Special Topics: Data Analytics and Visualization in Healthcare

CSCI-GA.3033-096 (19635)

Lab assignment 2

Name: Deepali Chugh
NetID: dc4600

Part I

- Dataset used: 'Clinic_Patients_messy.csv'
- Errors found:

1. There is a single column named '**Systolic_BP&Diastolic_BP**', which has both **Systolic_BP** as well as **Diastolic_BP** values**.** The values are separated by '/'. Therefore, it is a better approach towards analyzing both the trends in the future, by splitting the values into separate columns.
    - Step taken:
        - Used the **Series.str.split (separator, expand = True)** method, where separator = **'/'**, Series = **data**
        - data[['Systolic_BP','Diastolic_BP']] = data['Systolic_BP&Diastolic_BP'].str.split('/',expand=True)
        - This is followed by dropping the original column 'Systolic_BP&Diastolic_BP' using drop() function.
        - data.drop('Systolic_BP&Diastolic_BP', axis=1, inplace=True)

2. Missing values in column 'Height' and 'Weight'.
    - Steps taken:
        - Used **fillna() function** to replace the null values of Height and Weight and fill with **new value 0**.
        - data['Height'].fillna(0 , inplace = True)
          data['Weight'].fillna(0 , inplace = True)

3. Data type of Gender, Diastolic_BP and Systolic_BP is 'Object'
    - Steps taken:
        - Used the **astype()** function to convert the datatype of Gender column from 'Object' to 'String', Systolic_BP from 'Object' to int and Diastolic_BP from 'Object' to int.
        - data['Gender'] = data['Gender'].astype('string')
        - data['Systolic_BP'] = data['Systolic_BP'].astype(int)
        - data['Diastolic_BP'] = data['Diastolic_BP'].astype(int)

4. Certain values in **column 'Age' have 'years'** mentioned in addition to the numeric value of Age. So, the approach is to remove the years keyword from the value and retain the numeric value only.
    - Steps taken:
        - Split the Age column, separated by a space ' ' and name the second column as 'Age2'.
        - Drop the column 'Age2'.
        - data[['Age','Age2']] = data['Age'].str.split(' ',expand=True)
          data.drop('Age2', axis=1, inplace=True)

5. Datatype of Age is ‘Object’, which should be converted to Integer.
    - Steps taken:
        - Used the **astype()** function to convert the datatype of Age to int.
        - data['Age'] = data['Age'].astype(int)

6. Filter rows by removing Height and Weight equal to 0, which was earlier filled for the missing values in the columns originally.
    - Steps:
        - data = (data[(data['Height'] != 0) & (data['Weight'] != 0)])

7. Outliers: Eliminating outliers by plotting in a boxplot and observing the outliers. Therefore, for the column ‘Age’, there is an outlier with two Age values that are 180 and 183.
    - Steps taken:
        - Filtered rows in the Age column where ‘Age’ is greater than 150.
        - data = (data[data['Age'] < 150])

Part II

- Dataset used: ‘covid-19-data.csv’
- Errors found:

1. Splitting Columns: There is a single column named ‘**Last Update**’, which has both **Date** as well as **Time** values**.** The values are separated by ‘ ’. Therefore, the single column has been split into separate columns **‘Last Update Date’** and **‘Last Update Time’**.
    - Step taken:
        - Used the **Series.str.split (separator, expand = True)** method, where separator = **‘ ’**, Series = **data**
        - data[['Last Update Date','Last Update Time']] = data['Last Update'].str.split(' ',expand=True)
        - This is followed by dropping the Last Update column.
        - data.drop('Last Update', axis=1, inplace=True)

2. Converting to datetime: Upon running the **data.info()** function, the ObservationDate column is checked to be of the type ‘Object’. This is converted to the datetime type.
    - Step taken:
        - Used the **to_datetime()** function from the pandas library.
        - data['ObservationDate'] = pd.to_datetime(data['ObservationDate'])

3. Upon running the data.info() function, the ‘Last Update Date’ column is checked to be of the type ‘Object’. This is converted to the datetime type.
    - Step taken:
        - Used the **to_datetime()** function from the pandas library.
        - data['Last Update Date'] = pd.to_datetime(data['Last Update Date'])

4. Missing values: There are 78100 null values in the ‘Province/State’ column, found using data.info() instruction.
    - Steps taken:

- ❖ Used the **fillna()** function to fill the null values in the 'Province/State' column to a new value of 'No address'.
- ❖ data['Province/State'].fillna('No address', inplace = True)

5. Converting to String type: Upon running the data.info() function, the 'Province/State' column is checked to be of the type 'Object'. This is converted to the string type.
   - Steps taken:
     - ❖ Used the **astype()** function to convert the datatype of 'Province/State' column from 'Object' to 'String'

6. Upon running the data.info() function, the 'Country/Region' column is checked to be of the type 'Object'. This is converted to the string type.
   - Steps taken:
     - ❖ Used the **astype()** function to convert the datatype of 'Country/Region' column from 'Object' to 'String'
     - ❖ data['Country/Region'] = data['Country/Region'].astype('string')

7. Upon running the data.info() function, the 'Last Update Time' column is checked to be of the type 'Object'. This is converted to the string type.
   - Steps taken:
     - ❖ Used the **astype()** function to convert the datatype of 'Last Update Time' column from 'Object' to 'String'
     - ❖ data['Last Update Time'] = data['Last Update Time'].astype('string')

8. Outliers: Eliminating outliers by plotting in a boxplot and observing the outliers. Therefore, for the column '**Recovered'**, there is an outlier with a value that is less than 0.
   - Steps taken:
     - ❖ Filtered rows in the Recovered column where 'Recovered' is greater than or equal to 0.
     - ❖ data = (data[(data['Recovered'] >= 0)])