   easy  S08_set3_a4
4                        easy                 medium  S08_set3_a4
...                       ...                    ...          ...
1710                      NaN                   easy  S08_set3_a8
1711                      NaN                   easy  S08_set3_a8
1712                      NaN                   easy  S08_set3_a8
1713                      NaN                   easy  S08_set3_a8
1714                      NaN                 medium  S08_set3_a8

[1715 rows x 6 columns]
```

Working with JSON files(API)

Opening local JSON file

```
pd.read_json('Datasets/train.json')

          id      cuisine
ingredients
0      10259        greek  [romaine lettuce, black olives, grape
tomatoes...
1      25693  southern_us  [plain flour, ground pepper, salt,
tomatoes, g...
2      20130     filipino  [eggs, pepper, salt, mayonaise, cooking
```

```
oil, g...
3      22213       indian                [water, vegetable oil, wheat, salt]
4      13162       indian  [black pepper, shallots, cornflour, cayenne pe...
...      ...          ...                                               ...
39769  29109        irish  [light brown sugar, granulated sugar, butter, ...
39770  11462      italian  [KRAFT Zesty Italian Dressing, purple onion, b...
39771   2238        irish  [eggs, citrus fruit, raisins, sourdough starte...
39772  41882      chinese  [boneless chicken skinless thigh, minced garli...
39773   2362      mexican  [green chile, jalapeno chilies, onions, ground...

[39774 rows x 3 columns]
```

Opening JSON file from API

```
pd.read_json('https://api.exchangerate-api.com/v4/latest/INR')
```

```
                                provider  \
INR  https://www.exchangerate-api.com
AED  https://www.exchangerate-api.com
AFN  https://www.exchangerate-api.com
ALL  https://www.exchangerate-api.com
AMD  https://www.exchangerate-api.com
..                                ...
XPF  https://www.exchangerate-api.com
YER  https://www.exchangerate-api.com
ZAR  https://www.exchangerate-api.com
ZMW  https://www.exchangerate-api.com
ZWL  https://www.exchangerate-api.com

                          WARNING_UPGRADE_TO_V6  \
INR  https://www.exchangerate-api.com/docs/free
AED  https://www.exchangerate-api.com/docs/free
AFN  https://www.exchangerate-api.com/docs/free
ALL  https://www.exchangerate-api.com/docs/free
AMD  https://www.exchangerate-api.com/docs/free
..                                          ...
XPF  https://www.exchangerate-api.com/docs/free
YER  https://www.exchangerate-api.com/docs/free
ZAR  https://www.exchangerate-api.com/docs/free
ZMW  https://www.exchangerate-api.com/docs/free
ZWL  https://www.exchangerate-api.com/docs/free
```

```
                                          terms base       date  \
INR  https://www.exchangerate-api.com/terms  INR 2024-08-07
AED  https://www.exchangerate-api.com/terms  INR 2024-08-07
AFN  https://www.exchangerate-api.com/terms  INR 2024-08-07
ALL  https://www.exchangerate-api.com/terms  INR 2024-08-07
AMD  https://www.exchangerate-api.com/terms  INR 2024-08-07
..                                      ...  ...        ...
XPF  https://www.exchangerate-api.com/terms  INR 2024-08-07
YER  https://www.exchangerate-api.com/terms  INR 2024-08-07
ZAR  https://www.exchangerate-api.com/terms  INR 2024-08-07
ZMW  https://www.exchangerate-api.com/terms  INR 2024-08-07
ZWL  https://www.exchangerate-api.com/terms  INR 2024-08-07

     time_last_updated   rates
INR         1722988802  1.0000
AED         1722988802  0.0437
AFN         1722988802  0.8450
ALL         1722988802  1.0900
AMD         1722988802  4.6200
..                 ...     ...
XPF         1722988802  1.3000
YER         1722988802  2.9800
ZAR         1722988802  0.2200
ZMW         1722988802  0.3100
ZWL         1722988802  0.0448

[162 rows x 7 columns]
```

Working with SQL

Parameters like index_col, parse_dates, chunksize can also be used.

```
import mysql.connector

conn = mysql.connector.connect(host='localhost', user='root',
password='', database='world')

pd.read_sql_query('SELECT * FROM city', conn)

C:\Users\DILKHUSH\AppData\Local\Temp\ipykernel_12028\929584838.py:1:
UserWarning: pandas only supports SQLAlchemy connectable
(engine/connection) or database string URI or sqlite3 DBAPI2
connection. Other DBAPI2 objects are not tested. Please consider using
SQLAlchemy.
  pd.read_sql_query('SELECT * FROM city', conn)

        ID           Name CountryCode       District  Population
0        1          Kabul         AFG          Kabol     1780000
1        2       Qandahar         AFG       Qandahar      237500
2        3          Herat         AFG          Herat      186800
```

```
3        4  Mazar-e-Sharif         AFG          Balkh      127800
4        5       Amsterdam         NLD  Noord-Holland      731200
...    ...             ...         ...            ...         ...
4074  4075      Khan Yunis         PSE     Khan Yunis      123175
4075  4076          Hebron         PSE         Hebron      119401
4076  4077        Jabaliya         PSE     North Gaza      113901
4077  4078          Nablus         PSE         Nablus      100231
4078  4079           Rafah         PSE          Rafah       92020

[4079 rows x 5 columns]
```

```
pd.read_sql_query('SELECT * FROM country WHERE LifeExpectancy>50',
conn)
```

```
C:\Users\DILKHUSH\AppData\Local\Temp\ipykernel_12028\1890645157.py:1:
UserWarning: pandas only supports SQLAlchemy connectable
(engine/connection) or database string URI or sqlite3 DBAPI2
connection. Other DBAPI2 objects are not tested. Please consider using
SQLAlchemy.
  pd.read_sql_query('SELECT * FROM country WHERE LifeExpectancy>50',
conn)
```

```
    Code                   Name      Continent           Region
SurfaceArea  \
0    ABW                  Aruba  North America        Caribbean
193.0
1    AIA               Anguilla  North America        Caribbean
96.0
2    ALB                Albania         Europe  Southern Europe
28748.0
3    AND                Andorra         Europe  Southern Europe
468.0
4    ANT  Netherlands Antilles  North America        Caribbean
800.0
..   ...                    ...            ...              ...
...
189  VUT                Vanuatu        Oceania        Melanesia
12189.0
190  WSM                  Samoa        Oceania        Polynesia
2831.0
191  YEM                  Yemen           Asia      Middle East
527968.0
192  YUG             Yugoslavia         Europe  Southern Europe
102173.0
193  ZAF           South Africa         Africa  Southern Africa
1221037.0

     IndepYear  Population  LifeExpectancy      GNP    GNPOld  \
0          NaN      103000            78.4    828.0     793.0
1          NaN        8000            76.1     63.2       NaN
```

```
2       1912.0     3401200            71.6    3205.0    2500.0
3       1278.0       78000            83.5    1630.0       NaN
4          NaN      217000            74.7    1941.0       NaN
..         ...         ...             ...       ...       ...
189     1980.0      190000            60.6     261.0     246.0
190     1962.0      180000            69.2     141.0     157.0
191     1918.0    18112000            59.8    6041.0    5729.0
192     1918.0    10640000            72.4   17000.0       NaN
193     1910.0    40377000            51.1  116729.0  129092.0

                LocalName                                
GovernmentForm  \
0                   Aruba  Nonmetropolitan Territory of The
Netherlands
1                Anguilla                Dependent Territory of the
UK
2               Shqipëria                                
Republic
3                 Andorra                 Parliamentary
Coprincipality
4    Nederlandse Antillen  Nonmetropolitan Territory of The
Netherlands
..                    ...                                           ..
.
189               Vanuatu                                
Republic
190                 Samoa                        Parlementary
Monarchy
191              Al-Yaman                                
Republic
192            Jugoslavija                            Federal
Republic
193          South Africa                                
Republic

                HeadOfState  Capital Code2
0                   Beatrix      129    AW
1              Elisabeth II       62    AI
2            Rexhep Mejdani       34    AL
3                                 55    AD
4                   Beatrix       33    AN
..                      ...      ...   ...
189               John Bani     3537    VU
190  Malietoa Tanumafili II     3169    WS
191      Ali Abdallah Salih     1780    YE
192      Vojislav Koštunica     1792    YU
193             Thabo Mbeki      716    ZA

[194 rows x 15 columns]
```

Pandas Export

- to_csv
- to_excel
- to_html
- to_json
- to_sql

to_csv

```
df = pd.read_csv('Datasets/IPL_Ball_by_Ball.csv')
df.head()
```

```
   match_id  inning         batting_team                 bowling_team
over  \
0         1       1  Sunrisers Hyderabad  Royal Challengers Bangalore
1
1         1       1  Sunrisers Hyderabad  Royal Challengers Bangalore
1
2         1       1  Sunrisers Hyderabad  Royal Challengers Bangalore
1
3         1       1  Sunrisers Hyderabad  Royal Challengers Bangalore
1
4         1       1  Sunrisers Hyderabad  Royal Challengers Bangalore
1

   ball    batsman non_striker    bowler  is_super_over  ...  bye_runs
\
0     1  DA Warner    S Dhawan  TS Mills              0  ...         0

1     2  DA Warner    S Dhawan  TS Mills              0  ...         0

2     3  DA Warner    S Dhawan  TS Mills              0  ...         0

3     4  DA Warner    S Dhawan  TS Mills              0  ...         0

4     5  DA Warner    S Dhawan  TS Mills              0  ...         0


   legbye_runs  noball_runs  penalty_runs  batsman_runs  extra_runs  \
0            0            0             0             0           0
1            0            0             0             0           0
2            0            0             0             4           0
3            0            0             0             0           0
4            0            0             0             0           2

   total_runs  player_dismissed dismissal_kind fielder
0           0               NaN            NaN     NaN
1           0               NaN            NaN     NaN
2           4               NaN            NaN     NaN
```

```
3           0                NaN            NaN     NaN
4           2                NaN            NaN     NaN

[5 rows x 21 columns]
```

```python
temp = df.groupby('batsman')['batsman_runs'].sum().reset_index()

temp.to_csv('batsman_runs.csv', index=False)

batsman_vs_team = df.pivot_table(index='batsman',
columns='bowling_team', values='batsman_runs', aggfunc='sum')
```

to_excel

```python
! pip install openpyxl # Require for to_excel
```

```
Defaulting to user installation because normal site-packages is not
writeable
Collecting openpyxl
  Downloading openpyxl-3.1.5-py2.py3-none-any.whl.metadata (2.5 kB)
Collecting et-xmlfile (from openpyxl)
  Downloading et_xmlfile-1.1.0-py3-none-any.whl.metadata (1.8 kB)
Downloading openpyxl-3.1.5-py2.py3-none-any.whl (250 kB)
Downloading et_xmlfile-1.1.0-py3-none-any.whl (4.7 kB)
Installing collected packages: et-xmlfile, openpyxl
Successfully installed et-xmlfile-1.1.0 openpyxl-3.1.5
```

```python
temp.to_excel('batsman_runs.xlsx', index=False)
```

Multiple sheets

```python
with pd.ExcelWriter('ipl.xlsx') as writer:
    df.to_excel(writer, sheet_name='Batsman_runs')
    batsman_vs_team.to_excel(writer, sheet_name='Batsman_vs_team')
```

to_html

```python
sixes_heatmap = df[(df['batsman_runs']==6) & (df['ball'] <
7)].pivot_table(index='over', columns='ball', values='batsman_runs',
aggfunc='count')

sixes_heatmap.to_html('sixes_heatmap.html')
```

to_json

```python
batsman_runs = df.groupby(['batting_team', 'batsman'])
['batsman_runs'].sum().unstack()

batsman_runs.to_json('batsman_runs.json', indent=4)
```

to_sql

```
import pymysql
from sqlalchemy import create_engine

# {root}:{password}@{url}/{database}

engine = create_engine('mysql+pymysql://root:@localhost/test')

df.to_sql('ipl_delivery', con=engine, if_exists='append')

179078

temp.to_sql('batsman_runs', con=engine, if_exists='append')

516
```

Handling Data From API

```
import requests

url = "https://imdb-top-100-movies.p.rapidapi.com/"

headers = {
      "x-rapidapi-key":
"3adaf97e43msh566e2d44a6cf0bap11e68cjsn4df1ba89cd00",
      "x-rapidapi-host": "imdb-top-100-movies.p.rapidapi.com"
}

response = requests.get(url, headers=headers)

df = pd.DataFrame(response.json())
df

    rank                      title  \
0      1   The Shawshank Redemption
1      2              The Godfather
2      3            The Dark Knight
3      4     The Godfather Part II
4      5               12 Angry Men
..   ...                        ...
95    96             Reservoir Dogs
96    97                     Ikiru
97    98         Lawrence of Arabia
98    99               Citizen Kane
99   100                          M

                                          description  \
0   Two imprisoned men bond over a number of years...
1   The aging patriarch of an organized crime dyna...
```

```
2   When the menace known as the Joker wreaks havo...
3   The early life and career of Vito Corleone in ...
4   The jury in a New York City murder trial is fr...
..                                               ...
95  When a simple jewelry heist goes horribly wron...
96  A bureaucrat tries to find meaning in his life...
97  The story of T.E. Lawrence, the English office...
98  Following the death of publishing tycoon Charl...
99  When the police in a German city are unable to...

                                               image  \
0   https://m.media-amazon.com/images/M/MV5BMDFkYT...
1   https://m.media-amazon.com/images/M/MV5BM2MyNj...
2   https://m.media-amazon.com/images/M/MV5BMTMxNT...
3   https://m.media-amazon.com/images/M/MV5BMWMwMG...
4   https://m.media-amazon.com/images/M/MV5BMWU4N2...
..                                               ...
95  https://m.media-amazon.com/images/M/MV5BZmExNm...
96  https://m.media-amazon.com/images/M/MV5BYWM1Ym...
97  https://m.media-amazon.com/images/M/MV5BYWY5Zj...
98  https://m.media-amazon.com/images/M/MV5BYjBiOT...
99  https://m.media-amazon.com/images/M/MV5BODA4OD...

                                           big_image  \
0   https://m.media-amazon.com/images/M/MV5BMDFkYT...
1   https://m.media-amazon.com/images/M/MV5BM2MyNj...
2   https://m.media-amazon.com/images/M/MV5BMTMxNT...
3   https://m.media-amazon.com/images/M/MV5BMWMwMG...
4   https://m.media-amazon.com/images/M/MV5BMWU4N2...
..                                               ...
95  https://m.media-amazon.com/images/M/MV5BZmExNm...
96  https://m.media-amazon.com/images/M/MV5BYWM1Ym...
97  https://m.media-amazon.com/images/M/MV5BYWY5Zj...
98  https://m.media-amazon.com/images/M/MV5BYjBiOT...
99  https://m.media-amazon.com/images/M/MV5BODA4OD...

                              genre  \
0                           [Drama]
1                    [Crime, Drama]
2            [Action, Crime, Drama]
3                    [Crime, Drama]
4                    [Crime, Drama]
..                              ...
95                [Crime, Thriller]
96                          [Drama]
97  [Adventure, Biography, Drama]
98                 [Drama, Mystery]
99     [Crime, Mystery, Thriller]

                                           thumbnail rating      id
```

```
year  \
0   https://m.media-amazon.com/images/M/MV5BMDFkYT...    9.3    top1
1994
1   https://m.media-amazon.com/images/M/MV5BM2MyNj...    9.2    top2
1972
2   https://m.media-amazon.com/images/M/MV5BMTMxNT...    9.0    top3
2008
3   https://m.media-amazon.com/images/M/MV5BMWMwMG...    9.0    top4
1974
4   https://m.media-amazon.com/images/M/MV5BMWU4N2...    9.0    top5
1957
..                                                ...    ...     ...
...
95  https://m.media-amazon.com/images/M/MV5BZmExNm...    8.3   top96
1992
96  https://m.media-amazon.com/images/M/MV5BYWM1Ym...    8.3   top97
1952
97  https://m.media-amazon.com/images/M/MV5BYWY5Zj...    8.3   top98
1962
98  https://m.media-amazon.com/images/M/MV5BYjBiOT...    8.3   top99
1941
99  https://m.media-amazon.com/images/M/MV5BODA4OD...    8.3  top100
1931

       imdbid                             imdb_link
0   tt0111161  https://www.imdb.com/title/tt0111161
1   tt0068646  https://www.imdb.com/title/tt0068646
2   tt0468569  https://www.imdb.com/title/tt0468569
3   tt0071562  https://www.imdb.com/title/tt0071562
4   tt0050083  https://www.imdb.com/title/tt0050083
..        ...                                   ...
95  tt0105236  https://www.imdb.com/title/tt0105236
96  tt0044741  https://www.imdb.com/title/tt0044741
97  tt0056172  https://www.imdb.com/title/tt0056172
98  tt0033467  https://www.imdb.com/title/tt0033467
99  tt0022100  https://www.imdb.com/title/tt0022100

[100 rows x 12 columns]
```

`Note` - Data is also gathered by Web Scrapping

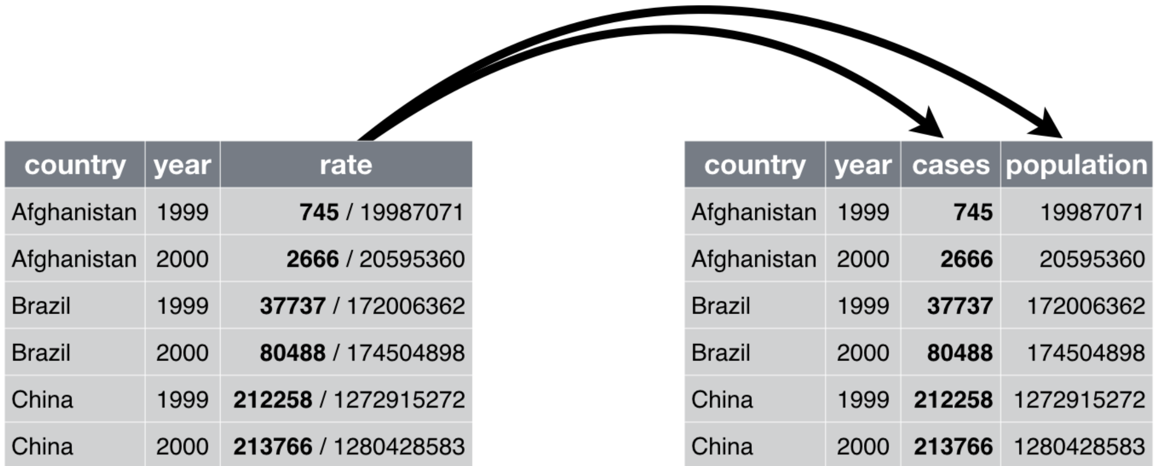
2b. Data Assesing

In this step, the data is to be understood more deeply. Before implementing methods to clean it, you will definitely need to have a better idea about what the data is about.

Types of Unclean Data

There are 2 kinds of unclean data

- `Dirty Data (Data with Quality issues)`: Dirty data, also known as low quality data. Low quality data has content issues.
  - Duplicated data
  - Missing Data
  - Corrupt Data
  - Inaccurate Data
- `Messy Data (Data with tidiness issues)`: Messy data, also known as untidy data. Untidy data has structural issues.Tidy data has the following properties:
  - Each variable forms a column
  - Each observation forms a row
  - Each observational unit forms a table



country	year	rate
Afghanistan	1999	**745** / 19987071
Afghanistan	2000	**2666** / 20595360
Brazil	1999	**37737** / 172006362
Brazil	2000	**80488** / 174504898
China	1999	**212258** / 1272915272
China	2000	**213766** / 1280428583

country	year	cases	population
Afghanistan	1999	**745**	19987071
Afghanistan	2000	**2666**	20595360
Brazil	1999	**37737**	172006362
Brazil	2000	**80488**	174504898
China	1999	**212258**	1272915272
China	2000	**213766**	1280428583

table3

Importing Libraries

```
import numpy as np
import pandas as pd
```

Loading Datasets

```
patients = pd.read_csv('Datasets/patients.csv')
treatments = pd.read_csv('Datasets/treatments.csv')
treatments_cut = pd.read_csv('Datasets/treatments_cut.csv')
adverse_reactions = pd.read_csv('Datasets/adverse_reactions.csv')
```

Steps of Data Accessing

1. Write Summary of Data

This is a dataset about 500 patients of which 350 patients participated in a clinical trial. None of the patients were using Novodra (a popular injectable insulin) or Auralin (the oral insulin being researched) as their primary source of insulin before. All were experiencing elevated hba1c levels.

All 350 patients were treated with Novodra to establish a baseline hba1c level and isulin dose. After 4 weeks, which isn't enough time to capture all the change in hba1c that can be attributed by the switch to Auralin or Novodra:

- 175 patients switched to Auralin for 24 weeks.
- 175 patients continued using Novodra for 24 weeks.

Data about patients feeling some adverse effects is also recorded.

2. Write Column Descriptions of every Dataset

Table -> `patients`:

- `patient_id`: the unique identifier for each patient in the Master Patient Index (i.e. patient database) of the pharmaceutical company that is producing Auralin
- `assigned_sex`: the assigned sex of each patient at birth (male or female)
- `given_name`: the given name (i.e. first name) of each patient
- `surname`: the surname (i.e. last name) of each patient
- `address`: the main address for each patient
- `city`: the corresponding city for the main address of each patient
- `state`: the corresponding state for the main address of each patient
- `zip_code`: the corresponding zip code for the main address of each patient
- `country`: the corresponding country for the main address of each patient (all United states for this clinical trial)
- `contact`: phone number and email information for each patient
- `birthdate`: the date of birth of each patient (month/day/year). The inclusion criteria for this clinical trial is age >= 18 (there is no maximum age because diabetes is a growing problem among the elderly population)
- `weight`: the weight of each patient in pounds (lbs)
- `height`: the height of each patient in inches (in)
- `bmi`: the Body Mass Index (BMI) of each patient. BMI is a simple calculation using a person's height and weight. The formula is BMI = kg/m2 where kg is a person's weight in kilograms and m2 is their height in metres squared. A BMI of 25.0 or more is overweight, while the healthy range is 18.5 to 24.9. The inclusion criteria for this clinical trial is 16 >= BMI >= 38.

Table -> `treatments` and `treatment_cut`:

- `given_name`: the given name of each patient in the Master Patient Index that took part in the clinical trial
- `surname`: the surname of each patient in the Master Patient Index that took part in the clinical trial
- `auralin`: the baseline median daily dose of insulin from the week prior to switching to Auralin (the number before the dash) and the ending median daily dose of insulin at the end of the 24 weeks of treatment measured over the 24th week of treatment (the number after the dash). Both are measured in units (shortform 'u'), which is the international unit of measurement and the standard measurement for insulin.
- `novodra`: same as above, except for patients that continued treatment with Novodra
- `hba1c_start`: the patient's HbA1c level at the beginning of the first week of treatment. HbA1c stands for Hemoglobin A1c. The HbA1c test measures what the average blood sugar has been over the past three months. It is thus a powerful way to get an overall sense of how well diabetes has been controlled. Everyone with diabetes should have this test 2 to 4 times per year. Measured in %.
- `hba1c_end`: the patient's HbA1c level at the end of the last week of treatment
- `hba1c_change`: the change in the patient's HbA1c level from the start of treatment to the end, i.e., hba1c_start - hba1c_end. For Auralin to be deemed effective, it must be

"noninferior" to Novodra, the current standard for insulin. This "noninferiority" is statistically defined as the upper bound of the 95% confidence interval being less than 0.4% for the difference between the mean HbA1c changes for Novodra and Auralin (i.e. Novodra minus Auralin).

Table -> `adverse_reactions`

- `given_name`: the given name of each patient in the Master Patient Index that took part in the clinical trial and had an adverse reaction (includes both patients treated Auralin and Novodra)
- `surname`: the surname of each patient in the Master Patient Index that took part in the clinical trial and had an adverse reaction (includes both patients treated Auralin and Novodra)
- `adverse_reaction`: the adverse reaction reported by the patient

3. Add any additional information

- insulin resistance varies person to person, which is why both starting median daily dose and ending median daily dose are required, i.e. to calculate change in dose.
- it is important to test drugs and medical products in the people they are meant to help. People of different age, race, sex, and ethnic group must be included in clinical trials. This diversity is reflected in the patients table.

4. Types of Assessment

There are 2 types of assessment styles

- `Manual` - Looking through the data manually in google sheets.
- `Automatic` - By using pandas functions such as head(), tail(), info(), describe() or sample()

Steps in Assessment

There are 2 steps involved in Assessment

- Discover
- Document

```
# For Manual assessment we are exporting the Data into excel file

with pd.ExcelWriter('clinical_trials.xlsx') as writer:
  patients.to_excel(writer,sheet_name='patients')
  treatments.to_excel(writer,sheet_name='treatments')
  treatments_cut.to_excel(writer,sheet_name='treatment_cut')
  adverse_reactions.to_excel(writer,sheet_name='adverse_reactions')
```

Documenting Issues

1. Dirty Data

Table - `Patients`

- patient_id = 9 has misspelled name 'Dsvid' instead of David `accuracy`
- state col sometimes contain full name and some times abbrivietation `consistency`
- zip code col has entries with 4 digit `validity`
- data missing for 12 patients in address,city, state,zip_code ,country, contact `completion`
- incorrect data type assigned to sex, zip code, birthdate `validity`
- duplicate entries by the name of John Doe `accuracy`
- one patient has weight = 48 pounds `accuracy`
- one patient has height = 27 inches `accuracy`

Table - `Treatments` & `Treatments_cut`

- given_name and surname col is is all lower case `consistency`
- remove u from Auralin and Novadra cols `validity`
- '-' in novadra and Auralin col treated as nan `validity`
- missing values in hba1c_change col `completion`
- 1 duplicate entry by the name Joseph day `accuracy`
- in hba1c_change 9 instead of 4 `accuracy`

Table - `Adverse_reactions`

- given_name and surname are all in lower case `consistency`

2. Messy Data

Table - `Patients`

- contact col contains both phone and email

Table - `Treatments` & `Treatments_cut`

- Auralin and Novadra col should be split into 2 cols start and end dose
- merge both the tables

Table - `Adverse_reactions`

- This table should not exist independently

Automatic Assessment

- head and tail
- sample
- info
- isnull
- duplicated
- describe

```
patients.head()
```

```
   patient_id assigned_sex given_name  surname                     
address  \
0           1       female        Zoe  Wellish       576 Brown Bear 
Drive   
1           2       female     Pamela     Hill  2370 University Hill 
Road   
2           3         male        Jae   Debord      1493 Poling Farm 
Road   
3           4         male       Liêm     Phan        2335 Webster 
Street   
4           5         male        Tim  Neudorf       1428 Turkey Pen 
Lane   

                 city       state  zip_code        country  \
0  Rancho California  California   92390.0  United States   
1          Armstrong    Illinois   61812.0  United States   
2               York    Nebraska   68467.0  United States   
3         Woodbridge          NJ    7095.0  United States   
4             Dothan          AL   36303.0  United States   

                                      contact  birthdate  weight  
height   bmi  
0        951-719-9170ZoeWellish@superrito.com  7/10/1976   121.7    
66  19.6  
1       PamelaSHill@cuvox.de+1 (217) 569-3204   4/3/1967   118.8    
66  19.2  
2            402-363-6804JaeMDebord@gustr.com  2/19/1980   177.8    
71  24.8  
3  PhanBaLiem@jourrapide.com+1 (732) 636-8246  7/26/1951   220.9    
70  31.7  
4             334-515-7487TimNeudorf@cuvox.de  2/18/1928   192.3    
27  26.1  

treatments.head()

  given_name     surname    auralin    novodra  hba1c_start  hba1c_end  
\
0   veronika    jindrová  41u - 48u          -         7.63       7.20   

1     elliot  richardson          -  40u - 45u         7.56       7.09   

2   yukitaka    takenaka          -  39u - 36u         7.68       7.25   

3       skye  gormanston  33u - 36u          -         7.97       7.62   

4     alissa      montez          -  33u - 29u         7.78       7.46   


   hba1c_change  
0           NaN  
```

```
1           0.97
2            NaN
3           0.35
4           0.32
```

```
treatments_cut.head()
```

```
  given_name    surname    auralin    novodra  hba1c_start  hba1c_end
\
0      jožka  resanovič  22u - 30u          -         7.56       7.22

1  inunnguaq   heilmann  57u - 67u          -         7.85       7.45

2      alwin   svensson  36u - 39u          -         7.78       7.34

3        thể      lương          -  61u - 64u         7.64       7.22

4     amanda    ribeiro  36u - 44u          -         7.85       7.47


   hba1c_change
0          0.34
1           NaN
2           NaN
3          0.92
4          0.38
```

```
adverse_reactions.head()
```

```
  given_name     surname            adverse_reaction
0      berta  napolitani  injection site discomfort
1       lena        baer                hypoglycemia
2     joseph         day                hypoglycemia
3     flavia  fiorentino                       cough
4    manouck     wubbels           throat irritation
```

```
patients.sample(5)
```

```
     patient_id assigned_sex  given_name      surname  \
149         150         male  Wawrzyniec   Nowakowski
247         248         male      Tuukka   Leppäluoto
238         239         male       Aksel  Vestergaard
242         243         male        John      O'Brian
190         191         male      Regolo        Nucci

                      address           city state  zip_code
country  \
149    1525 Crestview Terrace  Mountain Home    TX   78058.0  United
States
247        1886 Bicetown Road       New York    NY   10011.0  United
States
```

```
238  2246 Pheasant Ridge Road   Philadelphia    PA   19139.0  United
States
242                       NaN           NaN   NaN       NaN
NaN
190        3595 Stuart Street       Gibsonia    PA   15044.0  United
States

                                         contact  birthdate  weight
height  \
149  830-640-5848WawrzyniecNowakowski@teleworm.us  9/18/1937   170.5
71
247      917-408-8855TuukkaLeppaluoto@teleworm.us   3/7/1978   211.0
73
238     AkselHVestergaard@armyspy.com215-528-2193   5/2/1988   187.2
78
242                                           NaN  2/25/1957   205.3
74
190       RegoloNucci@einrot.com+1 (724) 449-6928  9/15/1935   213.0
67

      bmi
149  23.8
247  27.8
238  21.6
242  26.4
190  33.4
```

```
patients.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 503 entries, 0 to 502
Data columns (total 14 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   patient_id    503 non-null    int64
 1   assigned_sex  503 non-null    object
 2   given_name    503 non-null    object
 3   surname       503 non-null    object
 4   address       491 non-null    object
 5   city          491 non-null    object
 6   state         491 non-null    object
 7   zip_code      491 non-null    float64
 8   country       491 non-null    object
 9   contact       491 non-null    object
 10  birthdate     503 non-null    object
 11  weight        503 non-null    float64
 12  height        503 non-null    int64
 13  bmi           503 non-null    float64
dtypes: float64(3), int64(2), object(9)
memory usage: 55.1+ KB
```

```
treatments.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 280 entries, 0 to 279
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   given_name    280 non-null    object
 1   surname       280 non-null    object
 2   auralin       280 non-null    object
 3   novodra       280 non-null    object
 4   hba1c_start   280 non-null    float64
 5   hba1c_end     280 non-null    float64
 6   hba1c_change  171 non-null    float64
dtypes: float64(3), object(4)
memory usage: 15.4+ KB
```

```
treatments_cut.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 70 entries, 0 to 69
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   given_name    70 non-null     object
 1   surname       70 non-null     object
 2   auralin       70 non-null     object
 3   novodra       70 non-null     object
 4   hba1c_start   70 non-null     float64
 5   hba1c_end     70 non-null     float64
 6   hba1c_change  42 non-null     float64
dtypes: float64(3), object(4)
memory usage: 4.0+ KB
```

```
adverse_reactions.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 34 entries, 0 to 33
Data columns (total 3 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   given_name        34 non-null     object
 1   surname           34 non-null     object
 2   adverse_reaction  34 non-null     object
dtypes: object(3)
memory usage: 948.0+ bytes
```

```
patients.describe()
```

```
       patient_id      zip_code      weight      height         bmi
count  503.000000    491.000000  503.000000  503.000000  503.000000
```

```
mean   252.000000  49084.118126  173.434990   66.634195   27.483897
std    145.347859  30265.807442   33.916741    4.411297    5.276438
min      1.000000   1002.000000   48.800000   27.000000   17.100000
25%    126.500000  21920.500000  149.300000   63.000000   23.300000
50%    252.000000  48057.000000  175.300000   67.000000   27.200000
75%    377.500000  75679.000000  199.500000   70.000000   31.750000
max    503.000000  99701.000000  255.900000   79.000000   37.700000
```

```
# 48 pound weight is suspicious

patients[patients['weight'] == 48.8]
```

```
     patient_id assigned_sex given_name   surname              address
\
210         211       female    Camilla  Zaitseva  4689 Briarhill Lane


        city state  zip_code        country  \
210  Wooster    OH   44691.0  United States

                                       contact   birthdate  weight
height  \
210  330-202-2145CamillaZaitseva@superrito.com  11/26/1938    48.8
63

      bmi
210  19.1
```

```
# height of 27 inches seems suspicious

patients[patients['height'] == 27]
```

```
   patient_id assigned_sex given_name  surname              address
city  \
4           5         male        Tim  Neudorf  1428 Turkey Pen Lane
Dothan

  state  zip_code        country                          contact
birthdate  \
4    AL   36303.0  United States  334-515-7487TimNeudorf@cuvox.de
2/18/1928

   weight  height   bmi
4   192.3      27  26.1
```

```
treatments.describe()
```

```
       hba1c_start   hba1c_end  hba1c_change
count   280.000000  280.000000    171.000000
mean      7.985929    7.589286      0.546023
std       0.568638    0.569672      0.279555
```

```
min       7.500000    7.010000      0.200000
25%       7.660000    7.270000      0.340000
50%       7.800000    7.420000      0.380000
75%       7.970000    7.570000      0.920000
max       9.950000    9.580000      0.990000
```

```python
treatments.sort_values('hba1c_change',na_position='first')
# Somehow digit 4 is treated as 9 in the data
```

```
    given_name      surname    auralin    novodra  hba1c_start  
hba1c_end  \
0     veronika     jindrová  41u - 48u          -         7.63  
7.20
2     yukitaka     takenaka          -  39u - 36u         7.68  
7.25
8        saber       ménard          -  54u - 54u         8.08  
7.70
9         asia      woźniak  30u - 36u          -         7.76  
7.37
10      joseph          day  29u - 36u          -         7.70  
7.19
..         ...          ...        ...        ...          ...  
...
49     jackson      addison          -  42u - 42u         7.99  
7.51
17        gina         cain          -  36u - 36u         7.88  
7.40
32       laura  ehrlichmann          -  43u - 40u         7.95  
7.46
245         wu         sung          -  47u - 48u         7.61  
7.12
138    giovana        rocha          -  23u - 21u         7.87  
7.38

     hba1c_change  
0             NaN  
2             NaN  
8             NaN  
9             NaN  
10            NaN  
..            ...  
49           0.98  
17           0.98  
32           0.99  
245          0.99  
138          0.99  

[280 rows x 7 columns]
```

Looking about the missing values in `patients` table

```python
patients[patients['address'].isnull()]
```

```
     patient_id assigned_sex given_name      surname address city
state  \
209         210       female     Lalita  Eldarkhanov     NaN  NaN
NaN
219         220         male         Mỹ        Quynh     NaN  NaN
NaN
230         231       female  Elisabeth      Knudsen     NaN  NaN
NaN
234         235       female    Martina     Tománková     NaN  NaN
NaN
242         243         male       John      O'Brian     NaN  NaN
NaN
249         250         male   Benjamin       Mehler     NaN  NaN
NaN
257         258         male        Jin         Kung     NaN  NaN
NaN
264         265       female   Wafiyyah       Asfour     NaN  NaN
NaN
269         270       female     Flavia   Fiorentino     NaN  NaN
NaN
278         279       female   Generosa        Cabán     NaN  NaN
NaN
286         287         male      Lewis         Webb     NaN  NaN
NaN
296         297       female        Chỉ          Lâm     NaN  NaN
NaN

     zip_code country contact   birthdate  weight  height   bmi
209       NaN     NaN     NaN   8/14/1950   143.4      62  26.2
219       NaN     NaN     NaN    4/9/1978   237.8      69  35.1
230       NaN     NaN     NaN   9/23/1976   165.9      63  29.4
234       NaN     NaN     NaN    4/7/1936   199.5      65  33.2
242       NaN     NaN     NaN   2/25/1957   205.3      74  26.4
249       NaN     NaN     NaN  10/30/1951   146.5      69  21.6
257       NaN     NaN     NaN   5/17/1995   231.7      69  34.2
264       NaN     NaN     NaN   11/3/1989   158.6      63  28.1
269       NaN     NaN     NaN   10/9/1937   175.2      61  33.1
278       NaN     NaN     NaN  12/16/1962   124.3      69  18.4
286       NaN     NaN     NaN    4/1/1979   155.3      68  23.6
296       NaN     NaN     NaN   5/14/1990   181.1      63  32.1
```

Checking duplicated values

```python
patients.duplicated().sum()
```

```
0
```

```
# checking duplicated values using given name and surname

patients[patients.duplicated(subset=['given_name', 'surname'])]
```

```
     patient_id assigned_sex given_name surname          address
city  \
229         230         male       John     Doe  123 Main Street  New
York
237         238         male       John     Doe  123 Main Street  New
York
244         245         male       John     Doe  123 Main Street  New
York
251         252         male       John     Doe  123 Main Street  New
York
277         278         male       John     Doe  123 Main Street  New
York

    state  zip_code        country                      contact
birthdate  \
229    NY   12345.0  United States  johndoe@email.com1234567890
1/1/1975
237    NY   12345.0  United States  johndoe@email.com1234567890
1/1/1975
244    NY   12345.0  United States  johndoe@email.com1234567890
1/1/1975
251    NY   12345.0  United States  johndoe@email.com1234567890
1/1/1975
277    NY   12345.0  United States  johndoe@email.com1234567890
1/1/1975

     weight  height   bmi
229   180.0      72  24.4
237   180.0      72  24.4
244   180.0      72  24.4
251   180.0      72  24.4
277   180.0      72  24.4
```

```
treatments.duplicated().sum()
```

```
1
```

```
treatments[treatments.duplicated()]
```

```
    given_name surname    auralin novodra  hba1c_start  hba1c_end  \
136     joseph     day  29u - 36u       -          7.7       7.19

     hba1c_change
136           NaN
```

```
# checking duplicated values using given name and surname

treatments[treatments.duplicated(subset=['given_name', 'surname'])]

    given_name surname     auralin novodra  hba1c_start  hba1c_end  \
136     joseph     day  29u - 36u       -          7.7       7.19

     hba1c_change
136           NaN

treatments_cut.duplicated().sum()

0

treatments_cut[treatments_cut.duplicated(subset=['given_name',
'surname'])]

Empty DataFrame
Columns: [given_name, surname, auralin, novodra, hba1c_start,
hba1c_end, hba1c_change]
Index: []

adverse_reactions.duplicated().sum()

0
```

2c. Data Cleaning

Data Quality Dimensions

- Completeness -> is data missing?
- Validity -> is data invalid -> negative height -> duplicate patient id
- Accuracy -> data is valid but not accurate -> weight -> 1kg
- Consistency -> both valid and accurate but written differently -> New Youk and NY

Order of severity

Completeness <- Validity <- Accuracy <- Consistency

Data Cleaning Order

1. Quality -> Completeness
2. Tidiness
3. Quality -> Validity
4. Quality -> Accuracy
5. Quality -> Consistency

Steps involved in Data cleaning

- Define

- Code
- Test

`Always make sure to create a copy of your pandas dataframe before you start the cleaning process`

```
patients_df = patients.copy()
treatments_df = treatments.copy()
treatments_cut_df = treatments_cut.copy()
adverse_reactions_df = adverse_reactions.copy()
```

According to the order we are handling completness issues first

There are 12 missing values in some columns of patients_df table so can't fill it by correct value so filling with No data is better approach

```
patients_df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 503 entries, 0 to 502
Data columns (total 14 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   patient_id    503 non-null    int64
 1   assigned_sex  503 non-null    object
 2   given_name    503 non-null    object
 3   surname       503 non-null    object
 4   address       491 non-null    object
 5   city          491 non-null    object
 6   state         491 non-null    object
 7   zip_code      491 non-null    float64
 8   country       491 non-null    object
 9   contact       491 non-null    object
 10  birthdate     503 non-null    object
 11  weight        503 non-null    float64
 12  height        503 non-null    int64
 13  bmi           503 non-null    float64
dtypes: float64(3), int64(2), object(9)
memory usage: 55.1+ KB

# code
patients_df.fillna({'zip_code':00000}, inplace=True)
patients_df.fillna('No data', inplace=True)

# test
patients_df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 503 entries, 0 to 502
Data columns (total 14 columns):
```

```
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   patient_id     503 non-null    int64
 1   assigned_sex   503 non-null    object
 2   given_name     503 non-null    object
 3   surname        503 non-null    object
 4   address        503 non-null    object
 5   city           503 non-null    object
 6   state          503 non-null    object
 7   zip_code       503 non-null    float64
 8   country        503 non-null    object
 9   contact        503 non-null    object
 10  birthdate      503 non-null    object
 11  weight         503 non-null    float64
 12  height         503 non-null    int64
 13  bmi            503 non-null    float64
dtypes: float64(3), int64(2), object(9)
memory usage: 55.1+ KB
```

Handling the missing and incorrect values in hba1c_change column in treatments_df and treatments_cut_df table

```
treatments_df.head()

  given_name     surname    auralin    novodra  hba1c_start  hba1c_end
\
0   veronika    jindrová  41u - 48u          -         7.63       7.20

1     elliot  richardson          -  40u - 45u         7.56       7.09

2   yukitaka    takenaka          -  39u - 36u         7.68       7.25

3       skye  gormanston  33u - 36u          -         7.97       7.62

4     alissa      montez          -  33u - 29u         7.78       7.46


   hba1c_change
0           NaN
1          0.97
2           NaN
3          0.35
4          0.32

# code
treatments_df['hba1c_change'] = treatments_df['hba1c_start'] -
treatments_df['hba1c_end']
treatments_cut_df['hba1c_change'] = treatments_cut_df['hba1c_start'] -
treatments_cut_df['hba1c_end']
```

```python
# test
treatments_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 280 entries, 0 to 279
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   given_name    280 non-null    object
 1   surname       280 non-null    object
 2   auralin       280 non-null    object
 3   novodra       280 non-null    object
 4   hba1c_start   280 non-null    float64
 5   hba1c_end     280 non-null    float64
 6   hba1c_change  280 non-null    float64
dtypes: float64(3), object(4)
memory usage: 15.4+ KB
```

```python
treatments_cut_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 70 entries, 0 to 69
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   given_name    70 non-null     object
 1   surname       70 non-null     object
 2   auralin       70 non-null     object
 3   novodra       70 non-null     object
 4   hba1c_start   70 non-null     float64
 5   hba1c_end     70 non-null     float64
 6   hba1c_change  70 non-null     float64
dtypes: float64(3), object(4)
memory usage: 4.0+ KB
```

Handling the contact column in patients_df table

```python
patients.head()
```

```
   patient_id assigned_sex given_name  surname
address  \
0           1       female        Zoe  Wellish       576 Brown Bear
Drive
1           2       female     Pamela     Hill  2370 University Hill
Road
2           3         male        Jae   Debord      1493 Poling Farm
Road
3           4         male       Liêm     Phan        2335 Webster
Street
4           5         male        Tim  Neudorf       1428 Turkey Pen
```

```
Lane

                 city       state  zip_code        country  \
0  Rancho California  California   92390.0  United States
1          Armstrong    Illinois   61812.0  United States
2               York    Nebraska   68467.0  United States
3         Woodbridge          NJ    7095.0  United States
4             Dothan          AL   36303.0  United States

                                      contact  birthdate  weight
height   bmi
0        951-719-9170ZoeWellish@superrito.com  7/10/1976   121.7
66  19.6
1       PamelaSHill@cuvox.de+1 (217) 569-3204   4/3/1967   118.8
66  19.2
2            402-363-6804JaeMDebord@gustr.com  2/19/1980   177.8
71  24.8
3  PhanBaLiem@jourrapide.com+1 (732) 636-8246  7/26/1951   220.9
70  31.7
4             334-515-7487TimNeudorf@cuvox.de  2/18/1928   192.3
27  26.1
```

After Completion issues now we are handling untidy data

```
import re

def find_contact_details(text: str) -> tuple:
    if pd.isna(text):
        return np.nan, np.nan

    # phone number pattern
    phone_number_pattern = re.compile(r"(\+[\d]{1,3}\s)?(\(?[\d]
{3}\)?\s?-?[\d]{3}\s?-?[\d]{4})")
    # email pattern
    email_pattern = re.compile(r"[\w\.-]+@[\w\.-]+")

    # Extract phone number
    phone_number_matches = re.findall(phone_number_pattern, text)
    if len(phone_number_matches) > 0:
        # Flatten the tuple and join parts to form the complete phone
number
        phone_number = ''.join(phone_number_matches[0]).strip()
        # Remove the phone number part from the text
        remaining_text = re.sub(phone_number_pattern, "",
text).strip()
    else:
        phone_number = np.nan
        remaining_text = text

    # Extract email
```

```
    email_matches = re.findall(email_pattern, remaining_text)
    if len(email_matches) > 0:
        email = email_matches[0].strip()
    else:
        email = np.nan

    return phone_number, email

patients_df['phone'] =
patients_df['contact'].apply(find_contact_details).apply(lambda
x:x[0])
patients_df['email'] =
patients_df['contact'].apply(find_contact_details).apply(lambda
x:x[1])

# test
patients_df.head()
```

```
   patient_id assigned_sex given_name  surname
address  \
0           1       female        Zoe  Wellish       576 Brown Bear
Drive
1           2       female     Pamela     Hill  2370 University Hill
Road
2           3         male        Jae   Debord      1493 Poling Farm
Road
3           4         male       Liêm     Phan        2335 Webster
Street
4           5         male        Tim  Neudorf       1428 Turkey Pen
Lane

                city       state  zip_code        country  \
0  Rancho California  California   92390.0  United States
1          Armstrong    Illinois   61812.0  United States
2               York    Nebraska   68467.0  United States
3         Woodbridge          NJ    7095.0  United States
4             Dothan          AL   36303.0  United States

                                    contact  birthdate  weight
height  \
0        951-719-9170ZoeWellish@superrito.com  7/10/1976   121.7
66
1       PamelaSHill@cuvox.de+1 (217) 569-3204   4/3/1967   118.8
66
2            402-363-6804JaeMDebord@gustr.com  2/19/1980   177.8
71
3  PhanBaLiem@jourrapide.com+1 (732) 636-8246  7/26/1951   220.9
70
4             334-515-7487TimNeudorf@cuvox.de  2/18/1928   192.3
27
```

```
    bmi              phone                          email
0  19.6       951-719-9170   ZoeWellish@superrito.com
1  19.2  +1 (217) 569-3204       PamelaSHill@cuvox.de
2  24.8       402-363-6804       JaeMDebord@gustr.com
3  31.7  +1 (732) 636-8246  PhanBaLiem@jourrapide.com
4  26.1       334-515-7487         TimNeudorf@cuvox.de
```

Concatinating treatments_df and treatments_cut_df because both table contains same data

```
# code
treatments_df = pd.concat([treatments_df, treatments_cut_df])

# test
treatments_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 350 entries, 0 to 69
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   given_name    350 non-null    object
 1   surname       350 non-null    object
 2   auralin       350 non-null    object
 3   novodra       350 non-null    object
 4   hba1c_start   350 non-null    float64
 5   hba1c_end     350 non-null    float64
 6   hba1c_change  350 non-null    float64
dtypes: float64(3), object(4)
memory usage: 21.9+ KB
```

Handling the columns Novodra and Auralin also creating different columns for dosage

```
treatments_df = treatments_df.melt(id_vars=['given_name', 'surname',
'hba1c_start', 'hba1c_end', 'hba1c_change'], var_name='type',
value_name='dosage_range')
```

since for every patient melt created two rows so we will drop null values that are represented using '-'

```
treatments_df = treatments_df[treatments_df['dosage_range'] != '-']

treatments_df['dosage_start'] =
treatments_df['dosage_range'].str.split('-').str.get(0)
treatments_df['dosage_end'] =
treatments_df['dosage_range'].str.split('-').str.get(1)

# droppin unneccessary column
treatments_df.drop(columns='dosage_range', inplace=True)
```

```
# removing u from dosage_start and dosage_end since it is not required
treatments_df['dosage_start'] =
treatments_df['dosage_start'].str.replace('u', '')
treatments_df['dosage_end'] =
treatments_df['dosage_end'].str.replace('u', '')

# changing dtype to int
treatments_df['dosage_start'] =
treatments_df['dosage_start'].astype('int')
treatments_df['dosage_end'] =
treatments_df['dosage_end'].astype('int')

# test
treatments_df
```

```
      given_name     surname  hba1c_start  hba1c_end  hba1c_change
type  \
0       veronika    jindrová         7.63       7.20          0.43
auralin
3           skye  gormanston         7.97       7.62          0.35
auralin
6         sophia      haugen         7.65       7.27          0.38
auralin
7          eddie      archer         7.89       7.55          0.34
auralin
9           asia     woźniak         7.76       7.37          0.39
auralin
..           ...         ...          ...        ...           ...
...
688  christopher    woodward         7.51       7.06          0.45
novodra
690        maret    sultygov         7.67       7.30          0.37
novodra
694        lixue       hsueh         9.21       8.80          0.41
novodra
696        jakob    jakobsen         7.96       7.51          0.45
novodra
698        berta  napolitani         7.68       7.21          0.47
novodra

     dosage_start  dosage_end
0              41          48
3              33          36
6              37          42
7              31          38
9              30          36
..            ...         ...
688            55          51
690            26          23
694            22          23
```

```
696             28          26
698             42          44

[350 rows x 8 columns]
```

Merging the adverse_reactions_df table with treatments table

```
# code
treatments_df = treatments_df.merge(adverse_reactions_df, how='left',
on=['given_name', 'surname'])

# test
treatments_df

      given_name     surname  hba1c_start  hba1c_end  hba1c_change
type  \
0       veronika    jindrová         7.63       7.20          0.43
auralin
1           skye  gormanston         7.97       7.62          0.35
auralin
2         sophia      haugen         7.65       7.27          0.38
auralin
3         eddie       archer         7.89       7.55          0.34
auralin
4           asia     woźniak         7.76       7.37          0.39
auralin
..           ...         ...          ...        ...           ...
...
345  christopher    woodward         7.51       7.06          0.45
novodra
346        maret    sultygov         7.67       7.30          0.37
novodra
347        lixue       hsueh         9.21       8.80          0.41
novodra
348        jakob    jakobsen         7.96       7.51          0.45
novodra
349        berta  napolitani         7.68       7.21          0.47
novodra

     dosage_start  dosage_end           adverse_reaction
0              41          48                        NaN
1              33          36                        NaN
2              37          42                        NaN
3              31          38                        NaN
4              30          36                        NaN
..            ...         ...                        ...
345            55          51                     nausea
346            26          23                        NaN
347            22          23  injection site discomfort
348            28          26                hypoglycemia
```

```
349            42          44  injection site discomfort

[350 rows x 9 columns]
```

After handling messy data now handling data with validity issues

zip code col in patients_df table has values in 4 digits, on researching I found that this is due to leading 0. so if we put leading zero it will be handled

```
patients_df['zip_code'].value_counts().head(15)

zip_code
0.0         12
12345.0      6
30303.0      4
10004.0      4
35203.0      3
15205.0      3
1730.0       3
60148.0      3
70112.0      3
11530.0      3
11590.0      3
10011.0      3
90017.0      3
98109.0      3
95814.0      2
Name: count, dtype: int64

patients_df['zip_code'] = patients_df['zip_code'].apply(lambda x: 
str(x).zfill(5))
```

Correcting Datatype of sex, zip_code and bithdate columns

```
patients_df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 503 entries, 0 to 502
Data columns (total 16 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   patient_id    503 non-null    int64 
 1   assigned_sex  503 non-null    object
 2   given_name    503 non-null    object
 3   surname       503 non-null    object
 4   address       503 non-null    object
 5   city          503 non-null    object
 6   state         503 non-null    object
 7   zip_code      503 non-null    object
```

```
 8   country        503 non-null    object
 9   contact        503 non-null    object
 10  birthdate      503 non-null    object
 11  weight         503 non-null    float64
 12  height         503 non-null    int64
 13  bmi            503 non-null    float64
 14  phone          491 non-null    object
 15  email          491 non-null    object
dtypes: float64(2), int64(2), object(12)
memory usage: 63.0+ KB
```

```
patients_df['zip_code'] = patients_df['zip_code'].astype('int')

patients_df['assigned_sex'] =
patients_df['assigned_sex'].astype('category')

patients_df['birthdate']
```

```
0       7/10/1976
1        4/3/1967
2       2/19/1980
3       7/26/1951
4       2/18/1928
          ...
498     4/10/1959
499     3/26/1948
500     1/13/1971
501     2/13/1952
502      5/3/1954
Name: birthdate, Length: 503, dtype: object
```

```
patients_df['birthdate'] = pd.to_datetime(patients_df['birthdate'],
format='%m/%d/%Y')
```

After handling validity issues now we are heading to accuracy issues

Correcting typo at patient_id 9

```
patients_df[patients_df['patient_id'] == 9]['given_name'] = 'David'
```

```
C:\Users\DILKHUSH\AppData\Local\Temp\ipykernel_10332\1526163683.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  patients_df[patients_df['patient_id'] == 9]['given_name'] = 'David'
```

removing the duplicate entries by the name John Doe

```
patients_df[patients_df.duplicated(subset=['given_name', 'surname'])]
```

```
     patient_id assigned_sex given_name surname          address
city  \
229         230          male       John     Doe  123 Main Street  New
York
237         238          male       John     Doe  123 Main Street  New
York
244         245          male       John     Doe  123 Main Street  New
York
251         252          male       John     Doe  123 Main Street  New
York
277         278          male       John     Doe  123 Main Street  New
York

    state  zip_code        country                      contact
birthdate  \
229    NY     12345  United States  johndoe@email.com1234567890 1975-
01-01
237    NY     12345  United States  johndoe@email.com1234567890 1975-
01-01
244    NY     12345  United States  johndoe@email.com1234567890 1975-
01-01
251    NY     12345  United States  johndoe@email.com1234567890 1975-
01-01
277    NY     12345  United States  johndoe@email.com1234567890 1975-
01-01

     weight  height   bmi       phone              email
229   180.0      72  24.4  1234567890  johndoe@email.com
237   180.0      72  24.4  1234567890  johndoe@email.com
244   180.0      72  24.4  1234567890  johndoe@email.com
251   180.0      72  24.4  1234567890  johndoe@email.com
277   180.0      72  24.4  1234567890  johndoe@email.com
```

```
patients_df.drop_duplicates(subset=['given_name', 'surname'],
inplace=True)
```

Removing entries with outliers

```
patients_df['weight'].describe()
```

```
count    498.000000
mean     173.369076
std       34.080497
min       48.800000
25%      148.825000
50%      174.450000
```

```
75%        199.725000
max        255.900000
Name: weight, dtype: float64
```

```python
patients_df['weight'].sort_values()
```

```
210     48.8
459    102.1
335    102.7
74     103.2
317    106.0
       ...
144    244.9
61     244.9
283    245.5
118    254.5
485    255.9
Name: weight, Length: 498, dtype: float64
```

```python
index_to_drop = patients_df[patients_df['weight'] == 48.8].index

patients_df = patients_df.drop(index_to_drop)

patients_df['weight'].sort_values()
```

```
459    102.1
335    102.7
74     103.2
317    106.0
171    106.5
       ...
61     244.9
144    244.9
283    245.5
118    254.5
485    255.9
Name: weight, Length: 497, dtype: float64
```

```python
patients_df['height'].describe()
```

```
count    497.000000
mean      66.587525
std        4.401806
min       27.000000
25%       63.000000
50%       67.000000
75%       69.000000
max       79.000000
Name: height, dtype: float64
```

```python
patients_df['height'].sort_values()
```

```
4      27
181    59
232    59
335    59
454    59
       ..
83     76
121    76
487    77
238    78
418    79
Name: height, Length: 497, dtype: int64
```

```
index_to_drop = patients_df[patients_df['height'] == 27].index

patients_df = patients_df.drop(index_to_drop)

patients_df['height'].sort_values()
```

```
171    59
335    59
423    59
454    59
181    59
       ..
83     76
121    76
487    77
238    78
418    79
Name: height, Length: 496, dtype: int64
```

removing duplicate entry by the name Joseph day in treatments_df table

```
treatments_df[treatments_df.duplicated(subset=['given_name',
'surname'])]

   given_name surname  hba1c_start  hba1c_end  hba1c_change
type  \
62     joseph     day          7.7       7.19          0.51  auralin


    dosage_start  dosage_end adverse_reaction
62            29          36     hypoglycemia

treatments_df.drop_duplicates(subset=['given_name', 'surname'],
inplace=True)
```

Lastly consistency issues need to be resolved

state column in patients_df table contains inconsistent values, so we are converting all the values in abbr form

```
patients_df['state'].value_counts()
```

```
state
California    36
TX            32
New York      25
CA            24
MA            22
PA            18
NY            17
GA            15
Illinois      14
Florida       13
MI            13
OH            13
OK            13
LA            13
NJ            12
No data       12
VA            11
MS            10
WI            10
IL            10
IN             9
MN             9
FL             9
TN             9
AL             8
NC             8
KY             8
WA             8
MO             7
ID             6
NV             6
KS             6
SC             5
IA             5
CT             5
ME             4
ND             4
Nebraska       4
RI             4
AR             4
CO             4
AZ             4
```

```
MD              3
DE              3
WV              3
OR              3
SD              3
MT              2
VT              2
DC              2
NE              2
AK              1
WY              1
NH              1
NM              1
Name: count, dtype: int64
```

```
# Mapping dictionary for states
state_mapping = {
    'California': 'CA', 'New York': 'NY', 'Illinois': 'IL', 'Florida':
'FL',
    'Texas': 'TX', 'Georgia': 'GA', 'Michigan': 'MI', 'Ohio': 'OH',
    'Oklahoma': 'OK', 'Louisiana': 'LA', 'New Jersey': 'NJ',
'Virginia': 'VA',
    'Massachusetts': 'MA', 'Pennsylvania': 'PA', 'Mississippi': 'MS',
    'Wisconsin': 'WI', 'Indiana': 'IN', 'Minnesota': 'MN',
'Tennessee': 'TN',
    'Alabama': 'AL', 'North Carolina': 'NC', 'Kentucky': 'KY',
    'Washington': 'WA', 'Missouri': 'MO', 'Idaho': 'ID', 'Nevada':
'NV',
    'Kansas': 'KS', 'South Carolina': 'SC', 'Iowa': 'IA',
'Connecticut': 'CT',
    'Maine': 'ME', 'North Dakota': 'ND', 'Nebraska': 'NE', 'Rhode
Island': 'RI',
    'Arkansas': 'AR', 'Colorado': 'CO', 'Arizona': 'AZ', 'Maryland':
'MD',
    'Delaware': 'DE', 'West Virginia': 'WV', 'Oregon': 'OR', 'South
Dakota': 'SD',
    'Montana': 'MT', 'Vermont': 'VT', 'Washington D.C.': 'DC',
'Alaska': 'AK',
    'Wyoming': 'WY', 'New Hampshire': 'NH', 'New Mexico': 'NM'
}
```

```
len(state_mapping)
```

```
49
```

```
len(patients_df['state'].replace(state_mapping).value_counts())
# 50 categories because we filled NA values with 'No data'
```

```
50
```

given_name and surname in treatments_df is in lower case but in patiets_df table they are in upper case

```
patients_df[['given_name', 'surname']]

    given_name      surname
0          Zoe      Wellish
1       Pamela         Hill
2          Jae       Debord
3         Liêm         Phan
5       Rafael        Costa
..         ...          ...
498    Mustafa    Lindström
499      Ruman      Bisliev
500      Jinke    de Keizer
501    Chidalu  Onyekaozulu
502        Pat      Gersten

[496 rows x 2 columns]

treatments_df[['given_name', 'surname']]

      given_name     surname
0       veronika     jindrová
1           skye  gormanston
2         sophia      haugen
3          eddie      archer
4           asia     woźniak
..           ...         ...
345  christopher    woodward
346        maret    sultygov
347        lixue       hsueh
348        jakob    jakobsen
349        berta  napolitani

[349 rows x 2 columns]

treatments_df['given_name'] = treatments_df['given_name'].str.title()
treatments_df['surname'] = treatments_df['surname'].str.title()

treatments_df[['given_name', 'surname']]

      given_name     surname
0       Veronika     Jindrová
1           Skye  Gormanston
2         Sophia      Haugen
3          Eddie      Archer
4           Asia     Woźniak
..           ...         ...
345  Christopher    Woodward
346        Maret    Sultygov
```

```
347        Lixue       Hsueh
348        Jakob    Jakobsen
349        Berta  Napolitani

[349 rows x 2 columns]
```

One cycle completed, Let's check again

Issues with the dataset

1. Dirty Data Table - `Patients`

- phone and email columns has missing values `Completness`

Table - `Treatments_df`

- adverse_reaction column has missing values `Completness`

1. Messy Data

- contact column should not be exist.

phone and email columns has missing values because the contact column also has null values.

```
patients_df.info()

<class 'pandas.core.frame.DataFrame'>
Index: 496 entries, 0 to 502
Data columns (total 16 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   patient_id    496 non-null    int64
 1   assigned_sex  496 non-null    category
 2   given_name    496 non-null    object
 3   surname       496 non-null    object
 4   address       496 non-null    object
 5   city          496 non-null    object
 6   state         496 non-null    object
 7   zip_code      496 non-null    int32
 8   country       496 non-null    object
 9   contact       496 non-null    object
 10  birthdate     496 non-null    datetime64[ns]
 11  weight        496 non-null    float64
 12  height        496 non-null    int64
 13  bmi           496 non-null    float64
 14  phone         484 non-null    object
 15  email         484 non-null    object
dtypes: category(1), datetime64[ns](1), float64(2), int32(1),
int64(2), object(9)
memory usage: 60.7+ KB

patients_df[patients_df['phone'].isnull()]
```

```
     patient_id assigned_sex given_name      surname  address     city  \
209         210       female     Lalita  Eldarkhanov  No data  No data

219         220         male         Mỹ        Quynh  No data  No data

230         231       female  Elisabeth      Knudsen  No data  No data

234         235       female    Martina    Tománková  No data  No data

242         243         male       John      O'Brian  No data  No data

249         250         male   Benjamin       Mehler  No data  No data

257         258         male        Jin         Kung  No data  No data

264         265       female   Wafiyyah       Asfour  No data  No data

269         270       female     Flavia   Fiorentino  No data  No data

278         279       female   Generosa        Cabán  No data  No data

286         287         male      Lewis         Webb  No data  No data

296         297       female        Chỉ          Lâm  No data  No data


       state  zip_code  country  contact  birthdate  weight  height  bmi  \
209  No data         0  No data  No data 1950-08-14   143.4      62  26.2
219  No data         0  No data  No data 1978-04-09   237.8      69  35.1
230  No data         0  No data  No data 1976-09-23   165.9      63  29.4
234  No data         0  No data  No data 1936-04-07   199.5      65  33.2
242  No data         0  No data  No data 1957-02-25   205.3      74  26.4
249  No data         0  No data  No data 1951-10-30   146.5      69  21.6
257  No data         0  No data  No data 1995-05-17   231.7      69  34.2
264  No data         0  No data  No data 1989-11-03   158.6      63  28.1
269  No data         0  No data  No data 1937-10-09   175.2      61  33.1
278  No data         0  No data  No data 1962-12-16   124.3      69  18.4
286  No data         0  No data  No data 1979-04-01   155.3      68  23.6
```

```
296  No data          0  No data  No data 1990-05-14   181.1      63
32.1

    phone email
209   NaN   NaN
219   NaN   NaN
230   NaN   NaN
234   NaN   NaN
242   NaN   NaN
249   NaN   NaN
257   NaN   NaN
264   NaN   NaN
269   NaN   NaN
278   NaN   NaN
286   NaN   NaN
296   NaN   NaN
```

```
patients_df.fillna({'phone':'0', 'email':'No data'}, inplace=True)
```

Dropping irrelevant column contact

```
patients_df.drop(columns='contact', inplace=True)
```

Handlin missing values of adverse_reaction column

```
treatments_df
```

```
      given_name     surname  hba1c_start  hba1c_end  hba1c_change
type  \
0       Veronika    Jindrová         7.63       7.20          0.43
auralin
1           Skye  Gormanston         7.97       7.62          0.35
auralin
2         Sophia      Haugen         7.65       7.27          0.38
auralin
3          Eddie      Archer         7.89       7.55          0.34
auralin
4           Asia     Woźniak         7.76       7.37          0.39
auralin
..           ...         ...          ...        ...           ...
...
345  Christopher    Woodward         7.51       7.06          0.45
novodra
346        Maret    Sultygov         7.67       7.30          0.37
novodra
347        Lixue       Hsueh         9.21       8.80          0.41
novodra
348        Jakob    Jakobsen         7.96       7.51          0.45
novodra
```

```
349        Berta  Napolitani         7.68       7.21          0.47
novodra

     dosage_start  dosage_end           adverse_reaction
0              41          48                        NaN
1              33          36                        NaN
2              37          42                        NaN
3              31          38                        NaN
4              30          36                        NaN
..            ...         ...                        ...
345            55          51                     nausea
346            26          23                        NaN
347            22          23  injection site discomfort
348            28          26               hypoglycemia
349            42          44  injection site discomfort

[349 rows x 9 columns]
```

```
treatments_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 349 entries, 0 to 349
Data columns (total 9 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   given_name        349 non-null    object 
 1   surname           349 non-null    object 
 2   hba1c_start       349 non-null    float64
 3   hba1c_end         349 non-null    float64
 4   hba1c_change      349 non-null    float64
 5   type              349 non-null    object 
 6   dosage_start      349 non-null    int32  
 7   dosage_end        349 non-null    int32  
 8   adverse_reaction  34 non-null     object 
dtypes: float64(3), int32(2), object(4)
memory usage: 24.5+ KB
```

```
treatments_df.fillna({'adverse_reaction':'No reaction'}, inplace=True)
```

```
treatments_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 349 entries, 0 to 349
Data columns (total 9 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   given_name        349 non-null    object 
 1   surname           349 non-null    object 
 2   hba1c_start       349 non-null    float64
 3   hba1c_end         349 non-null    float64
 4   hba1c_change      349 non-null    float64
```

```
 5   type               349 non-null    object
 6   dosage_start       349 non-null    int32
 7   dosage_end         349 non-null    int32
 8   adverse_reaction   349 non-null    object
dtypes: float64(3), int32(2), object(4)
memory usage: 24.5+ KB
```

```
treatments_df['adverse_reaction'].value_counts()
```

```
adverse_reaction
No reaction                    315
hypoglycemia                    19
injection site discomfort        6
headache                         3
throat irritation                2
nausea                           2
cough                            2
Name: count, dtype: int64
```

so almost data assesin and cleaning is completed but it can be further explored because it is iterative process.

3. Exploratory Data Analysis

Why do EDA

- Model building
- Analysis and reporting
- Validate assumptions
- Handling missing values
- feature engineering
- detecting outliers

Steps

1. Categorize Columns in three Types

- **Numerical**
- **Categorical**
- **Mixed**

2. Univariate Analysis

Univariate analysis focuses on analyzing each feature in the dataset independently.

- **Distribution analysis**: The distribution of each feature is examined to identify its shape, central tendency, and dispersion.

- **Identifying potential issues**: Univariate analysis helps in identifying potential problems with the data such as outliers, skewness, and missing values

The shape of a data distribution refers to its overall pattern or form as it is represented on a graph. Some common shapes of data distributions include:

- **Normal Distribution**: A symmetrical and bell-shaped distribution where the mean, median, and mode are equal and the majority of the data falls in the middle of the distribution with gradually decreasing frequencies towards the tails.
- **Skewed Distribution**: A distribution that is not symmetrical, with one tail being longer than the other. It can be either positively skewed (right-skewed) or negatively skewed (left-skewed).
- **Bimodal Distribution**: A distribution with two peaks or modes.
- **Uniform Distribution**: A distribution where all values have an equal chance of occurring.

The shape of the data distribution is important in identifying the presence of outliers, skewness, and the type of statistical tests and models that can be used for further analysis.

Dispersion is a statistical term used to describe the spread or variability of a set of data. It measures how far the values in a data set are spread out from the central tendency (mean, median, or mode) of the data.

There are several measures of dispersion, including:

- **Range**: The difference between the largest and smallest values in a data set.
- **Variance**: The average of the squared deviations of each value from the mean of the data set.
- **Standard Deviation**: The square root of the variance. It provides a measure of the spread of the data that is in the same units as the original data.
- **Interquartile range (IQR)**: The range between the first quartile (25th percentile) and the third quartile (75th percentile) of the data.

Dispersion helps to describe the spread of the data, which can help to identify the presence of outliers and skewness in the data.

Steps of doing Univariate Analysis on Numerical columns

- **Descriptive Statistics**: Compute basic summary statistics for the column, such as mean, median, mode, standard deviation, range, and quartiles. These statistics give a general understanding of the distribution of the data and can help identify skewness or outliers.
- **Visualizations**: Create visualizations to explore the distribution of the data. Some common visualizations for numerical data include histograms, box plots, and

density plots. These visualizations provide a visual representation of the distribution of the data and can help identify skewness an outliers.

- **Identifying Outliers**: Identify and examine any outliers in the data. Outliers can be identified using visualizations. It is important to determine whether the outliers are due to measurement errors, data entry errors, or legitimate differences in the data, and to decide whether to include or exclude them from the analysis.

- **Skewness**: Check for skewness in the data and consider transforming the data or using robust statistical methods that are less sensitive to skewness, if necessary.

- **Conclusion**: Summarize the findings of the EDA and make decisions about how to proceed with further analysis.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv('Datasets/train.csv')

df

     PassengerId  Survived  Pclass  \
0              1         0       3
1              2         1       1
2              3         1       3
3              4         1       1
4              5         0       3
..           ...       ...     ...
886          887         0       2
887          888         1       1
888          889         0       3
889          890         1       1
890          891         0       3

                                                  Name     Sex   Age
SibSp  \
0                              Braund, Mr. Owen Harris    male  22.0
1
1    Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0
1
2                               Heikkinen, Miss. Laina  female  26.0
0
3         Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0
1
4                             Allen, Mr. William Henry    male  35.0
0
..                                                 ...     ...   ...
...
```

```
886                            Montvila, Rev. Juozas    male  27.0
0
887                     Graham, Miss. Margaret Edith  female  19.0
0
888         Johnston, Miss. Catherine Helen "Carrie"  female   NaN
1
889                            Behr, Mr. Karl Howell    male  26.0
0
890                              Dooley, Mr. Patrick    male  32.0
0

     Parch            Ticket     Fare Cabin Embarked
0        0         A/5 21171   7.2500   NaN        S
1        0          PC 17599  71.2833   C85        C
2        0  STON/O2. 3101282   7.9250   NaN        S
3        0            113803  53.1000  C123        S
4        0            373450   8.0500   NaN        S
..     ...               ...      ...   ...      ...
886      0            211536  13.0000   NaN        S
887      0            112053  30.0000   B42        S
888      2        W./C. 6607  23.4500   NaN        S
889      0            111369  30.0000  C148        C
890      0            370376   7.7500   NaN        Q

[891 rows x 12 columns]
```

Column Types

- **Numerical** - Age,Fare,PassengerId
- **Categorical** - Survived, Pclass, Sex, SibSp, Parch,Embarked
- **Mixed** - Name, Ticket, Cabin

Univariate Analysis on Numerical columns

Age

conclusions

- Age is almost normally distributed.
- Nearly 20% values are missing.
- There are some outliers but they are real values.

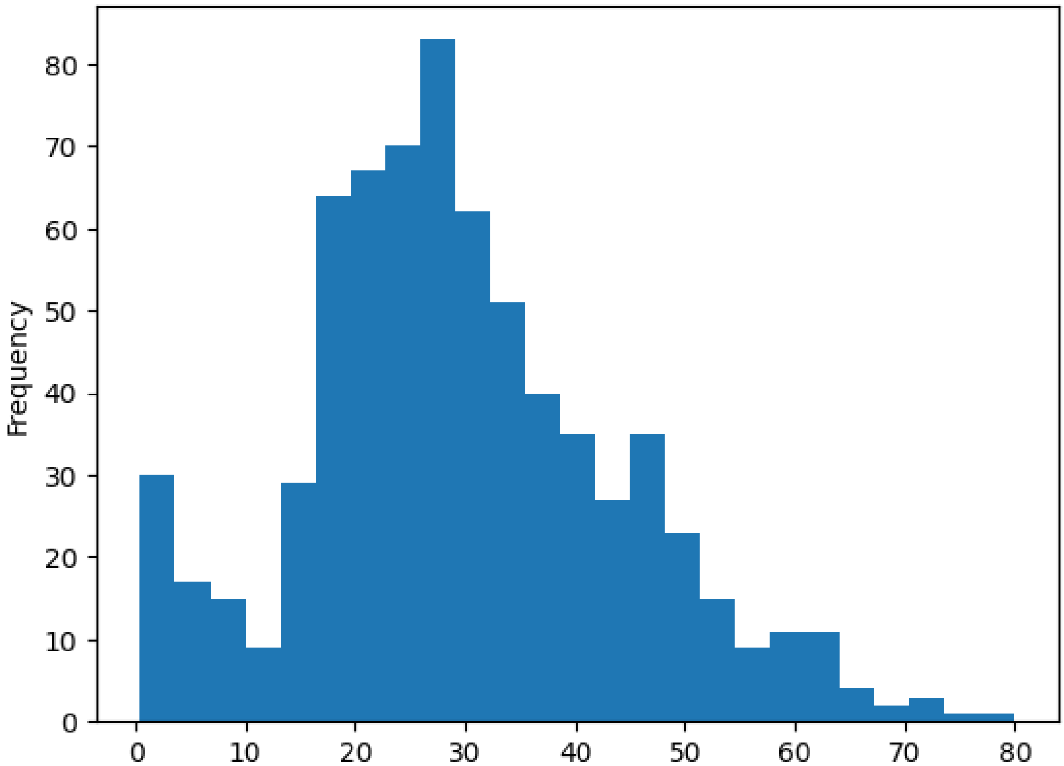
```
df['Age'].describe()

count    714.000000
mean      29.699118
std       14.526497
min        0.420000
25%       20.125000
50%       28.000000
```

```
75%        38.000000
max        80.000000
Name: Age, dtype: float64
```

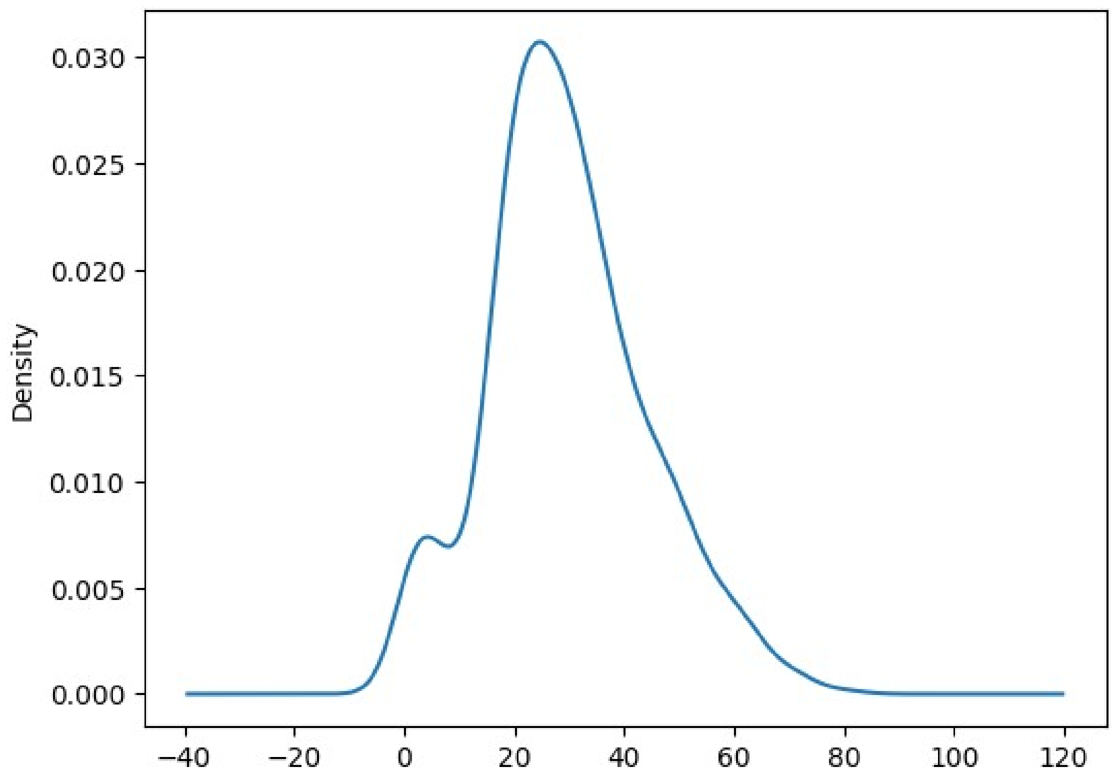
```
df['Age'].plot(kind='hist', bins=25)
```

```
<Axes: ylabel='Frequency'>
```



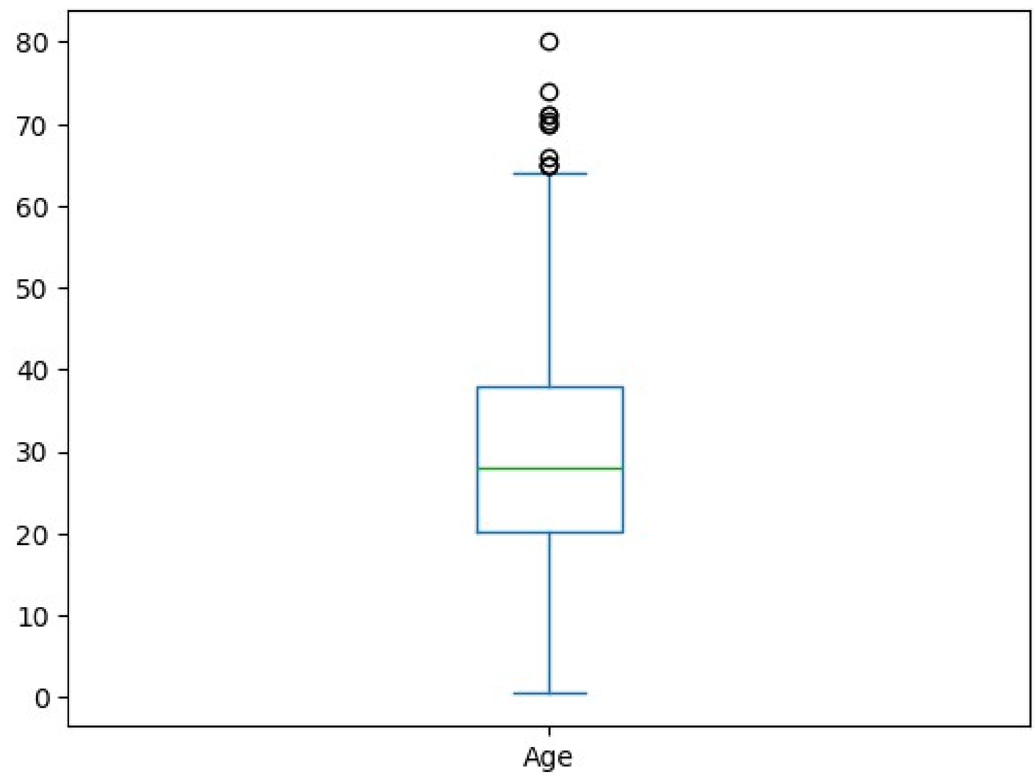

```
df['Age'].plot(kind='kde')
```

```
<Axes: ylabel='Density'>
```




```
df['Age'].plot(kind='box')
```

<Axes: >




```
df['Age'].skew()

0.38910778230082704

df[df['Age']>65]

     PassengerId  Survived  Pclass
Name  \
33            34         0       2                Wheadon, Mr. Edward
H
96            97         0       1            Goldschmidt, Mr. George
B
116          117         0       3                      Connors, Mr.
Patrick
493          494         0       1                  Artagaveytia, Mr.
Ramon
630          631         1       1  Barkworth, Mr. Algernon Henry
Wilson
672          673         0       2              Mitchell, Mr. Henry
Michael
745          746         0       1             Crosby, Capt. Edward
Gifford
851          852         0       3                     Svensson, Mr.
Johan

      Sex    Age  SibSp  Parch      Ticket     Fare Cabin Embarked
```

```
33   male  66.0      0      0  C.A. 24579  10.5000   NaN        S
96   male  71.0      0      0    PC 17754  34.6542    A5        C
116  male  70.5      0      0      370369   7.7500   NaN        Q
493  male  71.0      0      0    PC 17609  49.5042   NaN        C
630  male  80.0      0      0       27042  30.0000   A23        S
672  male  70.0      0      0  C.A. 24580  10.5000   NaN        S
745  male  70.0      1      1   WE/P 5735  71.0000   B22        S
851  male  74.0      0      0      347060   7.7750   NaN        S
```

```
df['Age'].isnull().sum()/len(df['Age'])
```

```
0.19865319865319866
```

Fare

conclusions

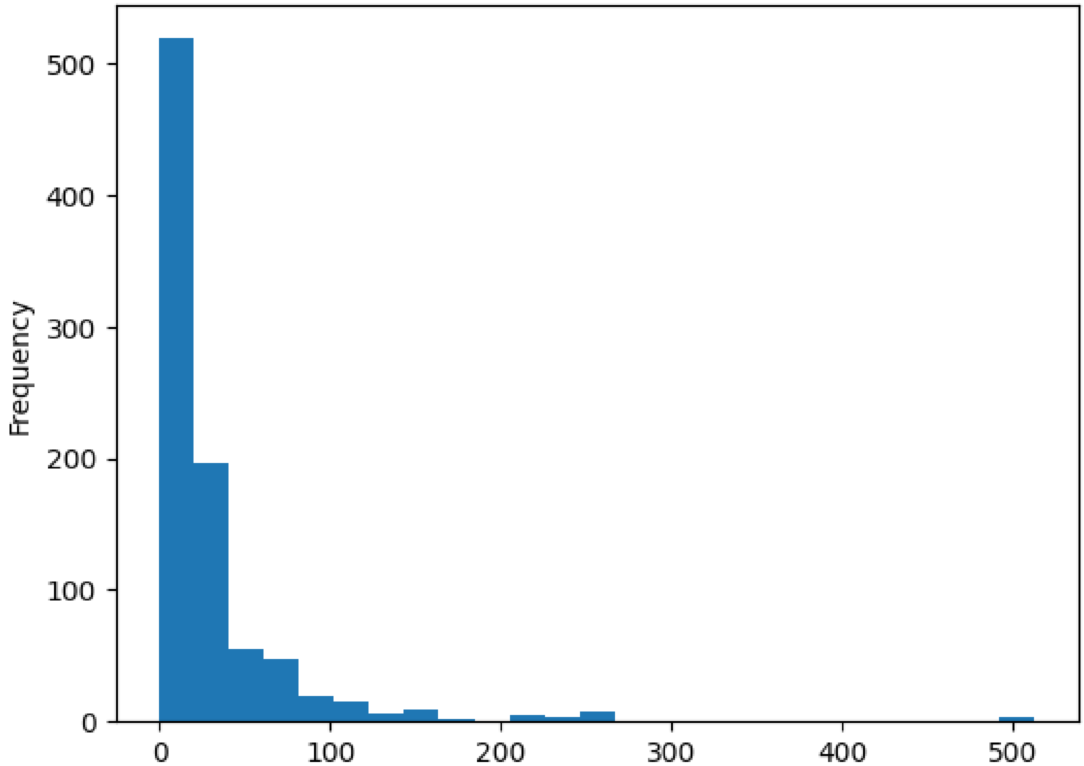
- Fare is highly positively skewed.
- Fare col actually contains group fare not individual fare.
- we need to create new col called individual fare.

```
df['Fare'].describe()
```

```
count    891.000000
mean      32.204208
std       49.693429
min        0.000000
25%        7.910400
50%       14.454200
75%       31.000000
max      512.329200
Name: Fare, dtype: float64
```

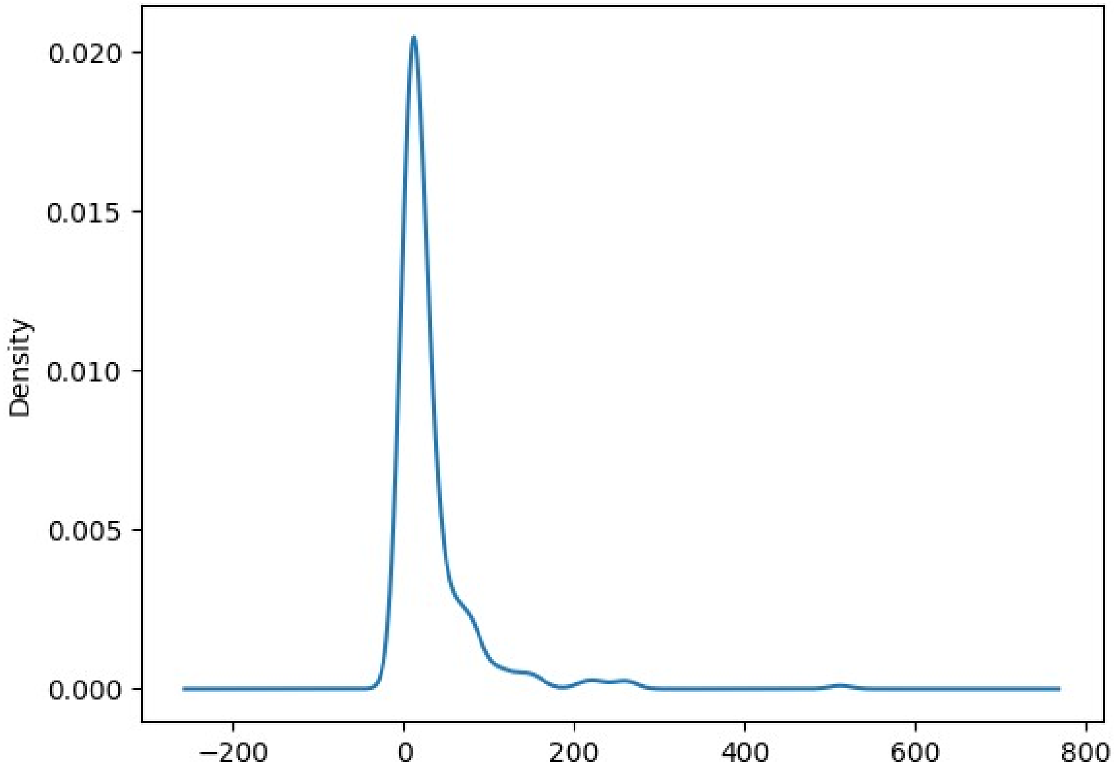
```
df['Fare'].plot(kind='hist', bins=25)
```

```
<Axes: ylabel='Frequency'>
```




```
df['Fare'].plot(kind='kde')
```

```
<Axes: ylabel='Density'>
```

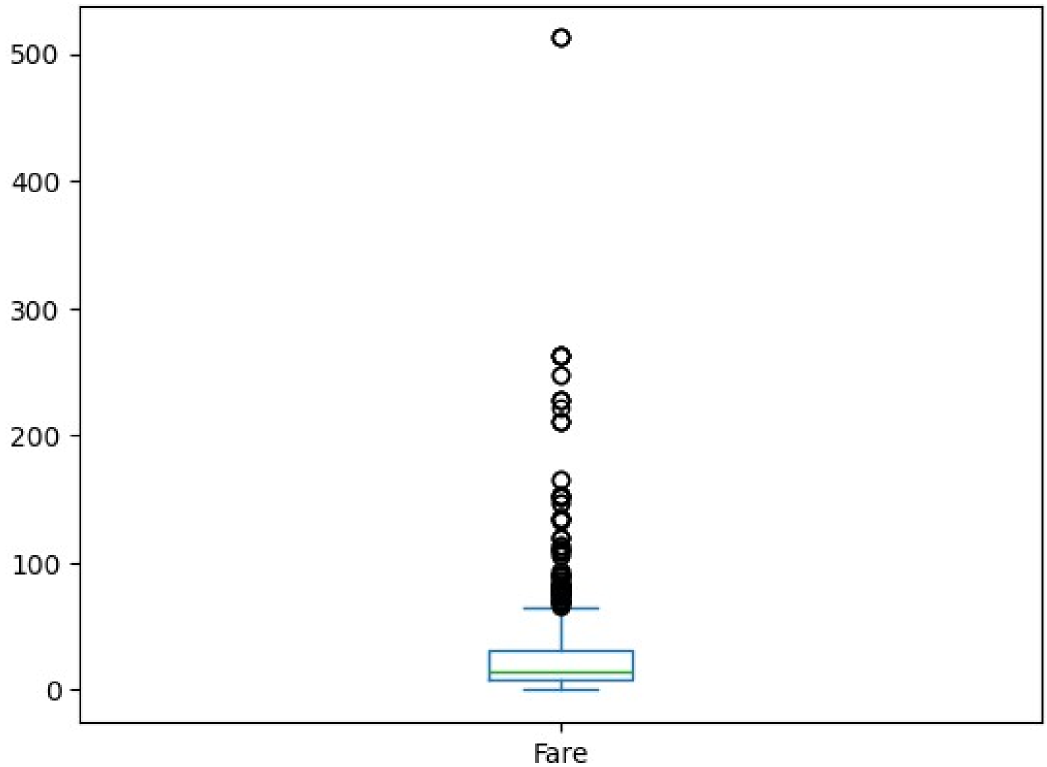



```
df['Fare'].skew()
```

```
4.787316519674893
```

```
df['Fare'].plot(kind='box')
```

```
<Axes: >
```




```
df[df['Fare']>250]

     PassengerId  Survived  Pclass
Name  \
27            28         0       1         Fortune, Mr. Charles
Alexander
88            89         1       1             Fortune, Miss. Mabel
Helen
258          259         1       1                       Ward, Miss.
Anna
311          312         1       1             Ryerson, Miss. Emily
Borie
341          342         1       1         Fortune, Miss. Alice
Elizabeth
438          439         0       1                      Fortune, Mr.
Mark
679          680         1       1     Cardeza, Mr. Thomas Drake
Martinez
737          738         1       1                 Lesurer, Mr.
Gustave J
742          743         1       1  Ryerson, Miss. Susan Parker
"Suzette"

        Sex    Age  SibSp  Parch    Ticket      Fare           Cabin
Embarked
```

```
27     male  19.0      3      2      19950  263.0000      C23 C25 C27
S
88   female  23.0      3      2      19950  263.0000      C23 C25 C27
S
258  female  35.0      0      0   PC 17755  512.3292              NaN
C
311  female  18.0      2      2   PC 17608  262.3750  B57 B59 B63 B66
C
341  female  24.0      3      2      19950  263.0000      C23 C25 C27
S
438    male  64.0      1      4      19950  263.0000      C23 C25 C27
S
679    male  36.0      0      1   PC 17755  512.3292      B51 B53 B55
C
737    male  35.0      0      0   PC 17755  512.3292             B101
C
742  female  21.0      2      2   PC 17608  262.3750  B57 B59 B63 B66
C
```

```
df['Fare'].isnull().sum()
```

```
0
```

Steps of doing Univariate Analysis on Categorical columns

Descriptive Statistics: Compute the frequency distribution of the categories in the column. This will give a general understanding of the distribution of the categories and their relative frequencies.

Visualizations: Create visualizations to explore the distribution of the categories. Some common visualizations for categorical data include count plots and pie charts. These visualizations provide a visual representation of the distribution of the categories and can help identify any patterns or anomalies in the data.

Missing Values: Check for missing values in the data and decide how to handle them. Missing values can be imputed or excluded from the analysis, depending on the research question and the data set.

Conclusion: Summarize the findings of the EDA and make decisions about how to proceed with further analysis.

Survived

conclusions

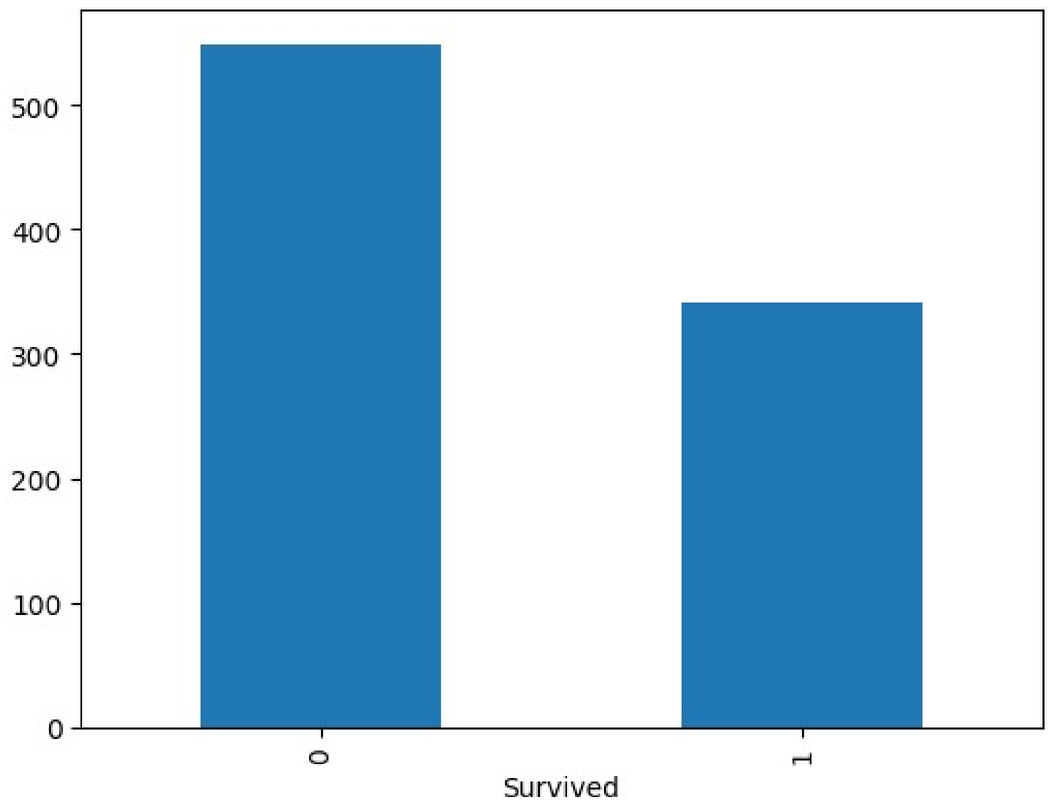
- almost 62% passesengers lost their lives while only 38% passesenger are alive.
- there are no null values.

```
df['Survived'].value_counts()
```

```
Survived
0    549
1    342
Name: count, dtype: int64
```

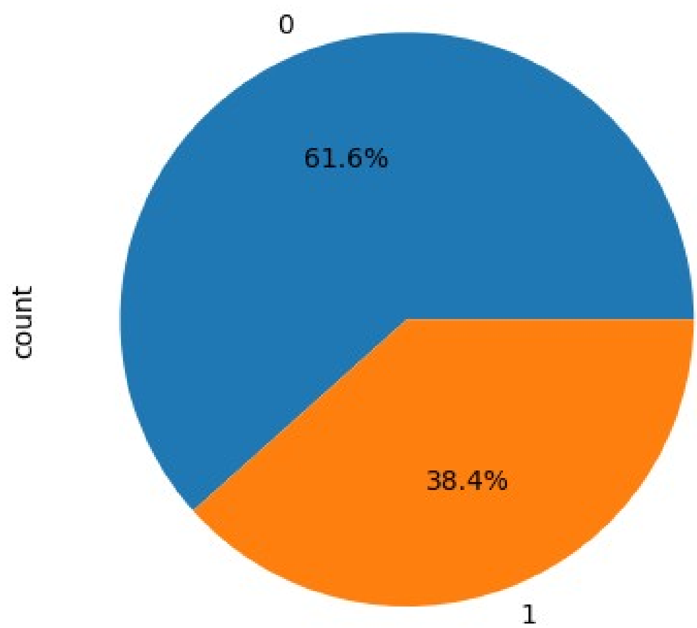
```
df['Survived'].value_counts().plot(kind='bar')
```

```
<Axes: xlabel='Survived'>
```




```
df['Survived'].value_counts().plot(kind='pie',autopct='%0.1f%%')
```

```
<Axes: ylabel='count'>
```




```python
df['Survived'].isnull().sum()
```

```
0
```

Pclass

conclusions

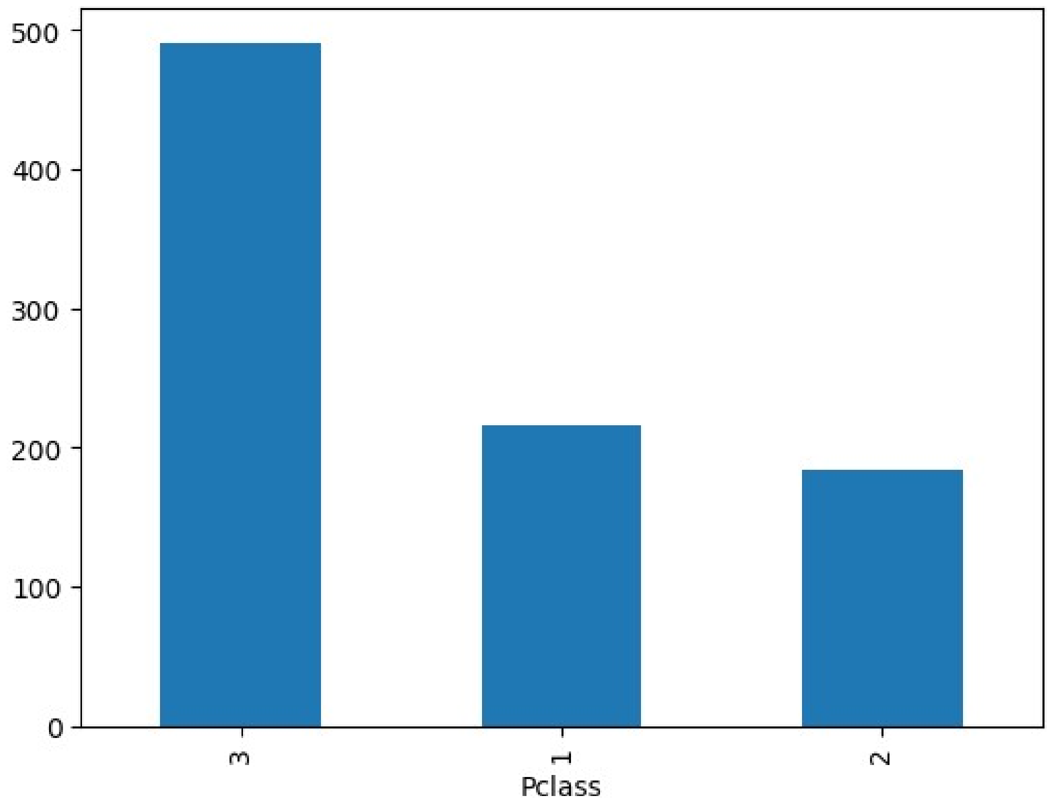
- Number of passengers in Pclass 1 is more than Pclass 2, which seems suspicious.
- Mostly num of passengers increase wrt to lower classes.

```python
df['Pclass'].value_counts()
```

```
Pclass
3    491
1    216
2    184
Name: count, dtype: int64
```

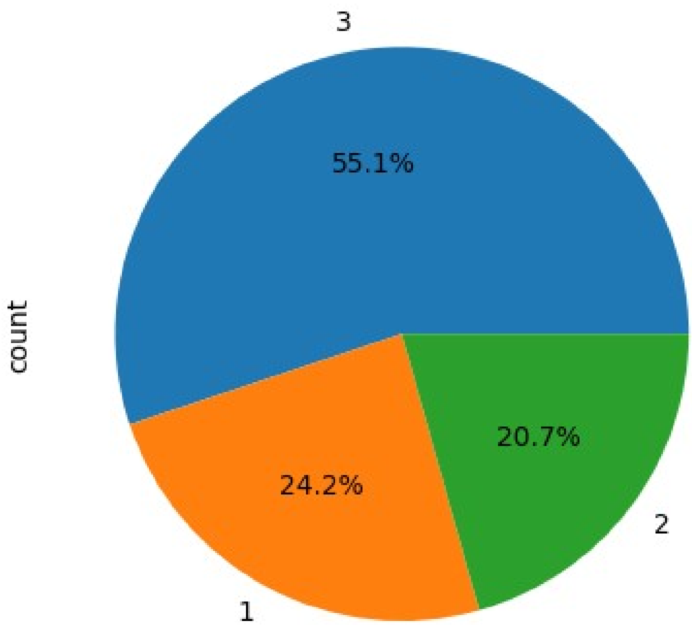
```python
df['Pclass'].value_counts().plot(kind='bar')
```

```
<Axes: xlabel='Pclass'>
```




```
df['Pclass'].value_counts().plot(kind='pie', autopct='%0.1f%%')
```

```
<Axes: ylabel='count'>
```




```
df['Pclass'].isnull().sum()
```

```
0
```

Sex

conclusions

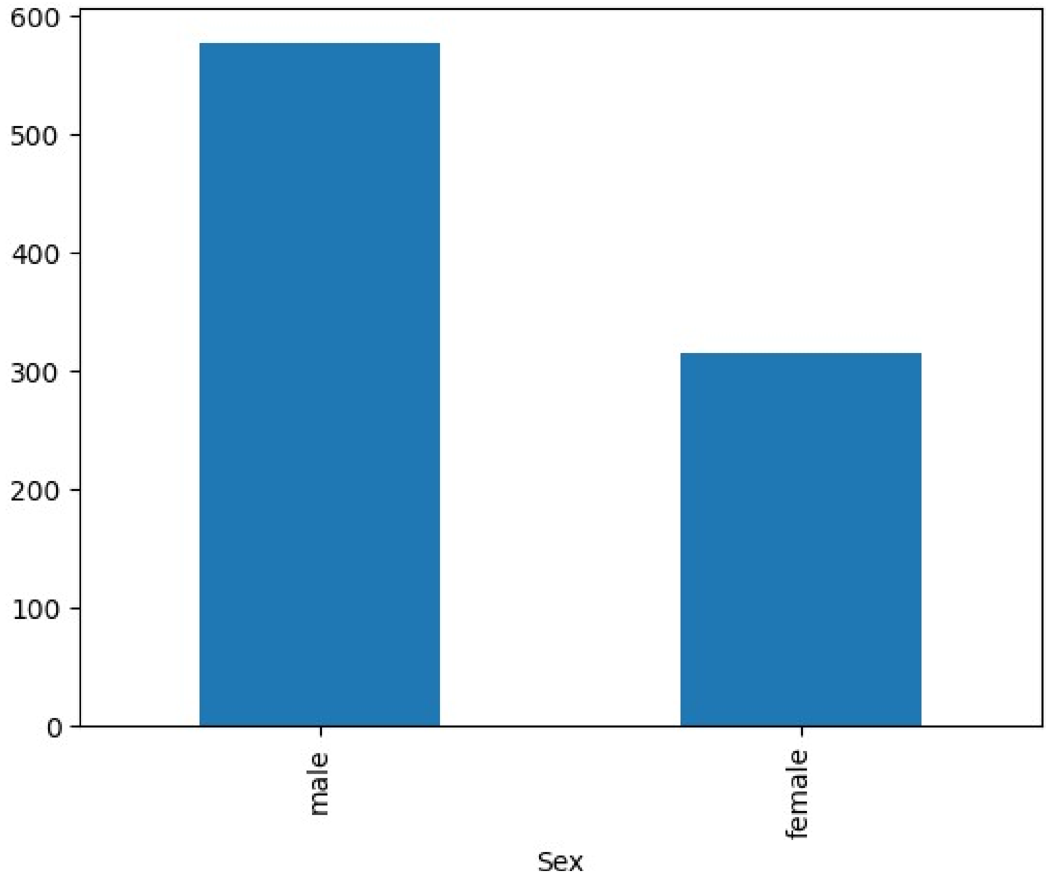
- 65% travellers are male and 35% travellers are female.

```
df['Sex'].value_counts()
```

```
Sex
male      577
female    314
Name: count, dtype: int64
```

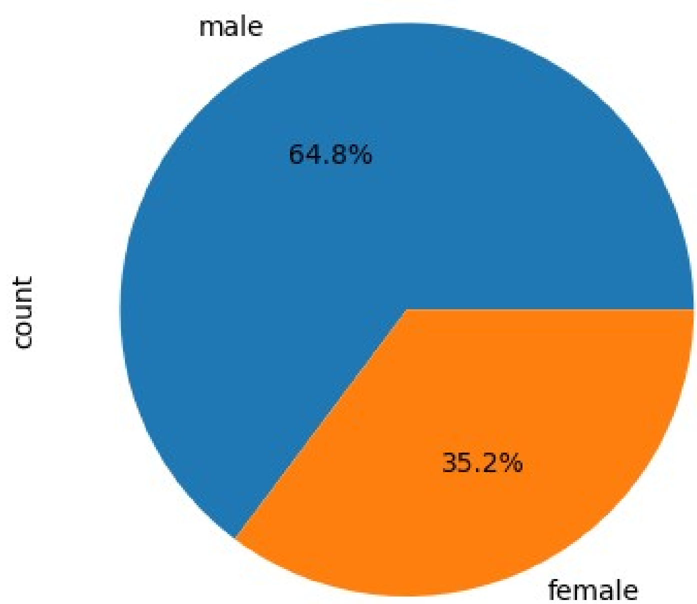
```
df['Sex'].value_counts().plot(kind='bar')
```

```
<Axes: xlabel='Sex'>
```




```
df['Sex'].value_counts().plot(kind='pie', autopct='%0.1f%%')
```

```
<Axes: ylabel='count'>
```




```
df['Sex'].isnull().sum()
```

```
0
```

Embarked

conclusions

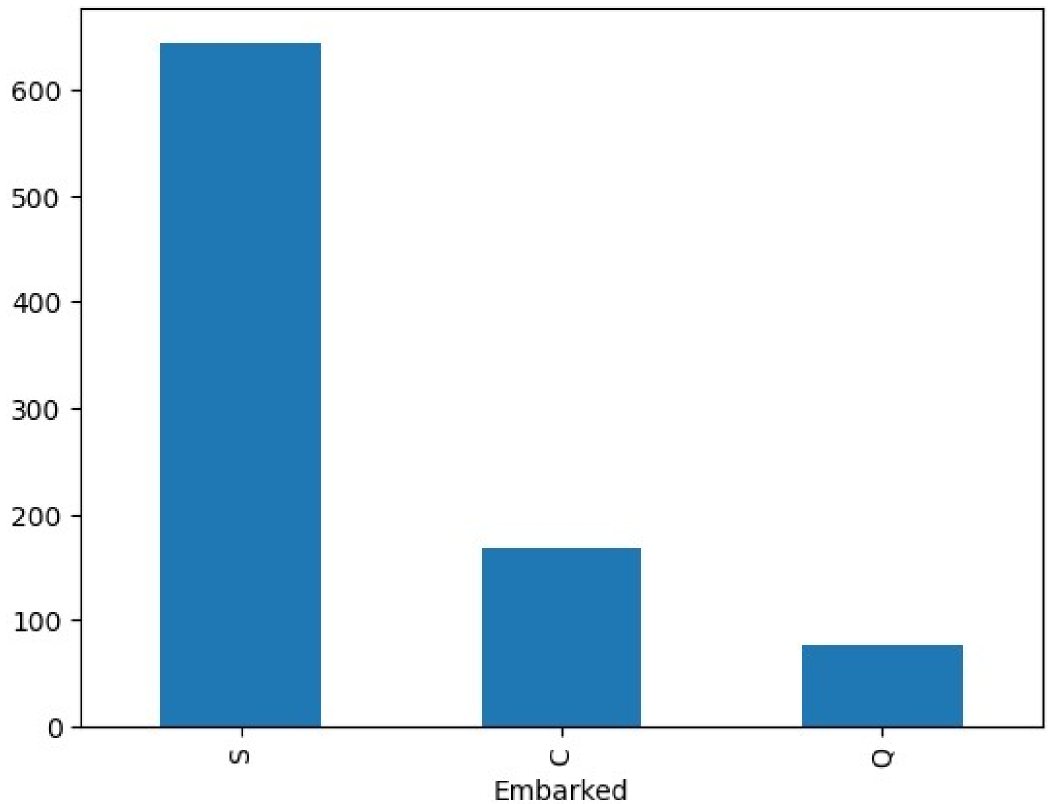
- There are 2 missing values.
- 72% peoples boarded from S
- 19% peoples boarded from C
- 9% peoples boarded from Q

```
df['Embarked'].value_counts()
```

```
Embarked
S    644
C    168
Q     77
Name: count, dtype: int64
```

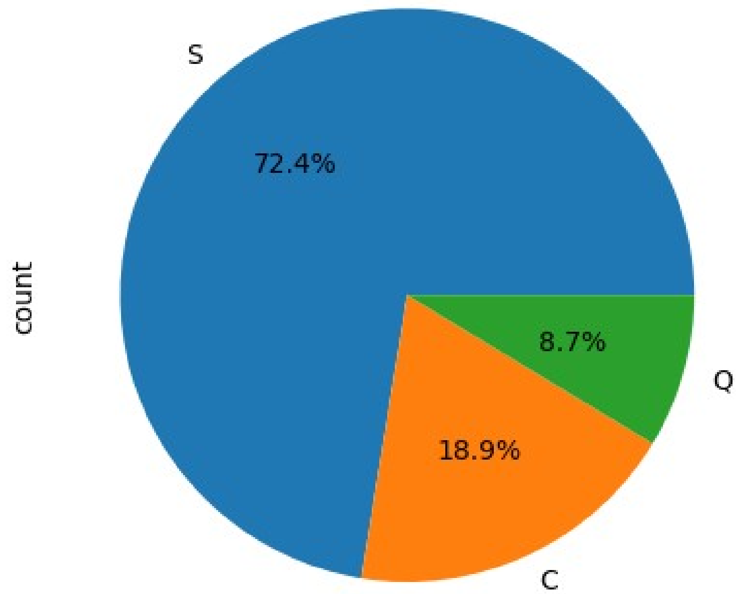
```
df['Embarked'].value_counts().plot(kind='bar')
```

```
<Axes: xlabel='Embarked'>
```




```python
df['Embarked'].value_counts().plot(kind='pie', autopct='%0.1f%%')
```

```
<Axes: ylabel='count'>
```




```
df['Embarked'].isnull().sum()
```

```
2
```

Steps of doing Bivariate Analysis

- Select 2 cols (Generally we select that col which is most important for prediction and one by one all other cols.)
- Understand type of relationship
    a. **Numerical - Numerical**
        a. You can plot graphs like scatterplot(regression plots), 2D histplot, 2D KDEplots
        b. Check correlation coefficent to check linear relationship
    b. **Numerical - Categorical** - create visualizations that compare the distribution of the numerical data across different categories of the categorical data.
        a. You can plot graphs like barplot, boxplot, kdeplot violinplot even scatterplots
    c. **Categorical - Categorical**
        a. You can create cross-tabulations or contingency tables that show the distribution of values in one categorical column, grouped by the values in the other categorical column.
        b. You can plots like heatmap, stacked barplots, treemaps
- Write your conclusions

Survived and Pclass

conclusions

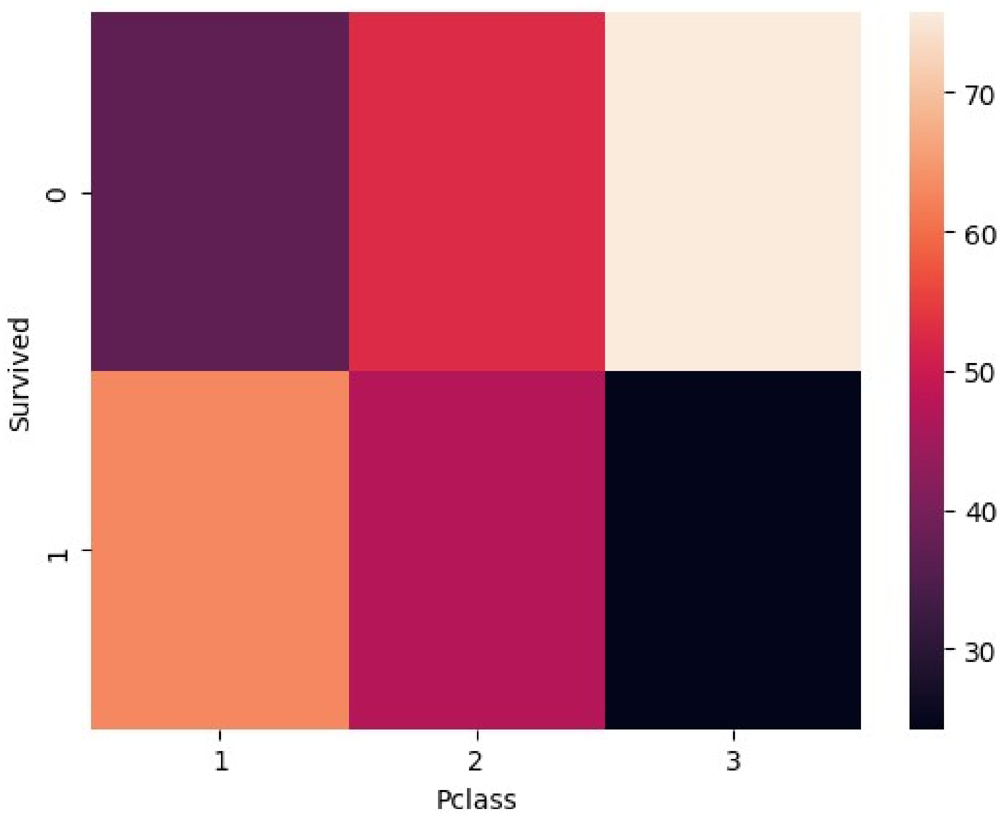
- Survival is highly dependent on Pclass
- 63% passengers of Pclass 1 has survived and 37% died
- 47% passengers of Pclass 2 has survived and 53% died
- 24% passengers of Pclass 1 has survived and 76% died
- Survival declines on increasing Pclass.

```
pd.crosstab(df['Survived'], df['Pclass'], normalize='columns')*100
```

```
Pclass            1          2          3
Survived
0         37.037037  52.717391  75.763747
1         62.962963  47.282609  24.236253
```

```
sns.heatmap(pd.crosstab(df['Survived'], df['Pclass'],
normalize='columns')*100)
```

```
<Axes: xlabel='Pclass', ylabel='Survived'>
```



Survived and Sex

conclusions

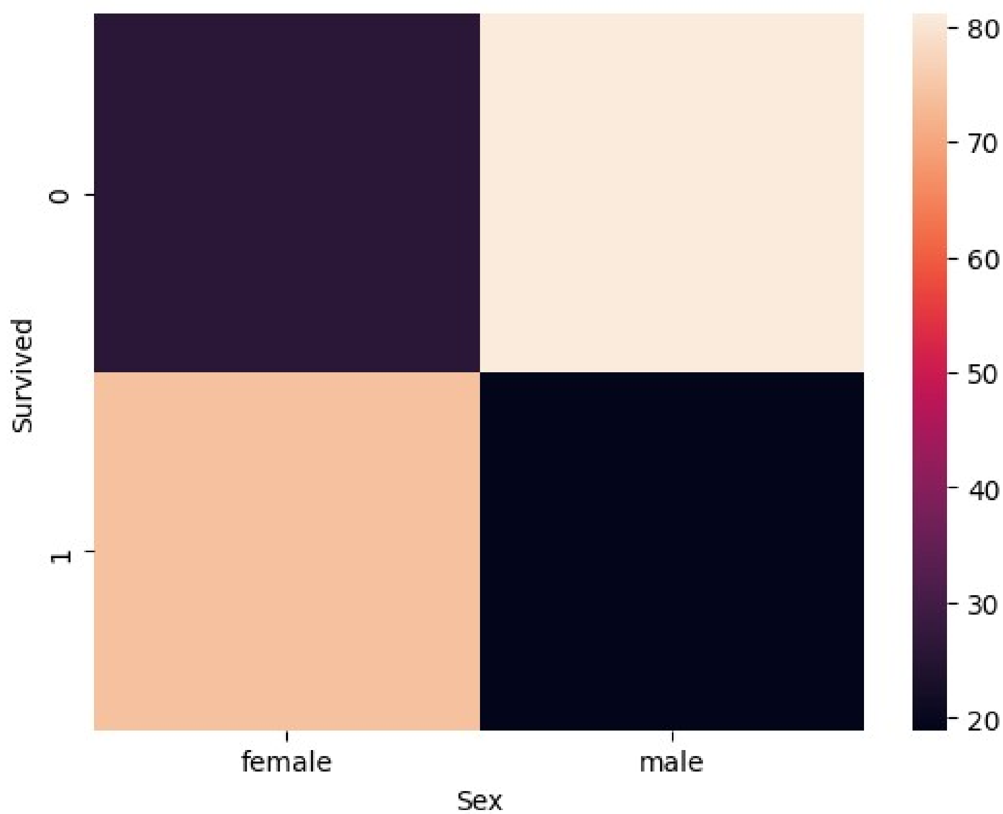
- Survival highly depends on Gender
- 74% females survived and 26% died.
- 19% males survived and 81% died.

```
pd.crosstab(df['Survived'], df['Sex'], normalize='columns')*100
```

Sex	female	male
Survived		
0	25.796178	81.109185
1	74.203822	18.890815

```
sns.heatmap(pd.crosstab(df['Survived'], df['Sex'],
normalize='columns')*100)
```

```
<Axes: xlabel='Sex', ylabel='Survived'>
```



Survived and Embarked

conclusions

- 55% passengeers survived who boarded from C and 45% died.
- 39% passengeers survived who boarded from C and 61% died.
- 34% passengeers survived who boarded from C and 66% died.
- It is suspicious because survival shouldn't be based on boarded station but however our analysis showing that there is something hidden. It may be due to that from C station passengers are females or Pclass 1

```
pd.crosstab(df['Survived'], df['Embarked'], normalize='columns')*100
```

```
Embarked          C          Q          S
Survived
0         44.642857  61.038961  66.304348
1         55.357143  38.961039  33.695652
```

```
pd.crosstab(df['Sex'], df['Embarked'], normalize='columns')*100
```

```
Embarked          C          Q          S
Sex
female    43.452381  46.753247  31.521739
male      56.547619  53.246753  68.478261
```

no answer to question because from Q there is also more females but their survival chances are low

```
pd.crosstab(df['Pclass'], df['Embarked'], normalize='columns')*100
```

```
Embarked          C          Q          S
Pclass
1         50.595238   2.597403  19.720497
2         10.119048   3.896104  25.465839
3         39.285714  93.506494  54.813665
```

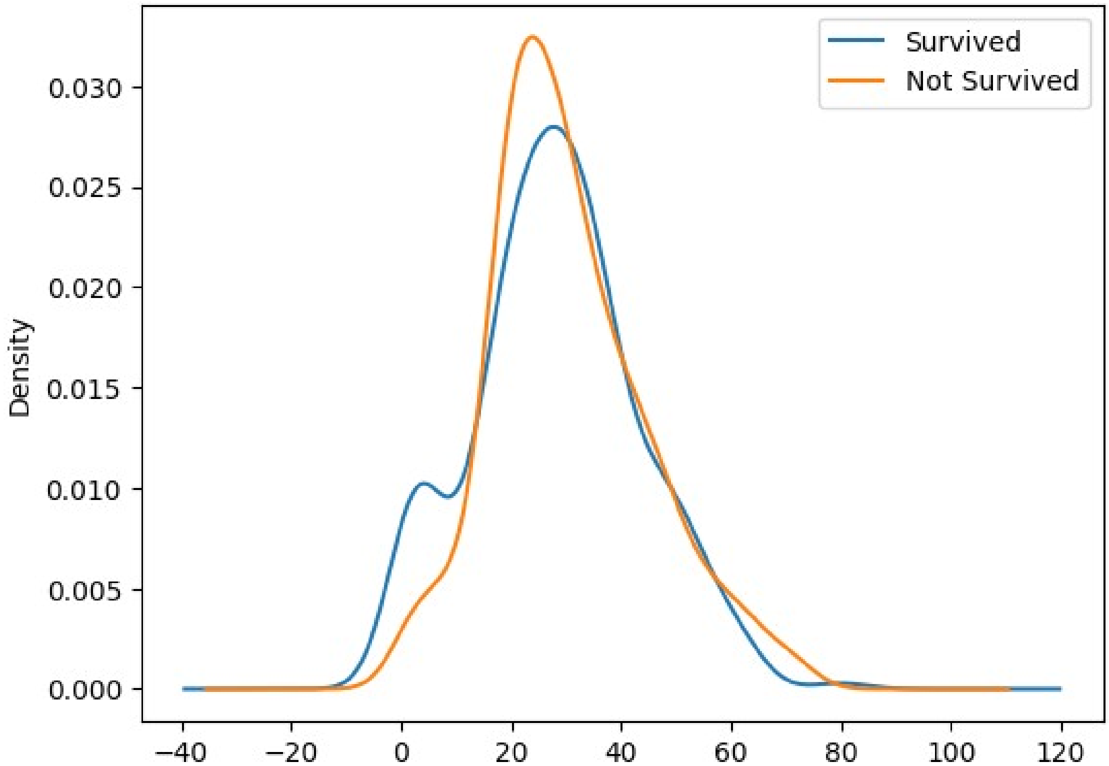
Here is the answer, above it is clear that the passengers from C are majority Pclass 1 travellers.

Survived and Age

conclusions

- if the age is below 10 then survival chances are more than dying
- if age is in 20 to 35 then chances of dying is more than survival
- if age is in 35 to 40 then chances of survival are more than dying. It may be due to Pclass.

```
df[df['Survived'] == 1]['Age'].plot(kind='kde', label='Survived')
df[df['Survived'] == 0]['Age'].plot(kind='kde', label='Not Survived')
plt.legend()
plt.show()
```




```
df[df['Pclass']==1]['Age'].mean()
```

```
38.233440860215055
```

Feature Engineering

Feature Engineering is the process of creating new columns from existing columns which will help in making predictions.

conclusions

- SibSb and Parch tells about the family or the passenges is alone so we can create new feature like family_size, family_type
- we can find individual fare based on family and Ticket

```
df['SibSp'].value_counts()
```

```
SibSp
0    608
1    209
2     28
4     18
3     16
8      7
5      5
Name: count, dtype: int64
```

```
df['Parch'].value_counts()
```

```
Parch
0    678
1    118
2     80
5      5
3      5
4      4
6      1
Name: count, dtype: int64
```

```
df[df['SibSp'] == 8]
```

```
     PassengerId  Survived  Pclass                               Name
Sex  \
159          160         0       3         Sage, Master. Thomas Henry
male
180          181         0       3       Sage, Miss. Constance Gladys
female
201          202         0       3                Sage, Mr. Frederick
male
324          325         0       3           Sage, Mr. George John Jr
male
792          793         0       3            Sage, Miss. Stella Anna
female
846          847         0       3           Sage, Mr. Douglas Bullen
male
863          864         0       3  Sage, Miss. Dorothy Edith "Dolly"
female

     Age  SibSp  Parch    Ticket   Fare Cabin Embarked
159  NaN      8      2  CA. 2343  69.55   NaN        S
180  NaN      8      2  CA. 2343  69.55   NaN        S
201  NaN      8      2  CA. 2343  69.55   NaN        S
324  NaN      8      2  CA. 2343  69.55   NaN        S
792  NaN      8      2  CA. 2343  69.55   NaN        S
846  NaN      8      2  CA. 2343  69.55   NaN        S
863  NaN      8      2  CA. 2343  69.55   NaN        S
```

In the above family there are total 11 members but in data there are only 7 persons, it is due to data split, because the whole titanic data is divided into two tables train.csv and test.csv

```
df1= pd.read_csv('Datasets/test2.csv')

df = pd.concat([df, df1])

df[df['SibSp'] == 8]
```

```
     PassengerId  Survived  Pclass                                Name
Sex  \
159          160       0.0       3         Sage, Master. Thomas Henry
male
180          181       0.0       3       Sage, Miss. Constance Gladys
female
201          202       0.0       3                Sage, Mr. Frederick
male
324          325       0.0       3           Sage, Mr. George John Jr
male
792          793       0.0       3            Sage, Miss. Stella Anna
female
846          847       0.0       3           Sage, Mr. Douglas Bullen
male
863          864       0.0       3  Sage, Miss. Dorothy Edith "Dolly"
female
188         1080       NaN       3                    Sage, Miss. Ada
female
360         1252       NaN       3        Sage, Master. William Henry
male

      Age  SibSp  Parch    Ticket   Fare Cabin Embarked
159   NaN      8      2  CA. 2343  69.55   NaN        S
180   NaN      8      2  CA. 2343  69.55   NaN        S
201   NaN      8      2  CA. 2343  69.55   NaN        S
324   NaN      8      2  CA. 2343  69.55   NaN        S
792   NaN      8      2  CA. 2343  69.55   NaN        S
846   NaN      8      2  CA. 2343  69.55   NaN        S
863   NaN      8      2  CA. 2343  69.55   NaN        S
188   NaN      8      2  CA. 2343  69.55   NaN        S
360  14.5      8      2  CA. 2343  69.55   NaN        S
```

```python
df['individual_fare'] = df['Fare']/(df['SibSp'] + df['Parch'] + 1)
```

```python
69.55/11
```

```
6.322727272727272
```

```python
df[df['SibSp'] == 8]
```

```
     PassengerId  Survived  Pclass                                Name
Sex  \
159          160       0.0       3         Sage, Master. Thomas Henry
male
180          181       0.0       3       Sage, Miss. Constance Gladys
female
201          202       0.0       3                Sage, Mr. Frederick
male
324          325       0.0       3           Sage, Mr. George John Jr
male
```

```
792          793        0.0        3            Sage, Miss. Stella Anna
female
846          847        0.0        3           Sage, Mr. Douglas Bullen
male
863          864        0.0        3  Sage, Miss. Dorothy Edith "Dolly"
female
188         1080        NaN        3                   Sage, Miss. Ada
female
360         1252        NaN        3       Sage, Master. William Henry
male

      Age  SibSp  Parch    Ticket   Fare Cabin Embarked
individual_fare
159   NaN      8      2  CA. 2343  69.55   NaN        S
6.322727
180   NaN      8      2  CA. 2343  69.55   NaN        S
6.322727
201   NaN      8      2  CA. 2343  69.55   NaN        S
6.322727
324   NaN      8      2  CA. 2343  69.55   NaN        S
6.322727
792   NaN      8      2  CA. 2343  69.55   NaN        S
6.322727
846   NaN      8      2  CA. 2343  69.55   NaN        S
6.322727
863   NaN      8      2  CA. 2343  69.55   NaN        S
6.322727
188   NaN      8      2  CA. 2343  69.55   NaN        S
6.322727
360  14.5      8      2  CA. 2343  69.55   NaN        S
6.322727
```

Now create family_size and family_type column
we are considering that if the family_size is 2 to 4 then it is small family else if family_size is more than 4 then it is big family

```
df['family_size'] = df['SibSp'] + df['Parch'] + 1

def transform_family_size(num):
    if num == 1:
        return 'alone'
    elif num > 1 and num < 5:
        return 'small'
    else:
        return 'big'

df['family_type'] = df['family_size'].apply(transform_family_size)
```

Now lets perform bivariate analysis on Survived and family_type columns

Survived and family_type

conclusions

- if passenger is alone than 30% chances of survival and 70% chances of dying.
- if family is small than chances of survival is 58% and 42% chances of dying.
- if family is big than chances of survival is 16% and 84% chances of dying.

```
pd.crosstab(df['Survived'], df['family_type'],
normalize='columns')*100

family_type      alone        big      small
Survived
0.0          69.646182  83.870968  42.123288
1.0          30.353818  16.129032  57.876712
```

Now Lets handle name column

```
df['Name']

0                                 Braund, Mr. Owen Harris
1       Cumings, Mrs. John Bradley (Florence Briggs Th...
2                                  Heikkinen, Miss. Laina
3            Futrelle, Mrs. Jacques Heath (Lily May Peel)
4                                Allen, Mr. William Henry
                              ...
413                                    Spector, Mr. Woolf
414                          Oliva y Ocana, Dona. Fermina
415                          Saether, Mr. Simon Sivertsen
416                                   Ware, Mr. Frederick
417                              Peter, Master. Michael J
Name: Name, Length: 1309, dtype: object

df['surname'] = df['Name'].str.split(',').str.get(0)

df['title'] =
df['Name'].str.split(',').str.get(1).str.strip().str.split('
').str.get(0)

df['title'].value_counts()

title
Mr.          757
Miss.        260
Mrs.         197
Master.       61
Rev.           8
Dr.            8
Col.           4
Mlle.          2
Major.         2
```

```
Ms.             2
Lady.           1
Sir.            1
Mme.            1
Don.            1
Capt.           1
the             1
Jonkheer.       1
Dona.           1
Name: count, dtype: int64
```

```
def categorize_title(title):
    if title in ['Mr.']:
        return 'Mr.'
    elif title in ['Ms.', 'Miss.']:
        return 'Ms.'
    elif title in ['Mrs.', 'Mme.']:
        return 'Mrs.'
    elif title in ['Master.']:
        return 'Master.'
    else:
        return 'other'

df['title'] = df['title'].apply(categorize_title)
```

Survived and title

conclusions

- results are similar like gender

```
pd.crosstab(df['Survived'],df['title'],normalize='columns')*100
```

```
title     Master.         Mr.       Mrs.        Ms.  other
Survived
0.0          42.5   84.332689  20.634921  30.054645   60.0
1.0          57.5   15.667311  79.365079  69.945355   40.0
```

Handling Cabin column
77% values are missing.

```
df['Cabin'].isnull().sum()/len(df['Cabin'])
```

```
0.774637127578304
```

```
df.fillna({'Cabin':'M'}, inplace=True)

df['Deck'] = df['Cabin'].str[0]

pd.crosstab(df['Deck'], df['Pclass'])
```

```
Pclass   1   2   3
Deck
A       22   0   0
B       65   0   0
C       94   0   0
D       40   6   0
E       34   4   3
F        0  13   8
G        0   0   5
M       67 254 693
T        1   0   0
```