Experiment No. 3

Aim: Data cleaning and data storage using python.

Theory: Data cleaning means fixing bad data in your data set.

Data cleaning is a crucial step in the data analysis process, particularly when it comes to analyzing social media data. Social media data sources such as Twitter, Facebook, Instagram, and LinkedIn are often messy, inconsistent, and contain a lot of noise. Therefore, cleaning the data before analyzing it is essential to ensure the accuracy and validity of the results.

Here are some of the commonly used data cleaning techniques in social media analytics: 1. Removing duplicates: Social media platforms generate a lot of redundant data, such as retweets, shares, and likes. Removing these duplicates can help simplify the data and reduce noise. 2. Filtering out spam: Social media is rife with spam content such as promotional posts, advertisements, and irrelevant comments. Removing spam can help improve the quality of the data.

- 3. Removing irrelevant content: Some social media data may be irrelevant to the research question, such as posts that are not related to the topic of interest. Removing irrelevant content can help narrow down the data and focus on the most relevant information.
- 4. Standardizing formats: Social media data may come in various formats, such as hashtags, mentions, or emojis. Standardizing these formats can help simplify the data and make it easier to analyze.
- 5. Correcting errors: Social media data may contain errors, such as misspellings, grammatical errors, or incomplete sentences. Correcting these errors can help improve the accuracy of the data.
- 6. Handling missing data: Social media data may contain missing values, such as empty fields or null values. Handling missing data can help avoid bias and improve the accuracy of the analysis.

In summary, data cleaning is an essential step in social media analytics. It helps ensure the accuracy and validity of the results by removing duplicates, filtering out spam, removing irrelevant content, standardizing formats, correcting errors, and handling missing data. Bad data could be:

- Empty cells
- Data in wrong format
- Wrong data
- Duplicates
- ➤ Dataset:

| | Duration | Date | Pulse | Maxpulse | Calories |
|----|----------|--------------|-------|----------|----------|
| 0 | 60 | '2020/12/01' | 110 | 130 | 409.1 |
| 1 | 60 | '2020/12/02' | 117 | 145 | 479.0 |
| 2 | 68 | '2020/12/03' | 103 | 135 | 340.0 |
| 3 | 45 | '2828/12/84' | 189 | 175 | 282.4 |
| 4 | 45 | '2020/12/05' | 117 | 148 | 486.0 |
| 5 | 68 | '2020/12/06' | 102 | 127 | 300.0 |
| 6 | 68 | '2020/12/07' | 110 | 136 | 374.0 |
| 7 | 450 | '2828/12/88' | 184 | 134 | 253.3 |
| | 30 | '2828/12/89' | 189 | 133 | 195.1 |
| 9 | 68 | '2020/12/10' | 98 | 124 | 269.0 |
| 10 | 68 | '2828/12/11' | 183 | 147 | 329.3 |
| 11 | 68 | '2828/12/12' | 188 | 128 | 250.7 |
| 12 | 60 | '2828/12/12' | 100 | 120 | 250.7 |
| 13 | 68 | '2020/12/13' | 186 | 128 | 345.3 |
| 14 | 68 | '2828/12/14' | 184 | 132 | 379.3 |
| 15 | 68 | '2828/12/15' | 98 | 123 | 275.0 |
| 16 | 60 | '2020/12/16' | 98 | 120 | 215.2 |
| 17 | 68 | '2020/12/17' | 100 | 120 | 300.0 |
| 18 | 45 | '2828/12/18' | 98 | 112. | NaN |
| 19 | 68 | '2828/12/19' | 183 | 123 | 323.0 |
| 28 | 45 | '2828/12/28' | 97 | 125 | 243.0 |
| 21 | 68 | '2020/12/21' | 188 | 131 | 364.2 |
| 22 | 45 | NaN | 100 | 119 | 282.0 |
| 23 | 60 | '2828/12/23' | 138 | 101 | 300.0 |
| 24 | 45 | '2020/12/24' | 105 | 132 | 246.0 |
| 25 | 68 | '2020/12/25' | 192 | 126 | 334.5 |
| 26 | 68 | 2828/12/26 | 100 | 128 | 258.8 |
| 27 | 68 | '2828/12/27' | 92 | 118 | 241.0 |
| 28 | 60 | '2020/12/28' | 103 | 132 | NaN |
| 29 | 68 | '2020/12/29' | 100 | 132. | 280.0 |
| 38 | 68 | '2828/12/38' | 102 | 129 | 380.3 |
| 31 | 68 | '2828/12/31' | 92 | 115 | 243.0 |

➤ **Remove Empty cells:**One way to deal with empty cells is to remove rows that contain empty cells. This is usually OK, since data sets can be very big, and removing a few rows will not have a big impact on the result.

Code:

 \triangleright

```
import pandas as pd
df = pd.read_csv('data.csv')
df.dropna(inplace = True)
print(df.to_string())
```

Output:

```
import pandas as pd
\mathbb{R}
     df = pd.read_csv('data.csv')
     new_df = df.dropna()
     print(new_df.to_string())
臣
         Ιď
              Duration
                                  Darte
                                         Pulse
                                                Maxpulse
                                                            Calories
                                                               489.1
     8
          8
                    68
                           2020/12/01
                                           118
                                                      130
          1
                          2020/12/02"
     1
                    60
                                           117
                                                      145
                                                               479.0
     2
          2
                          2020/12/03"
                    60
                                           103
                                                      135
                                                               340.0
                    45
                          2020/12/04*
                                           109
                                                      175
                                                               282.4
     4
          4
                    45
                          2020/12/05*
                                           117
                                                      148
                                                               486.8
     5
                    60
                          2020/12/06"
                                           182
                                                      127
                                                               300.0
     6
          6
                    60
                           2020/12/07
                                           110
                                                      136
                                                               374.0
                          2020/12/08"
                                           184
                                                      134
                                                               253.3
                   450
     8
          8
                    30
                          2020/12/09*
                                           189
                                                      133
                                                               195.1
     9
          9
                    60
                          2020/12/10"
                                            98
                                                      124
                                                               269.0
                    60
                          2020/12/11"
                                           103
                                                      147
                                                               329.3
     10
         10
     11
         11
                    60
                          2020/12/12*
                                           100
                                                      120
                                                               250.7
                          2020/12/12*
                                                               250.7
     12
         12
                    60
                                           100
                                                      120
                    60
                          2020/12/13"
                                           186
                                                      128
     13
         13
                                                               345.3
     14
         14
                    60
                          2020/12/14"
                                           164
                                                      132
                                                               379.3
     15
         15
                    60
                          2020/12/15"
                                            98
                                                      123
                                                               275.0
     16
         16
                    60
                          2020/12/16"
                                            98
                                                      129
                                                               215.2
                           2828/12/17
     17
         17
                    60
                                           166
                                                      120
                                                               300.0
                          2020/12/19"
     19
         19
                    60
                                           163
                                                      123
                                                               323.0
     20
         20
                    45
                          2020/12/20*
                                            97
                                                      125
                                                               243.0
     21
         21
                    60
                          2020/12/21"
                                           168
                                                      131
                                                               364.2
     23
         23
                    60
                          2020/12/23*
                                           138
                                                      181
                                                               300.0
     24
         24
                    45
                         "2020/12/24"
                                           185
                                                      132
                                                               246.0
     25
         25
                    60
                          2020/12/25"
                                           102
                                                      126
                                                               334.5
                           2020/12/26
                                           100
                                                      129
     26
         26
                    60
                                                               250.0
     27
         27
                    60
                           2020/12/27
                                            92
                                                      118
                                                               241.0
                    60
                          2020/12/29"
                                           166
     29
         29
                                                      132
                                                               280.0
                    60
                          2020/12/30"
                                           182.
                                                      129
     30
         30
                                                               380.3
                          2020/12/31"
                    60
                                            92.
     31
         31
                                                      115
                                                               243.0
```

➤ Convert Into a Correct Format: Cells with data of wrong format can make it difficult, or even impossible, to analyze data. To fix it, you have two options: remove the rows, or convert all cells in the columns into the same format In our Data Frame, we have two cells with the wrong format. Check out row 22 and 26, the 'Date' column should be a string that represents a date:

> Code:

```
import pandas as pd
df = pd.read_csv('data.csv')
df['Date'] = pd.to_datetime(df['Date'])
print(df.to_string())
```

Output:

| | Duration | Date | Pulse | Maxpulse | Calories |
|-----|----------|--------------|-------|----------|----------|
| | 60 | '2020/12/01' | 110 | 130 | 409.1 |
| 1. | 68 | "2020/12/02" | 117 | 345 | 479.0 |
| 2 | 60 | "2020/12/03" | 103 | 335 | 340.0 |
| - 8 | 45 | "2828/12/84" | 109 | 175 | 282.4 |
| 4 | 45 | "2020/12/05" | 117 | 148 | 406.0 |
| s | 60 | '2020/12/06' | 102 | 127 | 300.0 |
| 66 | 68 | "2828/12/87" | 110 | 136 | 374.0 |
| 7 | 450 | '2020/12/08' | 104 | 134 | 253.3 |
| | 30 | '2020/12/09' | 109 | 233 | 195.1 |
| 9 | 68 | '2828/12/18' | 98 | 124 | 269.0 |
| 10 | 60 | '2020/12/11' | 103 | 347 | 329.3 |
| 11 | 60 | "2020/12/12" | 100 | 128 | 250.7 |
| 12 | 68 | "2020/12/12" | 100 | 128 | 250.7 |
| 13 | 60 | '2020/12/13' | 106 | 128 | 345.3 |
| 14 | 68 | '2828/12/14' | 184 | 132 | 379.3 |
| 15 | 60 | '2020/12/15' | 98 | 123 | 275.0 |
| 16 | 60 | '2020/12/16' | 98 | 120 | 215.2 |
| 17 | 68 | '2828/12/17' | 100 | 128 | 300.0 |
| 18 | 45 | '2020/12/18' | 90 | 112 | NaN |
| 19 | 60 | '2020/12/19' | 103 | 123 | 323.0 |
| 28 | 45 | '2828/12/28' | 97 | 125 | 243.0 |
| 21 | 60 | '2020/12/21' | 108 | 131 | 364.2 |
| 22 | 45 | NaT | 100 | 119 | 282.0 |
| 23 | 60 | '2020/12/23' | 130 | 101 | 300.0 |
| 24 | 45 | '2020/12/24' | 105 | 132 | 246.0 |
| 25 | 60 | '2020/12/25' | 102 | 126 | 334.5 |
| 26 | 60 | '2020/12/26' | 100 | 120 | 250.0 |
| 27 | - | '2020/12/27' | 92 | 118 | 241.0 |
| 28 | 60 | '2020/12/28' | 103 | 132 | NaN |
| 29 | 60 | '2020/12/29' | 100 | 132 | 280.0 |
| 30 | • | '2020/12/30' | 102 | 129 | 380.3 |
| 31 | 68 | '2020/12/31' | 92 | 115 | 243.0 |
| | | | | | |

➤ Format Wrong Data: "Wrong data" does not have to be "empty cells" or "wrong format", it can just be wrong, like if someone registered "199" instead of "1.99". Sometimes you can spot wrong data by looking at the data set, because you have an expectation of what it should be. If you take a look at our data set, you can see that in row 7, the duration is 450, but for all the other rows the duration is between 30 and 60. It doesn't have to be wrong, but taking in consideration that this is the data set of someone's workout sessions, we conclude with the fact that this person did not work out in 450 minutes.

> Code:

```
Replace wrong value df.loc[7, 'Duration'] = 45
```

Output:

| | Duration | Date | Pulse | Maxpulse | Calories |
|----|----------|--------------|-------|----------|----------|
| 0 | 68 | '2828/12/81' | 110 | 130 | 489.1 |
| 1 | 60 | '2020/12/02' | 117 | 145 | 479.0 |
| 2 | 68 | "2020/12/03" | 103 | 135 | 340.0 |
| 3 | 45 | "2020/12/04" | 109 | 175 | 282.4 |
| 4 | 45 | "2020/12/05" | 117 | 148 | 486.0 |
| 5 | 60 | "2020/12/06" | 102 | 127 | 300.0 |
| 6 | 68 | "2020/12/07" | 110 | 136 | 374.0 |
| 7 | 45 | "2020/12/08" | 184 | 134 | 253.3 |
| 8 | 30 | '2828/12/89' | 109 | 133 | 195.1 |
| 9 | 60 | '2020/12/10' | 98 | 124 | 269.0 |
| 10 | 68 | '2020/12/11' | 103 | 147 | 329.3 |
| 11 | 68 | '2828/12/12' | 100 | 120 | 250.7 |
| 12 | 60 | '2020/12/12' | 100 | 120 | 250.7 |
| 13 | 68 | '2020/12/13' | 106 | 128 | 345.3 |
| 14 | 68 | '2020/12/14' | 184 | 132 | 379.3 |
| 15 | 60 | '2020/12/15' | 98 | 123 | 275.0 |
| 16 | 60 | '2020/12/16' | 98 | 120 | 215.2 |
| 17 | 68 | '2020/12/17' | 100 | 120 | 300.0 |
| 18 | 45 | '2828/12/18' | 98 | 112 | NaN |
| 19 | 60 | '2020/12/19' | 103 | 123 | 323.0 |
| 20 | 45 | "2020/12/20" | 97 | 125 | 243.0 |
| 21 | 68 | '2020/12/21' | 108 | 131 | 364.2 |
| 22 | 45 | NaN | 100 | 119 | 282.0 |
| 23 | 60 | '2020/12/23' | 130 | 101 | 300.0 |
| 24 | 45 | '2020/12/24' | 105 | 132 | 246.0 |
| 25 | 68 | '2020/12/25' | 102 | 126 | 334.5 |
| 26 | 60 | 20201226 | 100 | 120 | 250.0 |
| 27 | 60 | "2020/12/27" | 92 | 118 | 241.0 |
| 28 | 68 | "2020/12/28" | 103 | 132 | NaN |
| 29 | 68 | '2828/12/29' | 100 | 132 | 288.0 |
| 30 | 60 | '2828/12/38' | 102 | 129 | 380.3 |
| 31 | 60 | '2020/12/31' | 92 | 115 | 243.0 |
| | | | 1 . | 1 1 . | |

> Removing Duplicates:By taking a look at our test data set, we can assume that row 11 and 12 are duplicates. To discover duplicates, we can use the duplicated() method. The duplicated() method returns a Boolean values for each row:

 \triangleright

Code:

df.drop_duplicates(inplace = True)

Output:

| | Supplies. | | De Jane | Harris Tara | following. |
|-----|-----------|--------------|---------|-------------|------------|
| | Duration | Date | Pulse | Maxpulse | Calories |
| 0 | 60 | '2020/12/01' | 110 | 130 | 489.1 |
| 1 | 60 | '2020/12/02' | 117 | 145 | 479.0 |
| 2 | 68 | '2020/12/03' | 103 | 135 | 340.0 |
| 3 | 45 | '2828/12/84' | 189 | 175 | 282.4 |
| 4 | 45 | '2020/12/05' | 117 | 148 | 486.0 |
| 5 | 60 | '2020/12/06' | 102 | 127 | 300.0 |
| 6 | 68 | '2828/12/87' | 110 | 136 | 374.0 |
| 7 | 450 | '2828/12/88' | 184 | 134 | 253.3 |
| 8 | 30 | '2020/12/09' | 189 | 133 | 195.1 |
| 9 | 69 | '2020/12/10' | 98 | 124 | 269.0 |
| 10 | 68 | '2020/12/11' | 103 | 147 | 329.3 |
| 11 | 68 | '2828/12/12' | 188 | 120 | 250.7 |
| 13 | 60 | '2020/12/13' | 186 | 128 | 345.3 |
| 14 | 60 | '2020/12/14' | 184 | 132 | 379.3 |
| 15 | 68 | '2020/12/15' | 98 | 123 | 275.0 |
| 16 | 68 | '2828/12/16' | 98 | 120 | 215.2 |
| 17 | 60 | '2020/12/17' | 100 | 120 | 300.0 |
| 18 | 45 | '2020/12/18' | 90 | 112 | NaN |
| 19 | 60 | '2020/12/19' | 103 | 123 | 323.0 |
| 2.0 | 45 | '2828/12/28' | 97 | 125 | 243.0 |
| 21 | 60 | '2020/12/21' | 108 | 131 | 364.2 |
| 22 | 45 | NaN | 100 | 119 | 282.0 |
| 23 | 68 | '2020/12/23' | 130 | 101 | 380.0 |
| 24 | 45 | '2828/12/24' | 185 | 132 | 246.0 |
| 25 | 60 | '2020/12/25' | 102 | 126 | 334.5 |
| 26 | 69 | 20201226 | 100 | 120 | 250.0 |
| 27 | 68 | '2828/12/27' | 92 | 118 | 241.0 |
| 28 | 68 | '2828/12/28' | 183 | 132 | NaN |
| 29 | 60 | '2020/12/29' | 100 | 132 | 280.0 |
| 30 | 60 | '2020/12/30' | 102 | 129 | 380.3 |
| 31 | 68 | '2020/12/31' | 92 | 115 | 243.0 |
| | | | | | |

Conclusion- Data cleaning is a crucial step in the data preparation process, aimed at addressing and rectifying various issues within a dataset. The identification and correction of bad data, including empty cells, incorrect formats, erroneous values, and duplicates, are essential for ensuring the accuracy, reliability, and integrity of the data. Effective data cleaning practices contribute to improved data quality, leading to more meaningful and trustworthy insights when performing analyses or building machine learning models. By implementing techniques such as handling missing values, standardizing formats, validating data, and removing duplicates, data professionals can enhance the overall quality of their datasets and make informed decisions based on cleaner and more reliable information.