1. Data Transformation & EDA

March 7, 2022

0.1 1. Preliminary Exploratory Data Analysis (EDA)

[ ]:
```
import numpy as np
import pandas as pd
from pandas_profiling import ProfileReport

!pip3 install pandas_profiling --upgrade
```

```
Requirement already satisfied: pandas_profiling in /usr/local/lib/python3.7
/dist-packages (3.1.0)
Requirement already satisfied: numpy>=1.16.0 in /usr/local/lib/python3.7/dist-
packages (from pandas_profiling) (1.21.5)
Requirement already satisfied: pandas!=1.0.0,!=1.0.1,!=1.0.2,!=1.1.0,>=0.25.3 in
/usr/local/lib/python3.7/dist-packages (from pandas_profiling) (1.3.5)
Requirement already satisfied: visions[type_image_path]==0.7.4 in
/usr/local/lib/python3.7/dist-packages (from pandas_profiling) (0.7.4)
Requirement already satisfied: requests>=2.24.0 in /usr/local/lib/python3.7
/dist-packages (from pandas_profiling) (2.27.1)
Requirement already satisfied: pydantic>=1.8.1 in /usr/local/lib/python3.7/dist-
packages (from pandas_profiling) (1.9.0)
Requirement already satisfied: missingno>=0.4.2 in /usr/local/lib/python3.7
/dist-packages (from pandas_profiling) (0.5.1)
Requirement already satisfied: scipy>=1.4.1 in /usr/local/lib/python3.7/dist-
packages (from pandas_profiling) (1.7.3)
Requirement already satisfied: tqdm>=4.48.2 in /usr/local/lib/python3.7/dist-
packages (from pandas_profiling) (4.63.0)
Requirement already satisfied: markupsafe~=2.0.1 in /usr/local/lib/python3.7
/dist-packages (from pandas_profiling) (2.0.1)
Requirement already satisfied: seaborn>=0.10.1 in /usr/local/lib/python3.7/dist-
packages (from pandas_profiling) (0.11.2)
Requirement already satisfied: PyYAML>=5.0.0 in /usr/local/lib/python3.7/dist-
packages (from pandas_profiling) (6.0)
Requirement already satisfied: phik>=0.11.1 in /usr/local/lib/python3.7/dist-
packages (from pandas_profiling) (0.12.0)
Requirement already satisfied: htmlmin>=0.1.12 in /usr/local/lib/python3.7/dist-
packages (from pandas_profiling) (0.1.12)
Requirement already satisfied: joblib~=1.0.1 in /usr/local/lib/python3.7/dist-
```

```
packages (from pandas_profiling) (1.0.1)
Requirement already satisfied: tangled-up-in-unicode==0.1.0 in
/usr/local/lib/python3.7/dist-packages (from pandas_profiling) (0.1.0)
Requirement already satisfied: matplotlib>=3.2.0 in /usr/local/lib/python3.7
/dist-packages (from pandas_profiling) (3.2.2)
Requirement already satisfied: multimethod>=1.4 in /usr/local/lib/python3.7
/dist-packages (from pandas_profiling) (1.7)
Requirement already satisfied: jinja2>=2.11.1 in /usr/local/lib/python3.7/dist-
packages (from pandas_profiling) (2.11.3)
Requirement already satisfied: attrs>=19.3.0 in /usr/local/lib/python3.7/dist-
packages (from visions[type_image_path]==0.7.4->pandas_profiling) (21.4.0)
Requirement already satisfied: networkx>=2.4 in /usr/local/lib/python3.7/dist-
packages (from visions[type_image_path]==0.7.4->pandas_profiling) (2.6.3)
Requirement already satisfied: imagehash in /usr/local/lib/python3.7/dist-
packages (from visions[type_image_path]==0.7.4->pandas_profiling) (4.2.1)
Requirement already satisfied: Pillow in /usr/local/lib/python3.7/dist-packages
(from visions[type_image_path]==0.7.4->pandas_profiling) (7.1.2)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in
/usr/local/lib/python3.7/dist-packages (from
matplotlib>=3.2.0->pandas_profiling) (3.0.7)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7
/dist-packages (from matplotlib>=3.2.0->pandas_profiling) (1.3.2)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-
packages (from matplotlib>=3.2.0->pandas_profiling) (0.11.0)
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.7
/dist-packages (from matplotlib>=3.2.0->pandas_profiling) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-
packages (from pandas!=1.0.0,!=1.0.1,!=1.0.2,!=1.1.0,>=0.25.3->pandas_profiling)
(2018.9)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
/usr/local/lib/python3.7/dist-packages (from pydantic>=1.8.1->pandas_profiling)
(3.10.0.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-
packages (from python-dateutil>=2.1->matplotlib>=3.2.0->pandas_profiling)
(1.15.0)
Requirement already satisfied: charset-normalizer~=2.0.0 in
/usr/local/lib/python3.7/dist-packages (from requests>=2.24.0->pandas_profiling)
(2.0.12)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.7
/dist-packages (from requests>=2.24.0->pandas_profiling) (1.24.3)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7
/dist-packages (from requests>=2.24.0->pandas_profiling) (2021.10.8)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.7/dist-
packages (from requests>=2.24.0->pandas_profiling) (2.10)
Requirement already satisfied: PyWavelets in /usr/local/lib/python3.7/dist-
packages (from imagehash->visions[type_image_path]==0.7.4->pandas_profiling)
(1.2.0)
```

Initial exploratory data analysis of the original Public Use Microdata File (PUMF) dataset was conducted by reviewing the descriptive statistics provided in Statistics Canada's Data Dictionary document that accompanied the PUMF. The Data Dictionary listed all variables in the PUMF dataset, along with the answer categories (i.e. classes), codes utilized for each category, response frequencies, and percentages.

[ ]:
```
#Read file
df = pd.read_csv('HS.csv', index_col=None)
```

[ ]:
```
profile = ProfileReport(df, minimal=True)
profile.to_file(output_file="1_Raw Dataset Profile.html")
```

```
Summarize dataset:   0%|          | 0/5 [00:00<?, ?it/s]

Generate report structure:   0%|          | 0/1 [00:00<?, ?it/s]

Render HTML:   0%|          | 0/1 [00:00<?, ?it/s]

Export report to file:   0%|          | 0/1 [00:00<?, ?it/s]
```

1 2. Data Transformation

After examination of the Data Dictionary and the questionnaire used to gather responses, decisions were made on the appropriate process to clean and transform the data to remove irrelavant variables, delete invalid responses, and address missing values. The remainder of this section contains the code for this process.

[ ]:
```
#Drop rows containing observations that are not valid for the project
df.drop(df[df['EMP_30'] != 1].index, inplace = True)
df.drop(df[df['GEN_10'] == 9].index, inplace = True)
df.drop(df[df['GEN_15'] == 9].index, inplace = True)
```

[ ]:
```
#Create new derived variable, which will be the target variable for this␣
 ↪project
df['Worse_MH'] = np.where((df['GEN_10'] >= 4) & (df['GEN_15'] >= 4), 0, 1)
```

[ ]:
```
#Drop columns pertaining to variables that are not needed for the project
df = df.
 ↪drop(columns=['PUMFID','VERDATE','EMP_05','BMF_P','EMP_30','ENV_25A','ENV_25B','ENV_25C','G

#Drop columns pertaining to derived variables that are not needed for the␣
 ↪project
df = df.
 ↪drop(columns=['PPEDVEY1','PPEDVEY2','PPEDVFV1','PPEDVFV2','PPEDVGL1','PPEDVGL2','PPEDVGN1',
                ␣
 ↪'PPEDVOT1','PPEDVOT2','PPEDVRE1','PPEDVRE2','PPEDVRS1','PPEDVRS2','GENDVHDI','GENDVMHI'])
```

[ ]:
```
#Reassignment of "Valid Skip" class, for variables pertaining to questions that␣
↪were not asked of respondents based on their previous responses

#If ENV_30 = Valid Skip = 6, then reassign to No = 2
df['ENV_30'].mask(df['ENV_30'] == 6, 2, inplace=True)

#If PPE_10 = No = 2, then PPE_15A, PPE_15B, PPE_15C, PPE_15D, PPE_15E, PPE_15F,␣
↪PPE_15G, PPE_15H, PPE_15I, and PPE_15J = No = 2 (instead of "Valid Skip")
skip_set_1 =␣
↪['PPE_15A','PPE_15B','PPE_15C','PPE_15D','PPE_15E','PPE_15F','PPE_15G','PPE_15H','PPE_15I',
df[skip_set_1] = np.where(df[['PPE_10']] == 2, 2, df[skip_set_1])

#If PPE_20 = No = 2, then PPE_30A, PPE_30B, PPE_30C, PPE_30D, PPE_30E, PPE_30F,␣
↪PPE_30G, PPE_30H,...
#PPE_35A, PPE_35B, PPE_35C, PPE_35D, PPE_35E, PPE_35F, PPE_35G, PPE_35H = "Not␣
↪needed for job" = 1 (instead of "Valid Skip")
skip_set_2 =␣
↪['PPE_30A','PPE_30B','PPE_30C','PPE_30D','PPE_30E','PPE_30F','PPE_30G','PPE_30H','PPE_35A',
df[skip_set_2] = np.where(df[['PPE_20']] == 2, 1, df[skip_set_2])

#If PPE_20 = No = 2, then PPE_25, PPE_40A, PPE_40B, PPE_40C, PPE_40D, PPE_40E,␣
↪PPE_40F, PPE_40G, PPE_40H, PPE_40I,...
#PPE_45A, PPE_45B, PPE_45C, PPE_45D, PPE_45E, PPE_45F, PPE_45G, PPE_45H,␣
↪PPE_45I = "No" = 2 (instead of "Valid Skip")
skip_set_3 =␣
↪['PPE_25','PPE_40A','PPE_40B','PPE_40C','PPE_40D','PPE_40E','PPE_40F','PPE_40G','PPE_40H','
            ␣
↪'PPE_45B','PPE_45C','PPE_45D','PPE_45E','PPE_45F','PPE_45G','PPE_45H','PPE_45I']
df[skip_set_3] = np.where(df[['PPE_20']] == 2, 2, df[skip_set_3])
```

[ ]:
```
#Reassignment of "Not stated" class (i.e. missing value category), for␣
↪variables pertaining to questions that respondents did not answer

#If response to PPE access question = Not stated = 9, then update to "Not␣
↪needed for job" = 1
missing_set_1 =␣
↪['PPE_30A','PPE_30B','PPE_30C','PPE_30D','PPE_30E','PPE_30F','PPE_30G','PPE_30H','PPE_35A',
for col in missing_set_1:
  df[col].mask(df[col] == 9, 1, inplace=True)

#If response to PPE restriction question = Not stated = 9, then update to "No"␣
↪= 2
missing_set_2 =␣
↪['PPE_40A','PPE_40B','PPE_40C','PPE_40D','PPE_40E','PPE_40F','PPE_40G','PPE_40H','PPE_40I','
              ␣
↪'PPE_45A','PPE_45B','PPE_45C','PPE_45D','PPE_45E','PPE_45F','PPE_45G','PPE_45H','PPE_45I']
```

```python
for col in missing_set_2:
  df[col].mask(df[col] == 9, 2, inplace=True)

#If response to PPE or IPC protocol/practice question = Not stated = 9, then␣
↪remove from dataset
df.drop(df[(df['PPE_05'] == 9)|(df['PPE_10'] == 9)|(df['PPE_15A'] ==␣
↪9)|(df['PPE_15B'] == 9)|(df['PPE_15C'] == 9)|(df['PPE_15D'] ==␣
↪9)|(df['PPE_15E'] == 9)|\
           (df['PPE_15F'] == 9)|(df['PPE_15G'] == 9)|(df['PPE_15H'] ==␣
↪9)|(df['PPE_15I'] == 9)|(df['PPE_15J'] == 9)].index, inplace = True)
df.drop(df[(df['PPE_20'] == 9)|(df['PPE_25'] == 9)].index, inplace = True)
df.drop(df[(df['PPE_50A'] >= 96)|(df['PPE_50B'] >= 96)|(df['PPE_50C'] >=␣
↪96)|(df['PPE_50D'] >= 96)|(df['PPE_50E'] >= 96)|(df['PPE_50F'] >= 96)].
↪index, inplace = True)

#Handling of 15/16 remaining variables with classes of "Not stated" = 9:␣
↪Reassign to most frequent value
missing_set_3 =␣
↪['EMP_10','EMP_35','EMP_45','EMPDVGOC','ENV_30','ENVDVCON','ENVDVTYP','ENVDVGRW','GEN_05','
for col in missing_set_3:
  frequent = df[col].mode()
  df[col].mask(df[col] == 9, frequent[0], inplace=True)
```

[ ]:
```python
#Change non-ordinal variables to categorical or else they will be considered␣
↪numeric
df = df.astype('category')
```

[ ]:
```python
#View basic summary of variables
df.info()
df.describe()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 17319 entries, 0 to 18138
Data columns (total 72 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   GEODVGPR  17319 non-null  category
 1   EMP_10    17319 non-null  category
 2   EMP_35    17319 non-null  category
 3   EMP_45    17319 non-null  category
 4   EMPDVGOC  17319 non-null  category
 5   EMPDVGYW  17319 non-null  category
 6   ENV_30    17319 non-null  category
 7   ENVDVCON  17319 non-null  category
 8   ENVDVTYP  17319 non-null  category
 9   ENVDVGRW  17319 non-null  category
 10  PPE_05    17319 non-null  category
 11  PPE_10    17319 non-null  category
```

```
12  PPE_15A   17319 non-null  category
13  PPE_15B   17319 non-null  category
14  PPE_15C   17319 non-null  category
15  PPE_15D   17319 non-null  category
16  PPE_15E   17319 non-null  category
17  PPE_15F   17319 non-null  category
18  PPE_15G   17319 non-null  category
19  PPE_15H   17319 non-null  category
20  PPE_15I   17319 non-null  category
21  PPE_15J   17319 non-null  category
22  PPE_20    17319 non-null  category
23  PPE_25    17319 non-null  category
24  PPE_30A   17319 non-null  category
25  PPE_30B   17319 non-null  category
26  PPE_30C   17319 non-null  category
27  PPE_30D   17319 non-null  category
28  PPE_30E   17319 non-null  category
29  PPE_30F   17319 non-null  category
30  PPE_30G   17319 non-null  category
31  PPE_30H   17319 non-null  category
32  PPE_35A   17319 non-null  category
33  PPE_35B   17319 non-null  category
34  PPE_35C   17319 non-null  category
35  PPE_35D   17319 non-null  category
36  PPE_35E   17319 non-null  category
37  PPE_35F   17319 non-null  category
38  PPE_35G   17319 non-null  category
39  PPE_35H   17319 non-null  category
40  PPE_40A   17319 non-null  category
41  PPE_40B   17319 non-null  category
42  PPE_40C   17319 non-null  category
43  PPE_40D   17319 non-null  category
44  PPE_40E   17319 non-null  category
45  PPE_40F   17319 non-null  category
46  PPE_40G   17319 non-null  category
47  PPE_40H   17319 non-null  category
48  PPE_40I   17319 non-null  category
49  PPE_45A   17319 non-null  category
50  PPE_45B   17319 non-null  category
51  PPE_45C   17319 non-null  category
52  PPE_45D   17319 non-null  category
53  PPE_45E   17319 non-null  category
54  PPE_45F   17319 non-null  category
55  PPE_45G   17319 non-null  category
56  PPE_45H   17319 non-null  category
57  PPE_45I   17319 non-null  category
58  PPE_50A   17319 non-null  category
59  PPE_50B   17319 non-null  category
```

```
 60   PPE_50C    17319 non-null   category
 61   PPE_50D    17319 non-null   category
 62   PPE_50E    17319 non-null   category
 63   PPE_50F    17319 non-null   category
 64   GEN_05     17319 non-null   category
 65   GEN_20     17319 non-null   category
 66   AGEDVG4    17319 non-null   category
 67   GDRDVGRP   17319 non-null   category
 68   ISDVFLAG   17319 non-null   category
 69   PGDVFLA    17319 non-null   category
 70   IMMDVGST   17319 non-null   category
 71   Worse_MH   17319 non-null   category
dtypes: category(72)
memory usage: 1.3 MB
```

[ ]:

```
        GEODVGPR EMP_10 EMP_35 EMP_45 EMPDVGOC EMPDVGYW ENV_30  \
count      17319  17319  17319  17319    17319    17319  17319
unique         7      2      2      3        9        4      2
top           30      2      1      1        5        1      2
freq        7898  14136  12883  12636     7491     5661  12222

        ENVDVCON ENVDVTYP ENVDVGRW  ... PPE_50E PPE_50F GEN_05 GEN_20  \
count      17319    17319    17319  ...   17319   17319  17319  17319
unique         3        7        8  ...       6       6      5      5
top            2        1       30  ...       4       2      2      4
freq        9812     7259     7879  ...    4061    5335   6917   7304

        AGEDVG4 GDRDVGRP ISDVFLAG PGDVFLA IMMDVGST Worse_MH
count     17319    17319    17319   17319    17319    17319
unique        4        2        2       2        2        2
top           1        2        2       2        1        1
freq       5047    15316    17065   15603    15476    11929

[4 rows x 72 columns]
```

The initial working dataset has 72 categorical variables with 17,319 observations.

2 3. EDA

2.1 3.1 Univariate Analysis

[ ]:

```python
profile = ProfileReport(df)
profile.to_file(output_file="2. Transformed Dataset Profile.html")
```

```
Summarize dataset:   0%|          | 0/5 [00:00<?, ?it/s]


Generate report structure:   0%|          | 0/1 [00:00<?, ?it/s]
```

```
Render HTML:   0%|          | 0/1 [00:00<?, ?it/s]
```

```
Export report to file:   0%|          | 0/1 [00:00<?, ?it/s]
```

The Dataset Profile confirms the initial working dataset has no missing or invalid values.

Of the 72 categorical variables, 5 are nominal, 39 are nominal and dichotomous (i.e. contain only 2 classes), and 28 are ordinal.

The distribution in the bar chart for each variable was examined. When a distribution is too skewed, such as when there is only one dominant bar and the other categories are present in very low numbers, this is often not helpful in machine learning. Several considerations for feature selection were identified. * ISDVFLAG is a variable with only 2 classes, and 98.5% of responses fall into 1 class. Hence, this variable could potentially be removed as it will not contribution information that will be useful for prediction.

2.2 3.2 Bivariate Analysis: Correlation Analysis

[ ]:
```
#Kendall rank correlation for all 28 ordinal variable pairs

import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

#Columns names of ordinal variables
ord_var =␣
 ↪['PPE_30A','PPE_30B','PPE_30C','PPE_30D','PPE_30E','PPE_30F','PPE_30G','PPE_30H','PPE_35A',
        ␣
 ↪'PPE_50A','PPE_50B','PPE_50C','PPE_50D','PPE_50E','PPE_50F','GEN_05','GEN_20','AGEDVG4']

#Changing ordinal variables to integer datatype so that kendal rank correlation␣
 ↪can be performed
df[ord_var] = df[ord_var].astype(int)

#Kendal rank correlation and generation of heatmap
plt.figure(figsize=(20,20))
corr_mat = df[ord_var].corr(method="kendall")
mask = np.zeros_like(corr_mat, dtype=np.bool)
mask[np.triu_indices_from(mask)] = True
sns.heatmap(corr_mat, annot=True, mask=mask)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:16:
DeprecationWarning: `np.bool` is a deprecated alias for the builtin `bool`. To
silence this warning, use `bool` by itself. Doing this will not modify any
behavior and is safe. If you specifically wanted the numpy scalar type, use
`np.bool_` here.
```