assignment2

January 4, 2021

1 Assignment 2

For this assignment you'll be looking at 2017 data on immunizations from the CDC. Your datafile for this assignment is in `assets/NISPUF17.csv`. A data users guide for this, which you'll need to map the variables in the data to the questions being asked, is available at `assets/NIS-PUF17-DUG.pdf`. **Note: you may have to go to your Jupyter tree (click on the Coursera image) and navigate to the assignment 2 assets folder to see this PDF file).**

1.1 Question 1

Write a function called `proportion_of_education` which returns the proportion of children in the dataset who had a mother with the education levels equal to less than high school (<12), high school (12), more than high school but not a college graduate (>12) and college degree.

This function should return a dictionary in the form of (use the correct numbers, do not round numbers):

```
{"less than high school":0.2,
"high school":0.4,
"more than high school but not college":0.2,
"college":0.2}
```

page 145/202: EDUC1: EDUCATION OF MOTHER CATEGORIES (RECODE)

[4]:
```python
import pandas as pd
nis = pd.read_csv("assets/NISPUF17.csv")
def proportion_of_education():
    ans = {}

    count_less_than_high_school = 0
    count_high_school = 0
    count_more_than_high_school = 0
    count_college = 0

    education = nis['EDUC1'].tolist()
    total = len(education)

    for each_education in education:
        if each_education ==1 :
            count_less_than_high_school+=1
        elif each_education == 2:
            count_high_school+=1
        elif each_education == 3:
            count_more_than_high_school+=1
        elif each_education == 4:
            count_college+=1

    ans["less than high school"] = count_less_than_high_school/total
    ans["high school"] = count_high_school/total
    ans["more than high school but not college"] = count_more_than_high_school/
 ↪total
    ans["college"] = count_college/total

    return ans
    raise NotImplementedError()

proportion_of_education()
```

[4]:
```
{'less than high school': 0.10202002459160373,
 'high school': 0.172352011241876,
 'more than high school but not college': 0.24588090637625154,
 'college': 0.47974705779026877}
```

[17]:
```
assert type(proportion_of_education())==type({}), "You must return a dictionary.
 ↪"
assert len(proportion_of_education()) == 4, "You have not returned a dictionary␣
 ↪with four items in it."
assert "less than high school" in proportion_of_education().keys(), "You have␣
 ↪not returned a dictionary with the correct keys."
assert "high school" in proportion_of_education().keys(), "You have not␣
 ↪returned a dictionary with the correct keys."
assert "more than high school but not college" in proportion_of_education().
 ↪keys(), "You have not returned a dictionary with the correct keys."
assert "college" in proportion_of_education().keys(), "You have not returned a␣
 ↪dictionary with the correct keys."
```

1.2 Question 2

Let's explore the relationship between being fed breastmilk as a child and getting a seasonal influenza vaccine from a healthcare provider. Return a tuple of the average number of influenza vaccines for those children we know received breastmilk as a child and those who know did not.

This function should return a tuple in the form (use the correct numbers:

```
(2.5, 0.1)
```

CBF_01 – child ever fed breast milk 1.Yes 2.No 3.Don't know 99.Missing
P_NUMFLU – total number of seasonal influenza doses

[37]:
```python
import numpy as np
def average_influenza_doses():
    count_breastmilk = nis[nis['CBF_01'] == 1].shape[0]
    count_not_breastmilk = nis[nis['CBF_01'] == 2].shape[0]

    total_number_doses = np.sum(nis["P_NUMFLU"].fillna(0))

    return (count_breastmilk/total_number_doses, count_not_breastmilk/
 ↪total_number_doses)

    raise NotImplementedError()

average_influenza_doses()
```

[37]:
```
(0.8592527005489641, 0.14467859040198336)
```

[38]:
```python
assert len(average_influenza_doses())==2, "Return two values in a tuple, the␣
 ↪first for yes and the second for no."
```

1.3 Question 3

It would be interesting to see if there is any evidence of a link between vaccine effectiveness and sex of the child. Calculate the ratio of the number of children who contracted chickenpox but were vaccinated against it (at least one varicella dose) versus those who were vaccinated but did not contract chicken pox. Return results by sex.

This function should return a dictionary in the form of (use the correct numbers):

```
{"male":0.2,
"female":0.4}
```

Note: To aid in verification, the `chickenpox_by_sex()['female']` value the autograder is looking for starts with the digits `0.0077`.

P_NUMVRC – total number of varicella doses 0, 1, 2, NaN…
HAD_CPOX – did child ever have chicken pox 1.yes 2.no
SEX 1.male 2.female

[40]:
```python
def chickenpox_by_sex():
    ans = {}

    nis["P_NUMFLU"] = nis["P_NUMFLU"].fillna(0)

    # condition1: male, contracted chickenpox, vaccinated (at least one␣
 ↪varicella)
    condition1 = (nis["SEX"] == 1) & (nis["P_NUMVRC"] > 0) & (nis["HAD_CPOX"]␣
 ↪== 1)
    count1 = nis[condition1].shape[0]

    # condtion2: male, vaccinated but did not contract chicken pox
    condition2 = (nis["SEX"] == 1) & (nis["P_NUMVRC"] > 0) & (nis["HAD_CPOX"]␣
 ↪== 2)
```

```
    count2 = nis[condition2].shape[0]

    rapport_male = count1/count2

    # condition3: female, contracted chickenpox, vaccinated (at least one␣
↪varicella)
    condition3 = (nis["SEX"] == 2) & (nis["P_NUMVRC"] > 0) & (nis["HAD_CPOX"]␣
↪== 1)
    count3 = nis[condition3].shape[0]

    # condtion4: female, vaccinated but did not contract chicken pox
    condition4 = (nis["SEX"] == 2) & (nis["P_NUMVRC"] > 0) & (nis["HAD_CPOX"]␣
↪== 2)
    count4 = nis[condition4].shape[0]

    rapport_female = count3/count4

    ans["male"] = rapport_male
    ans["female"] = rapport_female

    return ans

    raise NotImplementedError()


chickenpox_by_sex()
```

[40]:
```
{'male': 0.009675583380762664, 'female': 0.0077918259335489565}
```

[41]:
```
assert len(chickenpox_by_sex())==2, "Return a dictionary with two items, the␣
↪first for males and the second for females."
```

1.4 Question 4

A correlation is a statistical relationship between two variables. If we wanted to know if vaccines work, we might look at the correlation between the use of the vaccine and whether it results in prevention of the infection or disease [1]. In this question, you are to see if there is a correlation between having had the chicken pox and the number of chickenpox vaccine doses given (varicella).

Some notes on interpreting the answer. If the `had_chickenpox_column` is either `1` (for yes) or `2` for no, and that the `num_chickenpox_vaccine_column` is the number of doses a child has been given of the varicella vaccine, then a positive correlation (e.g. `corr > 0`) would mean that an increase in `had_chickenpox_column` (which means more no's) would mean an increase in the `num_chickenpox_vaccine_column` (which means more doses of vaccine). If `corr < 0` then there is a negative correlation, indicating that having had chickenpox is related to an increase in the number of vaccine doses. Also, `pval` refers to the probability the relationship observed is significant. In this case `pval` should be very very small (will end in `e-18` indicating a very small number), which means the result unlikely to be by chance.

[1] This isn't really the full picture, since we are not looking at when the dose was given. It's

possible that children had chickenpox and then their parents went to get them the vaccine. Does this dataset have the data we would need to investigate the timing of the dose?

HAD_CPOX – did child ever have chicken pox 1.yes 2.no P_NUMVRC – total number of varicella doses: 0, 1, 2, NaN…

[15]:
```
import scipy.stats as stats
import numpy as np
import pandas as pd
nis = pd.read_csv("assets/NISPUF17.csv")

def corr_chickenpox():

    # this is just an example dataframe
    df=pd.DataFrame({"had_chickenpox_column":nis["HAD_CPOX"].tolist(),
                    "num_chickenpox_vaccine_column":nis["P_NUMVRC"].fillna(0).
 ↪tolist()})

    # here is some stub code to actually run the correlation
    corr, pval=stats.
 ↪pearsonr(df["had_chickenpox_column"],df["num_chickenpox_vaccine_column"])

    # just return the correlation
    return corr

    raise NotImplementedError()
corr_chickenpox()
```

[15]:
```
-0.01026880593978898
```

[16]:
```
assert -1<=corr_chickenpox()<=1, "You must return a float number between -1.0␣
 ↪and 1.0."
```

[ ]:
```

```