2_Life_Expectancy_Initial_Data_Analysis

January 29, 2023

Table of Contents

1 Life_Expectancy_WHO_UN_Analysis_Modeling

1.1 Initial_Data_Analysis

To: Magnimind

From: Matt Curcio, matt.curcio.ri@gmail.com

Date: 2023-01-29

Re: NOTEBOOK #2

This python notebook simply takes the original data file `Life_Expectancy_Data.csv` and produces an intermediate file `Clean_LE_Data_w_Means_1.csv`.

List Of Features	Description
Year	Year
Country	Country
Status	Developed(1) or Developing(0)
GDP	Gross Domestic Product per capita (in USD)
TotalExpen	Total Expenditure
PercExpen	Percent Expenditure

List Of Features	Description
Income	Income composition of resources, Human Development Index
Population	Population of country
Education	Years of Education
EtOH	Alcohol consumption, litres of pure alcohol per capita
HepB	Hepatitis B: % immunization coverage among 1-year-olds
Measles	Number of reported cases per 1,000 population
DTP	Diphtheria, tetanus toxoid & pertussis % immunization coverage among 1-year-olds
Polio	Pol3: % immunization coverage among 1-year-olds
HIV	HIV/AIDS: Deaths per 1,000 (0-4 years)
BMI	Average Body Mass Index of entire population
AdultMort	Adult Mortality Rates of both sexes (probability of dying between 15 and 60 years per 1000 population)
InfD	Number of Infant Deaths per 1,000 population
lt5y_D	Number of under-five deaths per 1,000 population
Thin1_19y	% Prevalence of thinness among children and adolescents $10 < \text{Age} < 19$
Thin5_9y	% Prevalence of thinness among children and adolescents $5 < \text{Age} < 9$
LifeExpectancy	Life Expectancy (Yr)

[1]:
```
# Common Python Libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

[2]:
```
!ls *.csv
```

```
Clean_LE_Data_FEng_4.csv      Life_Expectancy_Data.csv  y_test.csv
Clean_LE_Data_Post_EDA_3.csv  x_test.csv                y_train.csv
Clean_LE_Data_w_Means_2.csv   x_train.csv
```

1.2 Load, rename columns, information

[3]:
```
filename = 'Life_Expectancy_Data.csv'

column_names = ['Country','Year','Status','LifeExpectancy','AdultMort',
                'InfD','EtOH','PercExpen','HepB','Measles',
                'BMI','lt5yD','Polio','TotalExpen','DTP','HIV',
                'GDP','Population','Thin1_19y','Thin5_9y','Income',
                'Education']

df = pd.read_csv(filename, names=column_names, header=0)

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2938 entries, 0 to 2937
Data columns (total 22 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   Country          2938 non-null   object 
 1   Year             2938 non-null   int64  
 2   Status           2938 non-null   object 
 3   LifeExpectancy   2928 non-null   float64
 4   AdultMort        2928 non-null   float64
 5   InfD             2938 non-null   int64  
 6   EtOH             2744 non-null   float64
 7   PercExpen        2938 non-null   float64
 8   HepB             2385 non-null   float64
 9   Measles          2938 non-null   int64  
 10  BMI              2904 non-null   float64
 11  lt5yD            2938 non-null   int64  
 12  Polio            2919 non-null   float64
 13  TotalExpen       2712 non-null   float64
 14  DTP              2919 non-null   float64
 15  HIV              2938 non-null   float64
 16  GDP              2490 non-null   float64
 17  Population       2286 non-null   float64
 18  Thin1_19y        2904 non-null   float64
 19  Thin5_9y         2904 non-null   float64
 20  Income           2771 non-null   float64
 21  Education        2775 non-null   float64
dtypes: float64(16), int64(4), object(2)
memory usage: 505.1+ KB
```

1.3 Check number of null values

[4]:
```
df.isnull().sum()
```

[4]:
```
Country              0
Year                 0
Status               0
LifeExpectancy      10
AdultMort           10
InfD                 0
EtOH               194
PercExpen            0
HepB               553
Measles              0
BMI                 34
lt5yD                0
Polio               19
TotalExpen         226
DTP                 19
HIV                  0
GDP                448
Population         652
Thin1_19y           34
Thin5_9y            34
Income             167
Education          163
dtype: int64
```

1.3.1 NOTE 1:

- The feature 'LifeExpectancy' has 10 missing values.
- Therefore the 10 rows that have NAN values will be deleted. Because 'LifeExpectancy' is a Dependent variable, I will delete those 10 observations with NO labels rather than impute them.
- Drop 10 rows containing null in `LifeExpectancy` column
- The 'LifeExpectancy' feature appear to be **Missing Completely at Random(MCAR)**. The main advantage of **MCAR** is that the analysis is unbiased. Data lost with design fault do not impact other parameters in the model.

[5]:
```
df.dropna(subset=['LifeExpectancy'], inplace=True) # 10 rows deleted
```

[6]:
```
print('\nDataframe df shape:', df.shape)

df.isnull().sum()
```

```
Dataframe df shape: (2928, 22)
```

[6]:
```
Country              0
Year                 0
Status               0
LifeExpectancy       0
AdultMort            0
InfD                 0
EtOH               193
PercExpen            0
HepB               553
Measles              0
BMI                 32
lt5yD                0
Polio               19
TotalExpen         226
DTP                 19
HIV                  0
GDP                443
Population         644
Thin1_19y           32
Thin5_9y            32
Income             160
Education          160
dtype: int64
```

1.3.2 NOTE 2:

- The five features with the highest number of mising vlaues (out of 2928) are:

	Feature	Number Missing	% Missing
1	Population	644	22.0%
2	HepB	553	18.9%
3	GDP	448	15.3%
4	TotalExpen	226	7.7%
5	EtOH	193	6.6%

1.3.3 NOTE 3:

- Drop feature columns ['Population', 'HepB', 'GDP'] where % Missing is greater than 15%.
- More data scraping or gathering needs to be done in at least 5 areas.
  - 1 Country Population
  - 2 Hepititus B Vaccination rates
  - 3 Gross Domestic Product
  - 4 Total Expenditure of Country Funds: Health Related
  - 5 Ethanol Comsumption per capita

[7]:
```
df.drop(['Population', 'HepB', 'GDP'], axis=1, inplace=True)

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2928 entries, 0 to 2937
Data columns (total 19 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Country          2928 non-null   object
 1   Year             2928 non-null   int64
 2   Status           2928 non-null   object
 3   LifeExpectancy   2928 non-null   float64
 4   AdultMort        2928 non-null   float64
 5   InfD             2928 non-null   int64
 6   EtOH             2735 non-null   float64
 7   PercExpen        2928 non-null   float64
 8   Measles          2928 non-null   int64
 9   BMI              2896 non-null   float64
 10  lt5yD            2928 non-null   int64
 11  Polio            2909 non-null   float64
 12  TotalExpen       2702 non-null   float64
 13  DTP              2909 non-null   float64
 14  HIV              2928 non-null   float64
 15  Thin1_19y        2896 non-null   float64
 16  Thin5_9y         2896 non-null   float64
 17  Income           2768 non-null   float64
 18  Education        2768 non-null   float64
dtypes: float64(13), int64(4), object(2)
memory usage: 457.5+ KB
```

1.4 Imputation using column means

[8]:
```
df['InfD'].fillna(np.mean(df.InfD), inplace=True)
df['EtOH'].fillna(np.mean(df.EtOH), inplace=True)
df['PercExpen'].fillna(np.mean(df.PercExpen), inplace=True)
df['Measles'].fillna(np.mean(df.Measles), inplace=True)
df['BMI'].fillna(np.mean(df.BMI), inplace=True)
df['Polio'].fillna(np.mean(df.Polio), inplace=True)
df['TotalExpen'].fillna(np.mean(df.TotalExpen), inplace=True)
df['DTP'].fillna(np.mean(df.DTP), inplace=True)
df['Thin1_19y'].fillna(np.mean(df.Thin1_19y), inplace=True)
df['Thin5_9y'].fillna(np.mean(df.Thin5_9y), inplace=True)
df['Income'].fillna(np.mean(df.Income), inplace=True)
df['Education'].fillna(np.mean(df.Education), inplace=True)
```

```
# Convert Dev status to a binary variable, where Developing Nations = 0,␣
 ↪Developed Nation status = 1.
df['Status'] = df['Status'].apply(lambda x: 0 if x == 'Developing' else 1).
 ↪astype('int8')
```

1.5 Re-Check null data points

[9]:
```
print('\nShape of Cleaned and Imputed dataframe:', df.shape)

df.isnull().sum()
```

```
Shape of Cleaned and Imputed dataframe: (2928, 19)
```

[9]:
```
Country          0
Year             0
Status           0
LifeExpectancy   0
AdultMort        0
InfD             0
EtOH             0
PercExpen        0
Measles          0
BMI              0
lt5yD            0
Polio            0
TotalExpen       0
DTP              0
HIV              0
Thin1_19y        0
Thin5_9y         0
Income           0
Education        0
dtype: int64
```

1.6 Save intermediate dataframe

[10]:
```
df.to_csv("Clean_LE_Data_w_Means_2.csv", index=False)
```

[11]:
```
!ls *.csv
```

```
Clean_LE_Data_FEng_4.csv       Life_Expectancy_Data.csv  y_test.csv
Clean_LE_Data_Post_EDA_3.csv   x_test.csv                y_train.csv
Clean_LE_Data_w_Means_2.csv    x_train.csv
```

[12]:
```
df.head(3)
```

[12]:

	Country	Year	Status	LifeExpectancy	AdultMort	InfD	EtOH	\
0	Afghanistan	2015	0	65.0	263.0	62	0.01	
1	Afghanistan	2014	0	59.9	271.0	64	0.01	
2	Afghanistan	2013	0	59.9	268.0	66	0.01	

	PercExpen	Measles	BMI	lt5yD	Polio	TotalExpen	DTP	HIV	Thin1_19y	\
0	71.279624	1154	19.1	83	6.0	8.16	65.0	0.1	17.2	
1	73.523582	492	18.6	86	58.0	8.18	62.0	0.1	17.5	
2	73.219243	430	18.1	89	62.0	8.13	64.0	0.1	17.7	

	Thin5_9y	Income	Education
0	17.3	0.479	10.1
1	17.5	0.476	10.0
2	17.7	0.470	9.9

[ ]: