Python for Data Analysis and Visualisation Final Assignment

Introduction

This Jupyter Notebook contains the entirety of my analysis of `Scotland_teaching_file_1PCT.csv`. All Basic Requirements have been met, and all Additional Requirements have been met with the exception of using `nbconvert`.

An attempt was made to implement `nbconvert`, but there were issues with the installation of Jupyter that prevented nbconvert running. All code relating to this has subsequently been removed. Textual interpreation has not been provided on every plot as the data set provided used alphanumeric characters, and I believed that it would reduce the robustness and clarity of the analysis for minimal gain if this was provided on each plot.

No specific problems encountered worthy of note.

Reproducibility and Reusability

Reproducibility: This analysis is emminently reproducible. Given a dataframe of the same structure, all analysis can be redone with no change to the code framework.
Reusability: Owing to some of the data refinement steps, if the structure of the dataframe was changed, some of the code in `data_refinement.py` would have to be changed to cater for new columns, and the format these columns should be in. All subsequent analysis could then be done without changes to the code.

Provenance

All of this work is my own. Online documentation for each package was referred to when using packages.

Initialisation

Load in relevant packages and classes from other scripts

In [ ]:
```
import sys
sys.path.append('../code')
```

```
from data_refinement import DataLoader
from data_plotting import DataPlotter
from data_description import DataDescriber
```

Data Refinement

To run data refinement process from terminal, call data_refinement.py script with one argument as the location of the .csv containing the data

In [ ]:
```
dl = DataLoader('..\\data\\Scotland_teaching_file_1PCT.csv')
# Rename columns to a standard format
dl._rename_cols()
# Check data is in required format
dl.refine_data()
# Drop any duplioated rows
dl.drop_duplicates()
# Save as a refined dataset
dl.df.to_csv('..\\data\\Scotland_teaching_file_1PCT_refined.csv')
```

```
All data values in column 'Record_Number' match expected data type
All data values in column 'Region' match expected data type
All data values in column 'Residence_Type' match expected data type
All data values in column 'Family_Composition' match expected data type
All data values in column 'Sex' match expected data type
All data values in column 'Age' match expected data type
All data values in column 'Marital_Status' match expected data type
All data values in column 'Student' match expected data type
All data values in column 'Country_Of_Birth' match expected data type
All data values in column 'Health' match expected data type
All data values in column 'Ethnic_Group' match expected data type
All data values in column 'Religion' match expected data type
All data values in column 'Economic_Activity' match expected data type
All data values in column 'Occupation' match expected data type
All data values in column 'Industry' match expected data type
All data values in column 'Hours_Worked_Per_Week' match expected data type
All data values in column 'Approximate_Social_Grade' match expected data t
ype
No duplicated rows found
```

Basic Requirements

Descriptive Analysis

In [ ]:
```
dd = DataDescriber('..\\data\\Scotland_teaching_file_1PCT_refined.csv')
```

In [ ]:
```
dd.no_records()
```

```
The data set has 63388 rows
```

In [ ]:
```
dd.col_types()
```

```
Column Record_Number is of data type <class 'numpy.int64'>
Column Region is of data type <class 'str'>
Column Residence_Type is of data type <class 'str'>
Column Family_Composition is of data type <class 'str'>
Column Sex is of data type <class 'numpy.int64'>
Column Age is of data type <class 'numpy.int64'>
Column Marital_Status is of data type <class 'numpy.int64'>
Column Student is of data type <class 'numpy.int64'>
Column Country_Of_Birth is of data type <class 'numpy.int64'>
Column Health is of data type <class 'numpy.int64'>
Column Ethnic_Group is of data type <class 'numpy.int64'>
Column Religion is of data type <class 'numpy.int64'>
Column Economic_Activity is of data type <class 'str'>
Column Occupation is of data type <class 'str'>
Column Industry is of data type <class 'str'>
Column Hours_Worked_Per_Week is of data type <class 'str'>
Column Approximate_Social_Grade is of data type <class 'str'>
```

In all following analysis, please refer to these lists for textual interpretations of plots. Due to data being provided in alphanumeric form, it would be neither efficient nor robust to swap out alphanumerical values for textual interpretations at every step of the data analysis process

In [ ]:
```
dd.unique_values()
```

```
Column Residence_Type takes values ['C (Resident in a Communal Establishme
nt)', 'P (Not resident in a Communal Establishment)']
Column Family_Composition takes values ['0 (Not in a family)', '1 (Marrie
d/same-sex civil partnership couple family)', '2 (Cohabiting couple famil
y)', '3 (Lone parent family (male head))', '4 (Lone parent family (female
lead))', '5 (Other related family)', 'X (No code required (residents of a
communal establishment))']
Column Sex takes values ['1 (Male)', '2 (Female)']
Column Age takes values ['1 (0 to 15)', '2 (16 to 24)', '3 (25 to 34)', '4
(35 to 44)', '5 (45 to 54)', '6 (55 to 64)', '7 (65 to 74)', '8 (75 and ov
er)']
Column Marital_Status takes values ['1 (Single (Never married or never reg
istered a same-sex civil partnership)', '2 (Married or in a same sex-civil
partnership)', '3 (Separated, but still legally married or still legally i
n a same-sex civil partnership)', '4 (Divorced or formerly in a same-sex c
ivil partnership which is now legally dissolved)', '5 (Widowed or survivin
g partner from a same-sex civil partnership)']
Column Student takes values ['1 (Yes)', '2 (No)']
Column Country_Of_Birth takes values ['1 (UK)', '2 (Non UK)']
Column Health takes values ['1 (Very good health)', '2 (Good health)', '3
(Fair health)', '4 (Bad health)', '5 (Very bad health)']
Column Ethnic_Group takes values ['1 (White)', '2 (Mixed or multiple ethni
c group)', '3 (Asian)', '4 (African)', '5 (Caribbean or black)', '6 (Other
ethnic group)']
Column Religion takes values ['1 (No religion)', '2 (Christian)', '3 (Budd
hist)', '4 (Hindu)', '5 (Jewish)', '6 (Muslim)', '7 (Sikh)', '8 (Other rel
igion)', '9 (Not stated)']
Column Economic_Activity takes values ['1 (Economically active: Employe
d)', '2 (Economically active: Self-Employed)', '3 (Economically active: Un
employed)', '4 (Economically active: Full-time student)', '5 (Economically
inactive: Retired)', '6 (Economically inactive: Student)', '7 (Economicall
y inactive: Looking after home or family)', '8 (Economically inactive: Lon
g-term sick or disabled)', '9 (Economically inactive: Other)', 'X (No code
required (Aged under 16))']
Column Occupation takes values ['1 (Managers, Directors and Senior Officia
ls)', '2 (Professional Occupations)', '3 (Associate Professional and Techn
ical Occupations)', '4 (Administrative and Secretarial Occupations)', '5
(Skilled Trades Occupations)', '6 (Caring, Leisure and Other Service Occup
ations)', '7 (Sales and Customer Service Occupations)', '8 (Process, Plant
and Machine Operatives)', '9 (Elementary Occupations)', 'X (No code requir
ed (People aged under 16 and people who have never worked))']
Column Industry takes values ['1 (Agriculture, forestry and fishing)', '10
(Education)', '11 (Human health and social work activities)', '12 (Arts; e
ntertainment and recreation)', '13 (Other)', '2 (Mining and quarrying; Man
ufacturing; Electricity, gas, steam and air conditioning system; Water sup
ply)', '3 (Construction)', '4 (Wholesale and retail trade; Repair of motor
vehicles and motorcycles)', '5 (Accommodation and food service activitie
s)', '6 (Transport and storage; Information and communication)', '7 (Finan
cial and insurance activities)', '8 (Real estate activities; Professional
scientific and technical activities; Administrative and support service ac
tivities)', '9 (Public administration and defence)', 'X (No code required
(People aged under 16 and people who have never worked))']
Column Hours_Worked_Per_Week takes values ['1 (Part-time: 15 or less hours
worked)', '2 (Part-time: 16 to 30 hours worked)', '3 (Full-time: 31 to 48
hours worked)', '4 (Full-time 49 or more hours worked)', 'X (No code requi
red (People aged under 16 and people not working))']
```

```
Column Approximate_Social_Grade takes values ['1 (AB)', '2 (C1)', '3 (C
2)', '4 (DE)', 'X (No code required ( People aged under 16 and people resi
dent in communal establishments))']
```

To see the number of occurrences of each value for each column, please select column as a Factor in the below widget and select the column in the 'Summary stats' dropdown

In [ ]:
```
dd.group_data()
```

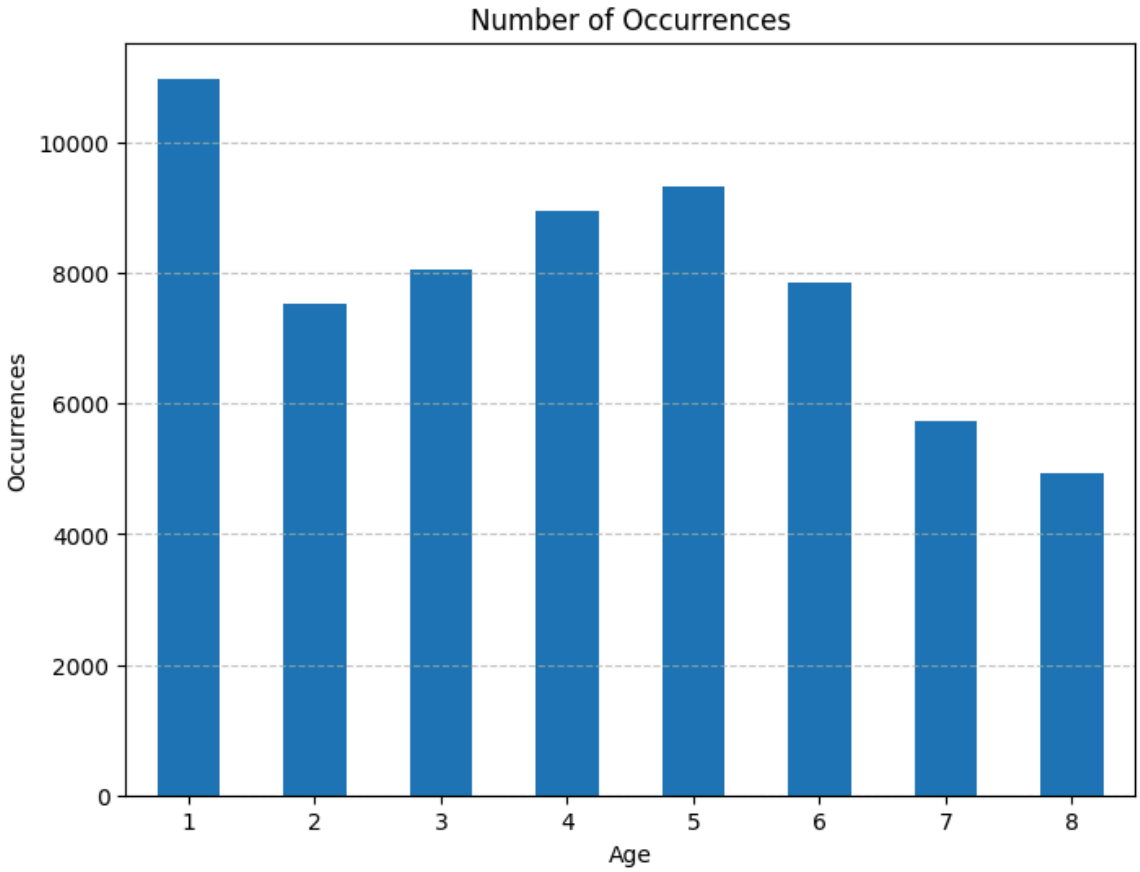
```
d:\University\Python for Data Analysis\Repos\PFDAAV\notebooks\../code\data
_description.py:64: FutureWarning: elementwise comparison failed; returnin
g scalar instead, but in the future will perform elementwise comparison
  if 'X' in list:
interactive(children=(Dropdown(description='Factor: ', index=4, options=
('Region', 'Residence_Type', 'Family_C…
<Figure size 1400x1000 with 0 Axes>
```

Plots

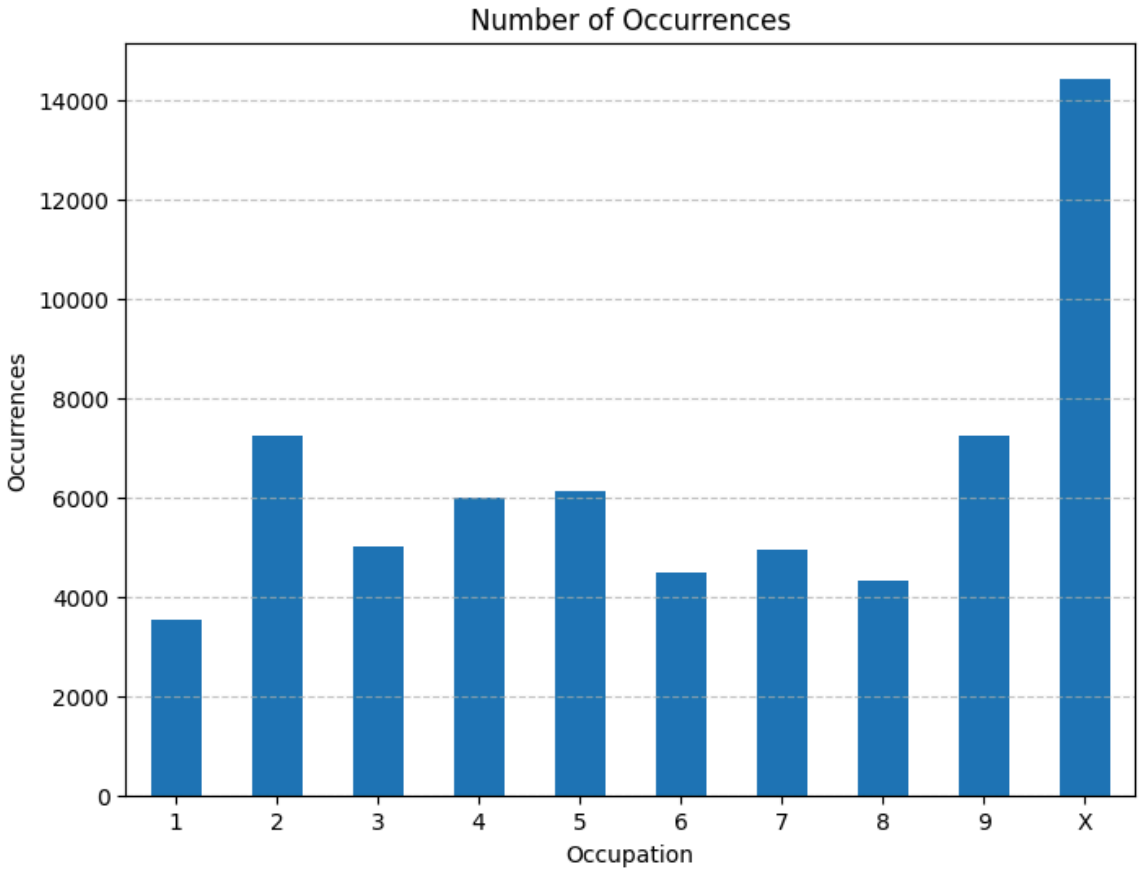
N.B. private methods used as interactive bar chart method `bar_chart()` intended for actual use. These plots are shown to demonstrate Basic Requirements.

In [ ]:
```
# Load in instance of DataPlotter class
dp = DataPlotter('..\\data\\Scotland_teaching_file_1PCT_refined.csv')
```

In [ ]:
```
dp._plot_bar_chart('Age')
```



In [ ]:
```
dp._plot_bar_chart('Occupation')
```

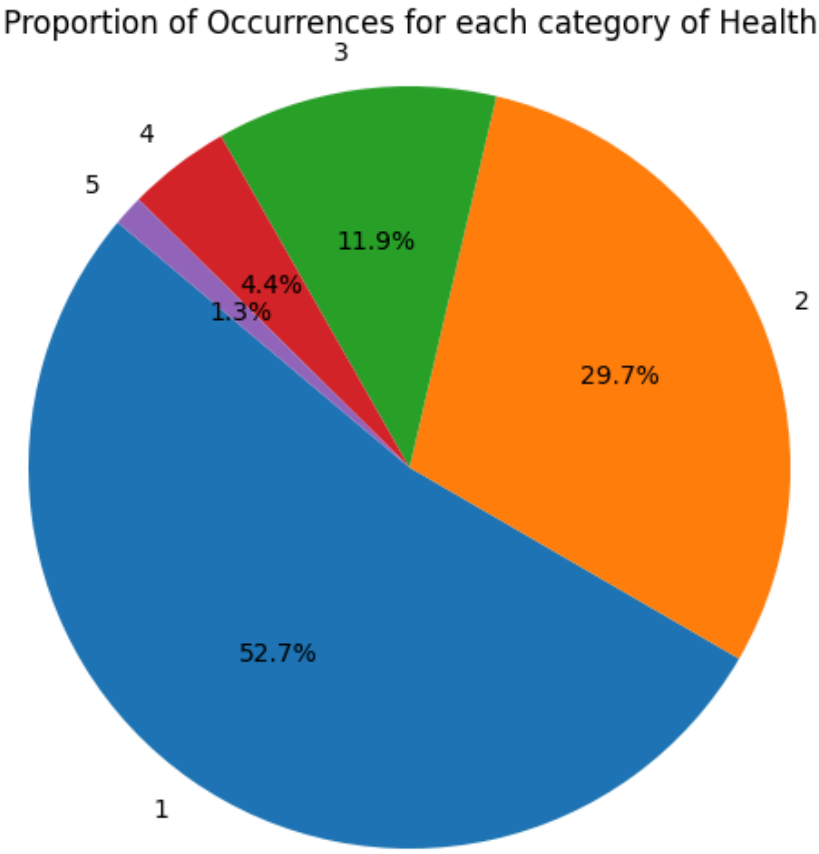


Additional Requirements

Pie Charts

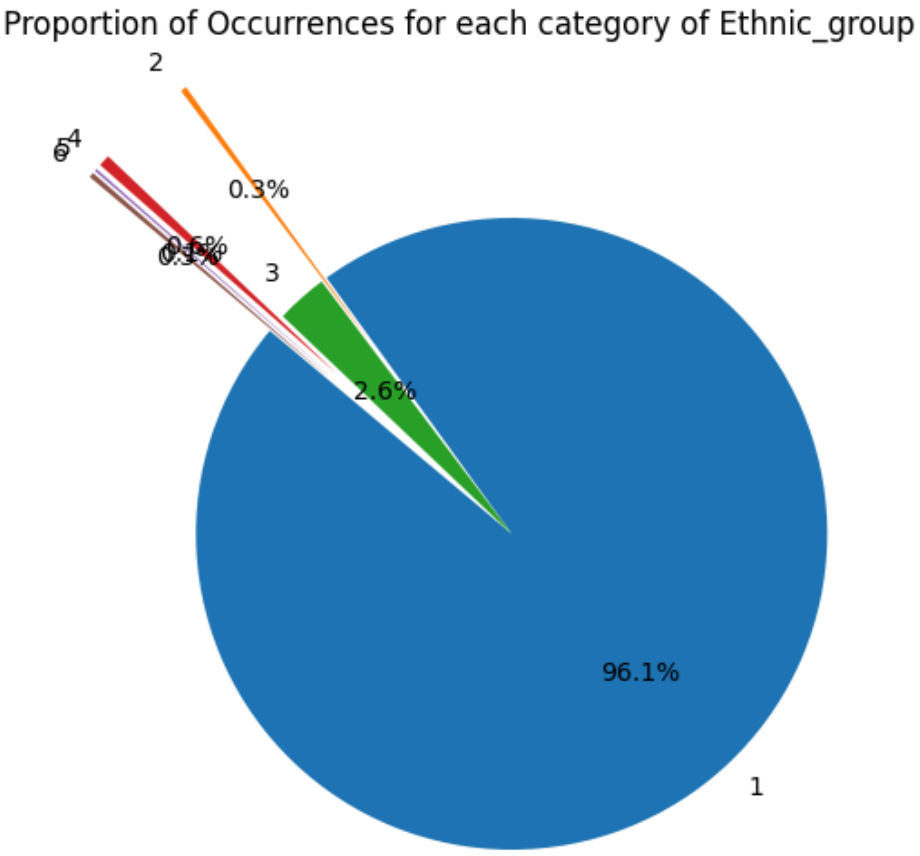
Use private method to show pie chart as specified in Additional Requirements. Intended functionality is for use of ipywidget, hence definition as private method Parameters can be set to choose whether smaller values are exploded from the main pie chart to aid clarity

In [ ]:
```
dp._plot_pie_chart('Health', 0.25, 0.01)
```



Pie chart is a poor choice of visual for this data owing to the dominance of category 1, as such the plot is not very clear

In [ ]:
```
dp._plot_pie_chart('Ethnic_Group', 0.75, 0.01)
```



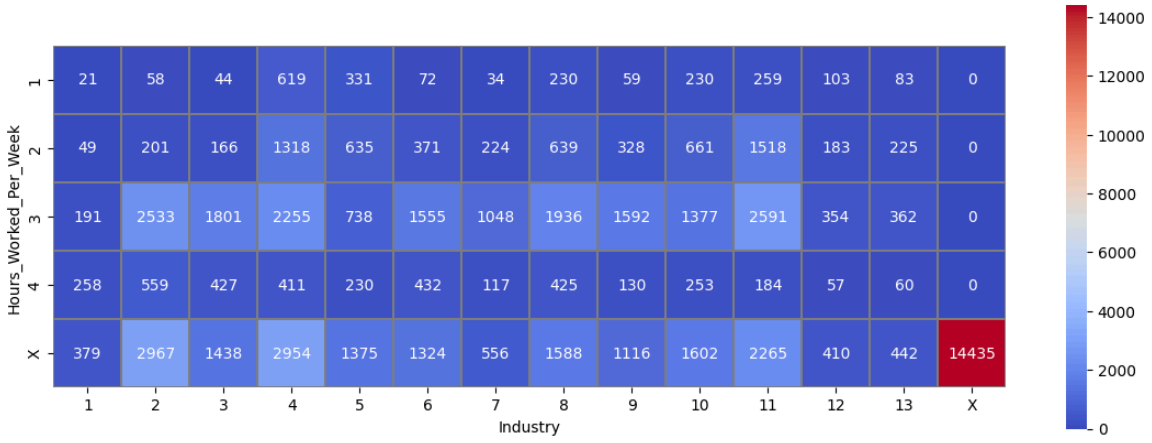
In the below widget, choose the factor you wish to plot along with how much you want small values exploded from the main plot, and the maximum value to explode

In [ ]:
```
dp.pie_chart()
```

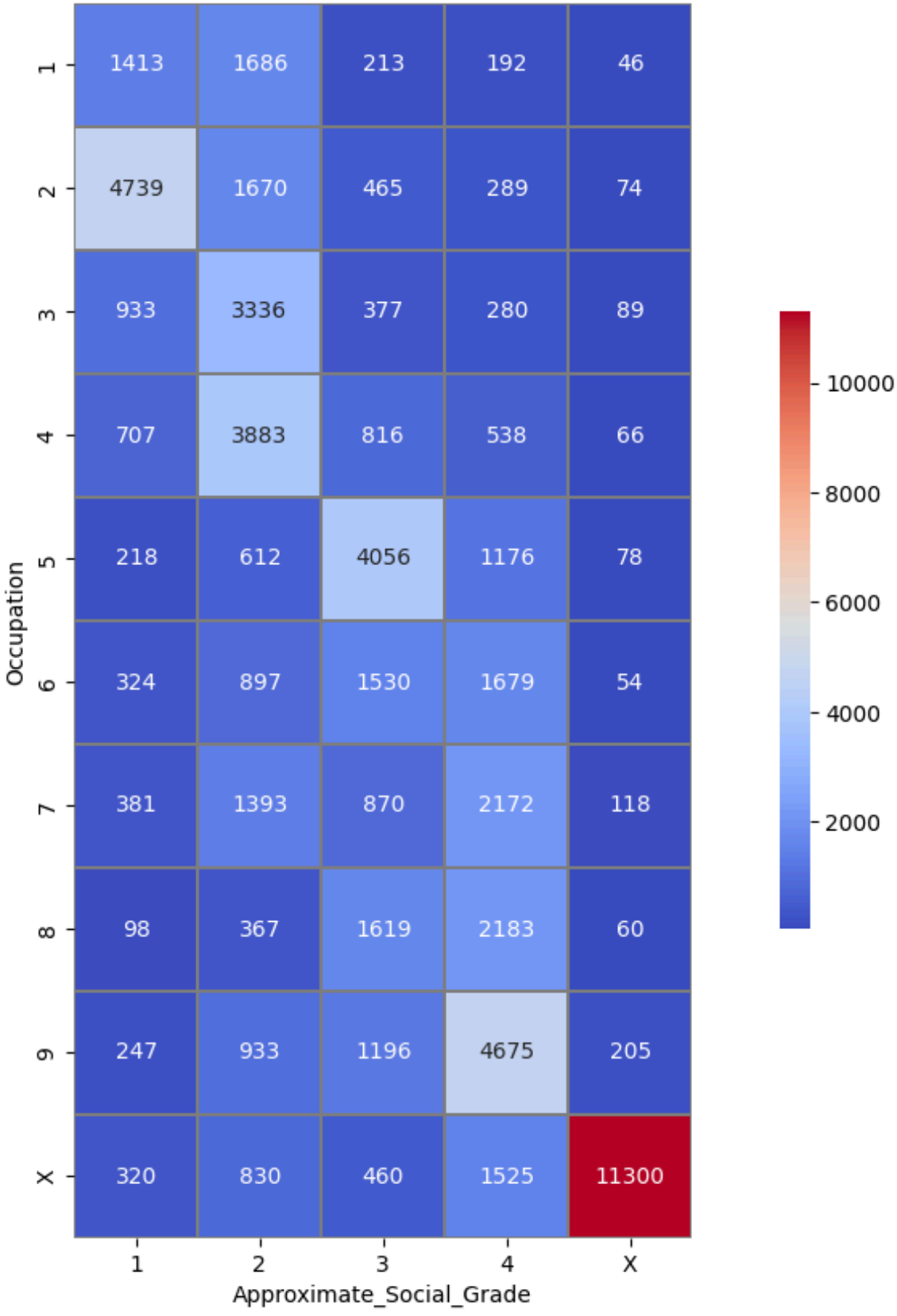
```
interactive(children=(Dropdown(description='Factor: ', index=4, options=
('Region', 'Residence_Type', 'Family_C…
```

Data Description

In [ ]:
```
dd._grouped_no_records('Hours_Worked_Per_Week', 'Industry')
```

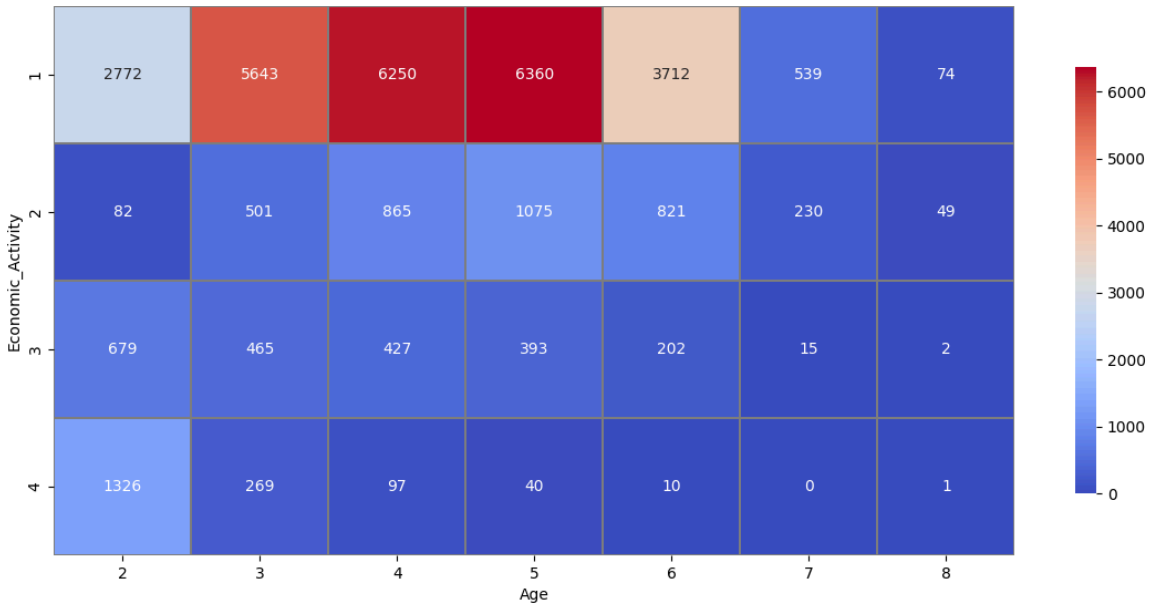


In [ ]:
```
dd._grouped_no_records('Occupation', 'Approximate_Social_Grade')
```



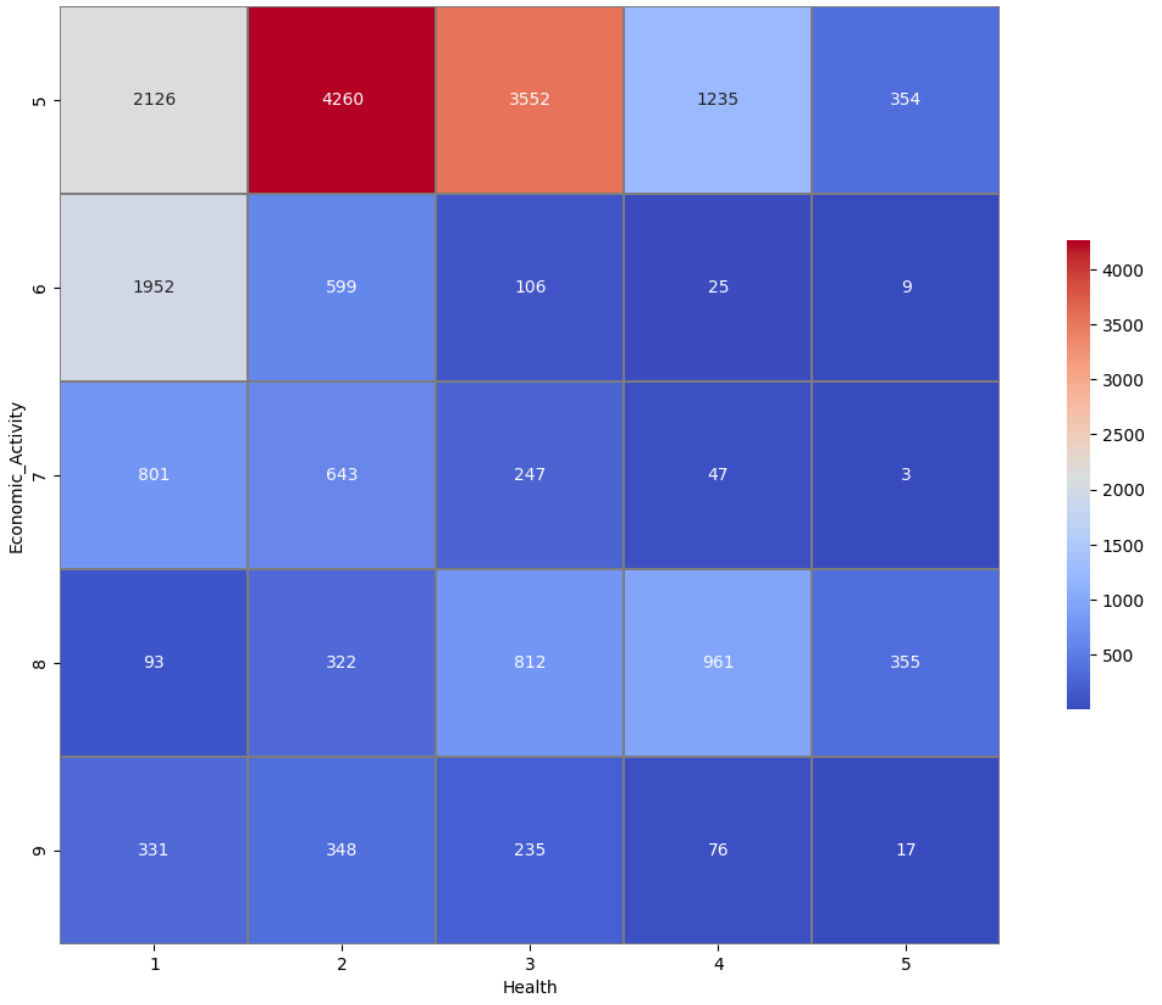
Below, we have subsetted to economocially active people (categories 1, 2, 3, and 4)

In [ ]:
```
dd._grouped_no_records(col1 = 'Economic_Activity', col2 = 'Age', col1_vals
```



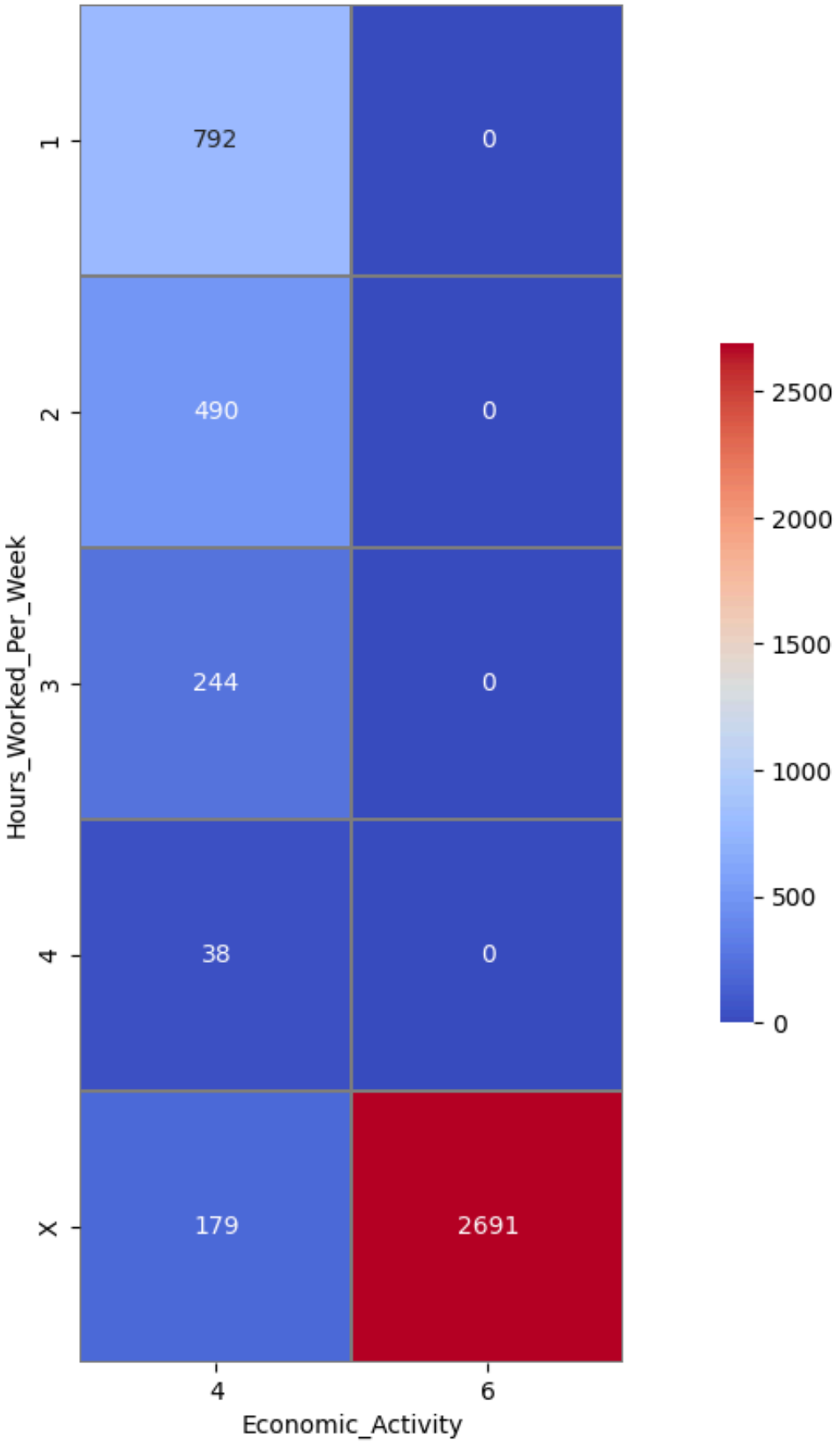
Below, we have subsetted to economocially inactive people (categories 5, 6, 7, 8, and 9)

In [ ]:
```
dd._grouped_no_records(col1 = 'Economic_Activity', col2 = 'Health', col1_v
```



Below, we have subsetted to students (categories 4 and 6)

In [ ]:
```
dd._grouped_no_records(col1 = 'Hours_Worked_Per_Week', col2 = 'Economic_Ac
```



In the below widget, select the two variables to compare using the 'Factor' dropdown boxes.

To select which values to subset by, select in the relevant box. Use ctrl + left click to select multiple.

To consider one factor in isolation, select it in the 'Summary stats' dropdown.

To switch below absolute count and proportion, select in 'Factor' dropdown menu.

In [ ]:
```
dd.group_data()
```

```
d:\University\Python for Data Analysis\Repos\PFDAAV\notebooks\../code\data
_description.py:64: FutureWarning: elementwise comparison failed; returnin
g scalar instead, but in the future will perform elementwise comparison
  if 'X' in list:
interactive(children=(Dropdown(description='Factor: ', index=4, options=
('Region', 'Residence_Type', 'Family_C…
```