LAB2-Release

September 29, 2022

0.0.1 Objectives:

- To compute the relative frequency of two events occurring at the same time.
- To compute the relative frequency of one event occurs given that another event has also occurred.
- To be familiar with properties of the relative frequency.

The file “rainfall.xlsx”, which is extracted from The Hong Kong Observatory https://www.hko.gov.hk/en/wxinfo/pastwx/mws/mws.htm, stores the rainfall in Hong Kong within a certain period. Each element in the column “Rainfall” in the file “rainfall.xlsx” indicates the rainfall on a day. A positive value indicates it rained on that day, while a 0 indicates it did not rain on that day. One may wonder if season information indicates whether a day is rainy or not. For this, the column “Seasons” stores the season labels.

Download the file “rainfall.xlsx” from the course website to complete the lab.

We recommend you use the pandas data analysis package in python to do the homework. In particular, you should find the groupby and count functions provided by pandas useful. More information about the pandas data analysis package can be found from here:

https://pandas.pydata.org/pandas-docs/stable/index.html

However, you are free to do the question using any methods or functions that you like.

After you have completed the notebok, export it as pdf for submission. You can do this in one of two ways: 1. Go to File, click Download as, click PDF via LaTeX (.pdf). 2. Go to File, click Download as, click HTML (.html), then convert the html file to pdf file.

We will first load the datafile and look at the first few lines. Do not change the code below.

[1]:
```python
# load pandas, a data analysis package
import pandas as pd

# read the data file and show the first few entries
data = pd.read_excel('rainfall.xlsx',index_col='Index')
data.head()
```

[1]:
```
             Date Season  Rainfall
Index
1      2021-09-01   fall      5.90
2      2021-09-02   fall      0.00
3      2021-09-03   fall      0.05
```

```
4      2021-09-04    fall      0.90
5      2021-09-05    fall      0.05
```

Part a: From the data, find the relative frequency that a day is rainy, and the relative frequency of its complement (i.e. a day is not rainy). In other words, count the number of days whose rainfall is greater than 0 (or less than or equal to 0) and divide by the total number of days in the dataset.

[64]:
```
# put your code here

# add a new column called "Rainy" which is True if the Rainfall value is␣
 ↪positive and False otherwise
data['Rainy'] = (data['Rainfall'] > 0)

# count how many days in the year it rained and how many days it did not
# we use the groupby command to separates the data by whether it rained
# ['Rainy'] selects only the 'Rainy' column
# "count()" counts how many cells in the column have valid entries
n_rain = data.groupby('Rainy')['Rainy'].count()

n = sum(n_rain)
p_rain = n_rain / n

# print(f'The number of the days whose rainfall is greater than 0 (or less than␣
 ↪or equal to 0) is\n')
# print(n_rain)

# print(f'\nThe total number of days is {n}')

# print(f'\nThe relative frequency that a day is rainy, and the relative␣
 ↪frequency of its complement are\n')
# print(p_rain)

print(f"Relative frequency that a day is not rainy: " + str(p_rain.iloc[0]) +␣
 ↪"\n")

print("Relative frequency that a day is rainy: " + str(p_rain.iloc[1]))
```

```
Relative frequency that a day is not rainy: 0.4684931506849315

Relative frequency that a day is rainy: 0.5315068493150685
```

Part b: Find the relative frequencies that a day is a spring day, a summer day, a fall day and a winter day, respectively. In other words, count the number of days in each season and divide by the total number of days in the dataset.

[57]:
```
# put your code here
# calculate the number of days in each season
n_season = data.groupby('Season')['Season'].count()
```

```
# calculate the relative frequencies that a day is spring, summer, fall and␣
 ↪windter
p_season = n_season / n

# print(f'The number of days in each season is\n')
# print(n_season)

print(f'\nThe relative frequencies that a day is a spring day, a summer day, a␣
 ↪fall day and a winter day are\n')
print(p_season)
```

```
The relative frequencies that a day is a spring day, a summer day, a fall day
and a winter day are

Season
fall      0.252055
spring    0.246575
summer    0.257534
winter    0.243836
Name: Season, dtype: float64
```

Part c: Find the The relative frequencies that a day a spring rainy day, a summer rainy day, a fall rainy day and a winter rainy day, respectively.

[60]:
```
# put your code here

# group by season and sum the boolean entries of "Rainy"
n_rainy_season = data.groupby('Season')['Rainy'].sum()
p_rainy_season = n_rainy_season/n

# print(f'\nThe number of days that a spring rainy day, a summer rainy day, a␣
 ↪fall rainy day, and a winter rainy\n')
# print(n_rainy_season)

print(f'\nThe relative frequencies that a day is a spring rainy day, a summer␣
 ↪rainy day, a fall rainy day and a winter rainy day are\n')
print(p_rainy_season)
```

```
The number of days that a spring rainy day, a summer rainy day, a fall rainy
day, and a winter rainy

Season
fall      56
spring    38
summer    66
```

```
winter    34
Name: Rainy, dtype: int64

The relative frequencies that a day is a spring rainy day, a summer rainy day, a
fall rainy day and a winter rainy day are

Season
fall      0.153425
spring    0.104110
summer    0.180822
winter    0.093151
Name: Rainy, dtype: float64
```

Part d: Fine the following values

The relative frequency that a day is rainy given that the day is a spring day

The relative frequency that a day is rainy given that the day is a summer day

The relative frequency that a day is rainy given that the day is a fall day

The relative frequency that a day is rainy given that the day is a winter day

[18]:
```python
#put your code here
p_rainy_givenseason = n_rainy_season/n_season
print(f'\nThe relative frequencies given that a day is rainy are\n')
print(p_rainy_givenseason)
```

```
The relative frequencies given that a day is rainy are

Season
fall      0.608696
spring    0.422222
summer    0.702128
winter    0.382022
dtype: float64
```