Session 3. Exploratory data analysis I: Descriptive statistics

2026-02-12

Highlights:

This is my mini-reflection. Paragraphs must be indented.

It can contain multiple paragraphs.

Write the concepts that in your opinion are threshold concepts in this exercise. A threshold concept is a key idea that once you grasp it, it changes your understanding of a topic, phenomenon, subject, method, etc. Write between three and five threshold concepts that apply to your learning experience working on this exercise.

> "We never look beyond our assumptions and what's worse, we have given up trying to meet others; we just meet ourselves."
>
> —– Muriel Barbery

Session outline

- What is EDA?
- Data summaries revisited
- Appropriate summary statistics by scale of measurement
- Properties of data: central tendency and spread
- Univariate description
- Bivariate description
- Multivariate description
- `Pandas`

Reminder

NOTE: This is an Quarto Markdown document. This type of document is a plain text file that can recognize chunks of code. When you execute code within the document, the results are displayed beneath. Quarto Markdown files are *computational notebooks* which implement a coding philosophy called *literate programming*. Literate computing emphasizes the use of natural language to communicate with humans and chunks of code to communicate with the computer. By making the main audience other humans, this style of coding flips around the usual way in which code is written (computer is main audience, humans come second). This helps to make learning how to code more intuitive and accessible.

Preliminaries

Load packages. Remember, packages are units of shareable code that augment the functionality of base `Python`. For this session, the following package/s is/are used:

```
import pandas as pd
import numpy as np
from scipy.stats import chisquare, chi2_contingency
```

```
# Set display options to show all rows and columns
pd.set_option('display.max_rows', None)    # Show all rows
pd.set_option('display.max_columns', None) # Show all columns
pd.set_option('display.width', None)       # Auto-detect width
pd.set_option('display.max_colwidth', None) # Show full column content
```

We also will utilize some data from the `edashop` R package. To convert these R data files to python files, we will use the `reticulate` package:

```
library(edashop)  # A Package for a Workshop on Exploratory Data Analysis
library(reticulate)
```

From `edashop`, we will also load the following data frames for this session:

```
data("cntr_sp_basico")
data("cntr_sp_head")
data("cntr_sp_ipvs")
```

These data frames contain information about census tracts in the state of São Paulo. You can check the documentation in the usual way:

```
?cntr_sp_basico
```

To be able to convert these datasets in Python, we need to convert them into structures that python recognizes. Following chunk transform them into a Pandas DataFrame:

```
cntr_sp_basico = r.cntr_sp_basico
cntr_sp_head = r.cntr_sp_head
cntr_sp_ipvs = r.cntr_sp_ipvs
```

What is EDA?

Exploratory Data Analysis is the process of learning from the data by concentrating on its intrinsic characteristics and attributes. John W. Tukey, the statistician most responsible for clarifying the distinction between exploratory and confirmatory data analysis, likened exploratory data analysis to detective work. Exploratory data analysis is useful to discover essential evidence regarding the phenomenon of interest, similar to checking fingerprints and alibis that can be used in a trial - the equivalent of confirmatory data analysis, where hypotheses are tested and the evidence is evaluated.

To effectively deploy EDA, it is important to approach the data with as few assumptions as possible. By allowing the data to speak for themselves, EDA aims to:

1. *Simplify* descriptions to make them easier to handle with available cognitive power; and
2. Look *below* previously described surfaces to make the description more effective.

The main tools of EDA are descriptive statistics and visualization techniques. The focus in this session is on descriptive statistics, with an emphasis on *appropriate* descriptors for different types of data.

Before proceeding, it is worthwhile to briefly think about the things that we are first interested in when we begin working with a data set. What are the most important characteristics of the data that you care about?

-
-
-

Data descriptions revisited

In the previous session we used the method `describe()` from Pandas to obtain quick statistic descriptions of data. These descriptions already provided some key information about the data, including some summary statistics. For example, in our table with information about the São Paulo Social Vulnerability Index:

```
cntr_sp_ipvs.describe()
```

```
               v11          v12           v13          v16           v19  \
count  66096.000000  66096.000000  6.609600e+04  64481.000000  64481.000000
mean     197.489303    194.068522 -4.480422e+07      7.639153    928.980186
std      111.878834    101.801226  3.069370e+08      2.721422    912.954666
min        1.000000      0.000000 -2.147484e+09      0.000000      6.845938
25%      131.000000    129.000000  0.000000e+00      5.769231    458.022495
50%      196.000000    194.000000  0.000000e+00      7.558140    626.276265
75%      259.000000    257.000000  0.000000e+00      9.342835    999.468193
max     4074.000000    954.000000  1.660000e+02     56.250000  26088.791950

               v20          v21           v22          v23           v24  \
count  64491.000000  64491.000000  64491.000000  64491.000000  64491.000000
mean       4.315545      0.620825     14.678787     55.461301     24.923542
std        5.977543      1.521744     11.500893     17.323476     23.361612
min        0.000000      0.000000      0.000000      0.000000      0.000000
25%        1.000000      0.000000      5.384615     48.065407      7.519844
50%        2.304147      0.000000     12.448133     60.073260     16.120219
75%        5.035971      0.724638     21.568627     67.521368     35.542169
max      100.000000    100.000000    100.000000    100.000000    100.000000

               v25          v26           v27          v28           v29  \
count  64491.000000  64491.000000  64534.000000  64481.000000  64534.000000
mean      15.299612      3.471985     47.190961   1648.450535     94.712482
std       12.216109      4.431347      5.128672   1715.200201      5.327575
min        0.000000      0.000000     12.000000      0.000000      0.000000
25%        5.533597      0.561798     43.608442    829.866667     92.307692
50%       12.765957      2.027027     47.026536   1128.031915     96.124031
75%       22.341384      4.733728     50.675902   1741.792929     98.644068
max      100.000000    100.000000     81.000000  73312.500000    100.000000

               v30          v40           v41          v42          v43
count  64534.000000  64481.000000  64481.000000  64481.00000  64481.000000
mean      13.624916     89.748210     87.988724     95.53692     99.790313
std        7.573649     27.852032     27.109637     17.13362      2.193945
min        0.000000      0.000000      0.000000      0.00000      0.000000
25%        8.411215     99.270073     96.341463    100.00000    100.000000
50%       12.500000    100.000000    100.000000    100.00000    100.000000
75%       17.424242    100.000000    100.000000    100.00000    100.000000
max      100.000000    100.000000    100.000000    100.00000    100.000000
```

We can see that the method `description()`, when set the `include` parameter equals to `all`, understands what type of data it is dealing with, and provides summaries that are different for categorical and quantitative variables.

```
cntr_sp_ipvs.describe(include = 'all')
```

	COD_SETOR	AGSN	IPVS
count	66096	66096	66096
unique	66096	2	7
top	350010505000001	Não especial	Vulnerabilidade muito baixa
freq	1	61977	25365
mean	NaN	NaN	NaN
std	NaN	NaN	NaN
min	NaN	NaN	NaN
25%	NaN	NaN	NaN
50%	NaN	NaN	NaN
75%	NaN	NaN	NaN
max	NaN	NaN	NaN

	v11	v12	v13	v16	v19
count	66096.000000	66096.000000	6.609600e+04	64481.000000	64481.000000
unique	NaN	NaN	NaN	NaN	NaN
top	NaN	NaN	NaN	NaN	NaN
freq	NaN	NaN	NaN	NaN	NaN
mean	197.489303	194.068522	-4.480422e+07	7.639153	928.980186
std	111.878834	101.801226	3.069370e+08	2.721422	912.954666
min	1.000000	0.000000	-2.147484e+09	0.000000	6.845938
25%	131.000000	129.000000	0.000000e+00	5.769231	458.022495
50%	196.000000	194.000000	0.000000e+00	7.558140	626.276265
75%	259.000000	257.000000	0.000000e+00	9.342835	999.468193
max	4074.000000	954.000000	1.660000e+02	56.250000	26088.791950

	v20	v21	v22	v23	v24
count	64491.000000	64491.000000	64491.000000	64491.000000	64491.000000
unique	NaN	NaN	NaN	NaN	NaN
top	NaN	NaN	NaN	NaN	NaN
freq	NaN	NaN	NaN	NaN	NaN
mean	4.315545	0.620825	14.678787	55.461301	24.923542
std	5.977543	1.521744	11.500893	17.323476	23.361612
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	1.000000	0.000000	5.384615	48.065407	7.519844
50%	2.304147	0.000000	12.448133	60.073260	16.120219
75%	5.035971	0.724638	21.568627	67.521368	35.542169
max	100.000000	100.000000	100.000000	100.000000	100.000000

	v25	v26	v27	v28	v29
count	64491.000000	64491.000000	64534.000000	64481.000000	64534.000000
unique	NaN	NaN	NaN	NaN	NaN
top	NaN	NaN	NaN	NaN	NaN
freq	NaN	NaN	NaN	NaN	NaN
mean	15.299612	3.471985	47.190961	1648.450535	94.712482
std	12.216109	4.431347	5.128672	1715.200201	5.327575
min	0.000000	0.000000	12.000000	0.000000	0.000000
25%	5.533597	0.561798	43.608442	829.866667	92.307692
50%	12.765957	2.027027	47.026536	1128.031915	96.124031
75%	22.341384	4.733728	50.675902	1741.792929	98.644068
max	100.000000	100.000000	81.000000	73312.500000	100.000000

```
                v30           v40           v41          v42           v43  \
count   64534.000000  64481.000000  64481.000000  64481.00000  64481.000000
unique           NaN           NaN           NaN          NaN           NaN
top              NaN           NaN           NaN          NaN           NaN
freq             NaN           NaN           NaN          NaN           NaN
mean       13.624916     89.748210     87.988724     95.53692     99.790313
std         7.573649     27.852032     27.109637     17.13362      2.193945
min         0.000000      0.000000      0.000000      0.00000      0.000000
25%         8.411215     99.270073     96.341463    100.00000    100.000000
50%        12.500000    100.000000    100.000000    100.00000    100.000000
75%        17.424242    100.000000    100.000000    100.00000    100.000000
max       100.000000    100.000000    100.000000    100.00000    100.000000

          zone code_muni  name_muni code_district name_district code_state
count    66096     66096      66096         66096         66096      66096
unique       2       645        645          1036          1029          1
top     URBANO   3550308  São Paulo     350950205      Campinas         35
freq     60482     18363      18363          1538          1538      66096
mean       NaN       NaN        NaN           NaN           NaN        NaN
std        NaN       NaN        NaN           NaN           NaN        NaN
min        NaN       NaN        NaN           NaN           NaN        NaN
25%        NaN       NaN        NaN           NaN           NaN        NaN
50%        NaN       NaN        NaN           NaN           NaN        NaN
75%        NaN       NaN        NaN           NaN           NaN        NaN
max        NaN       NaN        NaN           NaN           NaN        NaN
```

If you set remove the `include` parameter, only the numerical variables will considered for the summary statistics:

```
cntr_sp_ipvs.describe()
```

```
               v11           v12           v13           v16           v19  \
count  66096.000000  66096.000000  6.609600e+04  64481.000000  64481.000000
mean     197.489303    194.068522 -4.480422e+07      7.639153    928.980186
std      111.878834    101.801226  3.069370e+08      2.721422    912.954666
min        1.000000      0.000000 -2.147484e+09      0.000000      6.845938
25%      131.000000    129.000000  0.000000e+00      5.769231    458.022495
50%      196.000000    194.000000  0.000000e+00      7.558140    626.276265
75%      259.000000    257.000000  0.000000e+00      9.342835    999.468193
max     4074.000000    954.000000  1.660000e+02     56.250000  26088.791950

               v20           v21           v22           v23           v24  \
count  64491.000000  64491.000000  64491.000000  64491.000000  64491.000000
mean       4.315545      0.620825     14.678787     55.461301     24.923542
std        5.977543      1.521744     11.500893     17.323476     23.361612
min        0.000000      0.000000      0.000000      0.000000      0.000000
25%        1.000000      0.000000      5.384615     48.065407      7.519844
50%        2.304147      0.000000     12.448133     60.073260     16.120219
75%        5.035971      0.724638     21.568627     67.521368     35.542169
max      100.000000    100.000000    100.000000    100.000000    100.000000

               v25           v26           v27           v28           v29  \
count  64491.000000  64491.000000  64534.000000  64481.000000  64534.000000
mean      15.299612      3.471985     47.190961   1648.450535     94.712482
```

```
std        12.216109       4.431347       5.128672   1715.200201       5.327575
min         0.000000       0.000000      12.000000      0.000000       0.000000
25%         5.533597       0.561798      43.608442    829.866667      92.307692
50%        12.765957       2.027027      47.026536   1128.031915      96.124031
75%        22.341384       4.733728      50.675902   1741.792929      98.644068
max       100.000000     100.000000      81.000000  73312.500000     100.000000

                v30            v40            v41           v42            v43
count  64534.000000   64481.000000   64481.000000   64481.00000   64481.000000
mean      13.624916      89.748210      87.988724      95.53692      99.790313
std        7.573649      27.852032      27.109637      17.13362       2.193945
min        0.000000       0.000000       0.000000       0.00000       0.000000
25%        8.411215      99.270073      96.341463     100.00000     100.000000
50%       12.500000     100.000000     100.000000     100.00000     100.000000
75%       17.424242     100.000000     100.000000     100.00000     100.000000
max      100.000000     100.000000     100.000000     100.00000     100.000000
```

Properties of data

Description statistics are information reduction techniques. Recall that the objective of EDA is to see the data from different perspectives. Two important properties of data that we often wish to summarize are their central tendency and dispersion. These are discussed next.

Central tendency

A measure of central tendency is a summary of a distribution of values that gives a "typical" value, or the one most frequently observed. Conceptually, this is similar to organizing all data values and finding the location of the *center of mass* of the distribution. To illustrate the concept of center of mass consider the following sequence of quantitative values:

```
x = [20, 30, 32, 34, 41, 41, 45, 46, 48, 51, 53, 54, 54, 56, 57, 58, 58, 59, 60, 61, 64, 65, 65, 69, 71
```

The same sequence of values is shown below in the style of a stem-and-leaf table:

stem	leaf
2	0
3	024
4	11568
5	134467889
6	014559
7	1479
8	8
9	4

Where is the distribution "heavier"? Thereabouts will be its center of mass. There are various measures of central tendency, three of which are discussed next.

In the case of nominal variables, the categories do not have a meaningful order, and yet the center of mass is always the same. Consider for instance:

Mode

The mode of a distribution is the most frequent value found in a distribution. Since it only involves counting the instances of each values, it is appropriate for nominal and ordinal variables. We can find the mode by tabulating the values. Let us do so for the variable `AGSN` (factor) in data frame `cntr_sp_ipvs`.

```
cntr_sp_ipvs[["AGSN"]].describe()
```

```
                AGSN
count          66096
unique             2
top     Não especial
freq           61977
```

We see that the mode (top) of the distribution is "Não especial" (Non-subnormal), the most frequent value of the variable in this distribution. We can see that there are 66096 values for the "AGSN" variable, two possible unique values, with 61977 of the rows referring to the "Não especial" category.

However, how can we summarize *all* possible values in a categorical variable from a Pandas DataFrame? To do this, we can use the `value_counts()` method that first check for all categories within a variable, and then count the amount of instances in each of them:

```
cntr_sp_ipvs["AGSN"].value_counts()
```

```
AGSN
Não especial    61977
Subnormal        4119
Name: count, dtype: int64
```

If you want to check this information as proportions, you can set the `normalize` parameter as `True` within the `value_counts()` method:

```
cntr_sp_ipvs["AGSN"].value_counts(normalize=True)
```

```
AGSN
Não especial    0.937682
Subnormal       0.062318
Name: proportion, dtype: float64
```

Now, we see that the "Não especial" refers to around 94% of the cases, while the "Subnormal" category represent around 6% of all census tracts of the Sao Paulo Metropolitan Region.

Next, let us try variable `IPVS` (ordered factor). First as count:

```
cntr_sp_ipvs["IPVS"].value_counts()
```

```
IPVS
Vulnerabilidade muito baixa    25365
Vulnerabilidade média          10879
Vulnerabilidade baixa           9910
Não classificado                6323
Vulnerabilidade alta            6196
Baixíssima vulnerabilidade      4980
Vulnerabilidade muito alta      2443
Name: count, dtype: int64
```

And as proportions:

```
cntr_sp_ipvs["IPVS"].value_counts(normalize=True)
```

```
IPVS
Vulnerabilidade muito baixa    0.383760
Vulnerabilidade média          0.164594
Vulnerabilidade baixa          0.149933
Não classificado               0.095664
Vulnerabilidade alta           0.093742
Baixíssima vulnerabilidade     0.075345
Vulnerabilidade muito alta     0.036961
Name: proportion, dtype: float64
```

We see that the mode of this distribution is "Vulnerabilidade muito baixa" (extremely low vulnerability). Since ordinal variables have by definition a natural order, the shape of their distribution can be conveniently presented in the style of a stem-and-leaf table, with each "I" representing a thousand instances of the value:

stem	leaf
Não classificado	IIIII I
Baixíssima vulnerabilidade	IIIII
Vulnerabilidade muito baixa	IIIII IIIII IIIII IIIII IIIII
Vulnerabilidade baixa	IIIII IIIII
Vulnerabilidade média	IIIII IIIII I
Vulnerabilidade alta	IIIII I
Vulnerabilidade muito alta	II

Median

The median is the quantile that splits a quantitative variables in two parts of equal size, the bottom 50% and the top 50% of values.

Check again the stem-and-leaf table of our sample quantitative variable.

stem	leaf
2	0
3	024
4	11568
5	134467889
6	014559
7	1479
8	8
9	4

There are $n = 30$ observations in this vector. Which value splits the distribution in half?

For a `list` data with quantative values, as the `x` dataset, the median can be reported by using the `median()` method from the `numpy` library:

```
np.median(x)
```

```
57.5
```

Mean

The mean is probably the best known measure of central tendency, and it is defined as the sum of the values divided by the number of observations. Since it involves arithmetic operations it is not appropriate for categorical variables. The mean of quantitative variables can be reported by using the `mean()` method from the `numpy` library:

```
np.mean(x)
```

```
56.8
```

Spread

Another important property of a distribution of values is how wide or compact it is. Compare the two steam-and-leaf tables below.

stem	leaf
2	0
3	024
4	11568
5	134467889
6	014559
7	1479
8	8
9	4

stem	leaf
1	48
2	08
3	024
4	1156
5	13789
6	01459
7	149
8	468
9	45
10	7

The first stem-and-leaf table is more "compact": the tails of the distribution are closer together and the center of mass is "heavier", compared to the second table, that has a wider spread.

Minimum and maximum

The minimum and maximum values give an idea of how spread the distribution is. In the first of the preceding tables the minimum is 20 and the maximum is 94. In the second table, the minimum is 14 and the maximum is 107. The *range* is the difference between the maximum and the minimum:

```
94 - 20
```

```
[1] 74
```

```
107 - 14
```

```
[1] 93
```

The second distribution is more spread.

Inter-quartile range

The inter-quartile range is similar to the range, but instead of being calculated using the minimum and maximum values of the distribution, it uses the third and first quartiles. Quartiles are a form of quantile that divides a sequence of values in four equal parts, so the second quantile represents the value that separates the lowest 25% of the sample from the remaining 75%, and the third quantile is the value that splits the highest 25% of the sample from the lowest 75%.

If we describe the "v28" variable (average income of the person responsible for the household), we see that the quartiles are reported (25% is the first quartile, 50% is the median, and 75% is the third quartiles). The inter-quartile range can be calculated using those values.

```
cntr_sp_ipvs["v28"].describe()
```

```
count    64481.000000
mean      1648.450535
std       1715.200201
min          0.000000
25%        829.866667
50%       1128.031915
75%       1741.792929
max      73312.500000
Name: v28, dtype: float64
```

The difference between the third and first quartile is:

```
1741.792929 - 829.866667
```

```
[1] 911.9263
```

The inter-quartile range involves an arithmetic operation, which is why it is not an appropriate statistic for categorical variables.

Variance and standard deviation

The variance is another widely used measure of the spread of a distribution. It is defined as:

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

In this formula, $\bar{x}$ is the mean of x and n is the number of observations in the sample. Accordingly, $x_i - \bar{x}$ is the deviation of x_i from the mean of x. If we rewrite this as follows:

$$z_i = (x_i - \bar{x})^2$$

It is easy to see that the variance is actually the mean of the square of the deviations from the mean:

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n} z_i$$

The standard deviation is simply the square root of the variance and returns the variance to the same units as the original variable. The standard deviation is reported by using the `description()` method for pandas DataFrames:

```
cntr_sp_ipvs["v28"].describe()
```

```
count    64481.000000
mean      1648.450535
std       1715.200201
min          0.000000
25%        829.866667
50%       1128.031915
75%       1741.792929
max      73312.500000
Name: v28, dtype: float64
```

The standard deviation can also be calculated with the `std()` method of the `numpy` library:

```
np.std(cntr_sp_ipvs["v28"])
```

```
1715.1869010052912
```

We see that the typical deviation from the mean of `v28` (average income of the person responsible for the household) was about 1715.2 R$. To check for its variance, we use the method `var()`, also from `numpy`:

```
np.var(cntr_sp_ipvs["v28"])
```

```
2941866.1053801347
```

Univariate description

Summary statistics of central tendency and spread refer to a single variable and are appropriately called univariate descriptors. These descriptors are very important, and we neglect exploring them at our own peril. They often tell us important aspects of the data, including how complete a data set is, how much variation is there, whether there are atypical or unusual values.

As an example, let us calculate the mean, standard deviation, and maximum of `v28` (average income of the person responsible for the household):

```
mean_v28 = np.mean(cntr_sp_ipvs["v28"])

sd_v28 = np.std(cntr_sp_ipvs["v28"])

max_v28 = np.max(cntr_sp_ipvs["v28"])
```

The maximum average income of the person responsible for the household in the data set was R$ $73,312.50$. Just how common or unusual is this value? That depends on how close (or far away) from the mean of the distribution this is, as well as on the spread of the distribution. The deviation from the mean is:

```
max_v28 - mean_v28
```

```
71664.04946463306
```

That is, approximately R$ $71,664$. But the typical deviation from the mean in the sample was only about R$ $1,715.20$! Now, calculating:

```
(max_v28 - mean_v28)/sd_v28
```

```
41.7820643468242
```

This tells us that the census tract with the highest average income receives over forty-one times more than the average income of the census tracts. This observation is indeed quite unusual. How unusual was the census tract with the lowest average income? Let us retrieve the minimum duration:

```
min_v28 = np.min(cntr_sp_ipvs["v28"])
min_v28
```

```
0.0
```

That is, R$ 0. Again, the typical deviation from the mean in the sample was R$ $1,715.20$! So, calculating:

```
(min_v28 - mean_v28)/sd_v28
```

```
-0.9610909075860862
```

The census tract with the lowest average income is much closer to the mean, and approximately one standard deviation below the mean.

Univariate description is a powerful way to get to know our data before doing any more sophisticated explorations or analysis.

Bivariate description

Moving on from univariate description, understanding how two variables relate to one another is another key aspect of EDA.

Categorical variables: cross-tabulations

Univariate description of a categorical variable involves tabulating the number of instances of each response. This can be expanded to simultaneously tabulating two categorical variables. Again, we can use the `value_counts()` method, but now passing two (or more) variables to perfom the counts:

```
cntr_sp_ipvs[["AGSN", "IPVS"]].value_counts()
```

```
AGSN          IPVS
Não especial  Vulnerabilidade muito baixa    25321
              Vulnerabilidade média           9910
              Vulnerabilidade baixa           9504
              Vulnerabilidade alta            6196
              Não classificado                6067
              Baixíssima vulnerabilidade      4979
Subnormal     Vulnerabilidade muito alta      2443
              Vulnerabilidade média            969
              Vulnerabilidade baixa            406
              Não classificado                 256
              Vulnerabilidade muito baixa       44
              Baixíssima vulnerabilidade         1
Name: count, dtype: int64
```

And the values can be displayed as proportions:

```
cntr_sp_ipvs[["AGSN", "IPVS"]].value_counts(normalize = "True")
```

```
AGSN          IPVS
Não especial  Vulnerabilidade muito baixa    0.383094
              Vulnerabilidade média          0.149933
              Vulnerabilidade baixa          0.143791
              Vulnerabilidade alta           0.093742
              Não classificado               0.091791
              Baixíssima vulnerabilidade     0.075330
Subnormal     Vulnerabilidade muito alta     0.036961
              Vulnerabilidade média          0.014660
              Vulnerabilidade baixa          0.006143
              Não classificado               0.003873
              Vulnerabilidade muito baixa    0.000666
              Baixíssima vulnerabilidade     0.000015
Name: proportion, dtype: float64
```

What do we learn from this table?

Another way of displaying this information is by doing a cross tabulation by applying the method `.crosstab()`:

```
pd.crosstab(cntr_sp_ipvs["AGSN"], cntr_sp_ipvs["IPVS"])
```

```
IPVS          Não classificado  Baixíssima vulnerabilidade  \
AGSN
Não especial              6067                        4979
Subnormal                  256                           1

IPVS          Vulnerabilidade muito baixa  Vulnerabilidade baixa  \
AGSN
Não especial                        25321                   9504
Subnormal                              44                    406

IPVS          Vulnerabilidade média  Vulnerabilidade alta  \
AGSN
Não especial                   9910                  6196
Subnormal                       969                     0

IPVS          Vulnerabilidade muito alta
AGSN
Não especial                           0
Subnormal                           2443
```

To improve the readability of the previous output, we can add the total sum of the columns as a row at the bottom of the table:

```
crosstab_agsn_ipvs = pd.crosstab(cntr_sp_ipvs["AGSN"], cntr_sp_ipvs["IPVS"])
crosstab_agsn_ipvs.loc["Total"] = crosstab_agsn_ipvs.sum()
crosstab_agsn_ipvs
```

```
IPVS          Não classificado  Baixíssima vulnerabilidade  \
AGSN
Não especial              6067                        4979
Subnormal                  256                           1
Total                     6323                        4980
```

```
IPVS          Vulnerabilidade muito baixa  Vulnerabilidade baixa  \
AGSN
Não especial                        25321                   9504
Subnormal                              44                    406
Total                               25365                   9910

IPVS          Vulnerabilidade média  Vulnerabilidade alta  \
AGSN
Não especial                   9910                  6196
Subnormal                       969                     0
Total                         10879                  6196

IPVS          Vulnerabilidade muito alta
AGSN
Não especial                           0
Subnormal                           2443
Total                               2443
```

Or the total sums of the rows as a column at the right of the table:

```
crosstab_agsn_ipvs["Total"] = crosstab_agsn_ipvs.sum(axis=1)
crosstab_agsn_ipvs
```

```
IPVS          Não classificado  Baixíssima vulnerabilidade  \
AGSN
Não especial              6067                        4979
Subnormal                  256                           1
Total                     6323                        4980

IPVS          Vulnerabilidade muito baixa  Vulnerabilidade baixa  \
AGSN
Não especial                        25321                   9504
Subnormal                              44                    406
Total                               25365                   9910

IPVS          Vulnerabilidade média  Vulnerabilidade alta  \
AGSN
Não especial                   9910                  6196
Subnormal                       969                     0
Total                         10879                  6196

IPVS          Vulnerabilidade muito alta  Total
AGSN
Não especial                           0  61977
Subnormal                           2443   4119
Total                               2443  66096
```

There are in total $n = 66096$ valid observations when we consider variables `IPVS` and `AGSN` simultaneously. What else do we learn from this table?

If the category of the census tracts did not vary by the type of index category, we would expect the percentages in the bottom row to be roughly the same, since every category of census tract would have a uniform chance of being in any state. We can calculate the values in the table *if* this was true.

This is done by multiplying the row total by the column total for each `IPVS` and `AGSN` combination, and dividing by the size of the sample:

$$E_{ij} = \frac{1}{n}\sum_i x_i \sum_j x_j$$

For example, the expected value for "Baixíssima vulnerabilidade" and "Subnormal" is:

```
(4980 * 4119)/66096
```

```
[1] 310.3459
```

The observed value, on the other hand, is 0. Therefore, we see that census tracts with very low vulnerability belong to the subnormal class less often than if the precariousness of the census tracts were random. It is possible to summarize the differences between the observed and expected values in a cross-tabulation:

$$\chi^2 = \sum_i \sum_j \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$$

If the observed and expected values are identical in every case, the value of χ^2 would be zero. Contrariwise, χ^2 will tend to grown as the differences between observed and expected counts grow. This would suggest that the observed counts are unlikely to follow a random pattern.

We can use the `chi2_contingency()` function from the `scipy` library to computes χ^2 and produces also a p-value to aid in the decision whether the distribution follows a random pattern or not. A small p-value would indicate a small probability of the distribution being random:

```
chi2_table = pd.crosstab(cntr_sp_ipvs["IPVS"], cntr_sp_ipvs["AGSN"])
chi2_table
```

```
AGSN                          Não especial  Subnormal
IPVS
Não classificado                      6067        256
Baixíssima vulnerabilidade            4979          1
Vulnerabilidade muito baixa          25321         44
Vulnerabilidade baixa                 9504        406
Vulnerabilidade média                 9910        969
Vulnerabilidade alta                  6196          0
Vulnerabilidade muito alta               0       2443
```

```
chi2_contingency(chi2_table)
```

```
Chi2ContingencyResult(statistic=39354.83807837498, pvalue=0.0, dof=6, expected_freq=array([[ 5928.96046
       [ 4669.65413943,   310.34586057],
       [23784.29261983,  1580.70738017],
       [ 9292.42420116,   617.57579884],
       [10201.03762709,   677.96237291],
       [ 5809.87490922,   386.12509078],
       [ 2290.75603667,   152.24396333]]))
```

To visualize the test results in a better way, we can create different variables and assign their values by:

```
stat, p, dof, expected = chi2_contingency(chi2_table)
```

```
stat
```

```
39354.83807837498
```

```
p
```

```
0.0
```

```
dof
```

```
6
```

```
expected
```

```
array([[ 5928.96046659,   394.03953341],
       [ 4669.65413943,   310.34586057],
       [23784.29261983,  1580.70738017],
       [ 9292.42420116,   617.57579884],
       [10201.03762709,   677.96237291],
       [ 5809.87490922,   386.12509078],
       [ 2290.75603667,   152.24396333]])
```

We also get a DataFrame with expected values:

```
expected_df = pd.DataFrame(expected, index = chi2_table.index, columns = chi2_table.columns)
expected_df
```

```
AGSN                         Não especial    Subnormal
IPVS
Não classificado              5928.960467   394.039533
Baixíssima vulnerabilidade    4669.654139   310.345861
Vulnerabilidade muito baixa  23784.292620  1580.707380
Vulnerabilidade baixa         9292.424201   617.575799
Vulnerabilidade média        10201.037627   677.962373
Vulnerabilidade alta          5809.874909   386.125091
Vulnerabilidade muito alta    2290.756037   152.243963
```

Which can be compared to the observed counts:

```
chi2_table
```

```
AGSN                         Não especial  Subnormal
IPVS
Não classificado                     6067        256
Baixíssima vulnerabilidade           4979          1
Vulnerabilidade muito baixa         25321         44
Vulnerabilidade baixa                9504        406
Vulnerabilidade média                9910        969
Vulnerabilidade alta                 6196          0
Vulnerabilidade muito alta              0       2443
```

We see that the distribution is dominated by census tracts of class “Vulnerabilidade muito baixa”, which tend to be close to the expected values.

Quantitative variables: correlation

The mean and standard deviation are key univariate descriptors of quantitative variables. When we are interested in the relationship between two quantitative variables we use a related concept, the *covariance*. The covariance is the mean of the product of the deviations from the mean of two variables:

$$C(x, y) = \frac{1}{n} \sum_i (x_i - \bar{x})(y_i - \bar{y})$$

In the formula above, $\bar{x}$ and $\bar{y}$ are the means of the two variables. The differences from the mean can be positive, negative, or zero. When both deviations are positive, the product is positive, thus increasing the covariance. Likewise when the two deviations are negative. But when one deviation is positive and the other negative, the product is negative, which subtracts from the covariance. When differences are like-like (positive-positive or negative-negative) the covariance will tend to be a large positive number. The contrary happens when the differences are opposed.

The covariance can be normalized by the standard deviations of the variables to give the correlation coefficient:

$$r(x, y) = \frac{C(x, y)}{\sigma_x \cdot \sigma_y}$$

This quantity has the census tract of being bound between [-1, 1], and a value of zero indicates that the two variables do not *covary*. Correlations can be calculated from DataFrames using the `corr()` method:

```
cntr_sp_ipvs[["v28", "v19"]].corr()
```

```
          v28       v19
v28  1.000000  0.956923
v19  0.956923  1.000000
```

What does the correlation between these two variables indicate?

Categorical variables: multiple cross-tabulations

It is possible to cross-tabulate more than two variables, however in practice this is seldom done for more than three because the results quickly become difficult to read and interpret. Since the `cntr_sp_ipvs` has only two categorical variables, we'll join this data set with the `cntr_sp_basico`:

```
cntr_sp_ipvs_merged = cntr_sp_ipvs.merge(cntr_sp_basico, left_on = "COD_SETOR", right_on = "code_tract"
```

Note that because of the identifiable variable has different names ("COD_SETOR" and "code_tract"), we need to speficy the parameters `left_on` related to the first DataFrame (cntr_sp_ipvs), and `right_on` that refers to the second DataFrame (cntr_sp_basico).

Using the method `.value_counts()`:

```
cntr_sp_ipvs_merged[["AGSN", "situacao", "IPVS"]].value_counts()
```

```
AGSN          situacao                                  IPVS
Não especial  Área urbanizada de cidade ou vila         Vulnerabilidade muito baixa    23685
                                                        Vulnerabilidade média           9244
                                                        Vulnerabilidade baixa           8478
                                                        Vulnerabilidade alta            4828
                                                        Baixíssima vulnerabilidade      4764
                                                        Não classificado                2613
Subnormal     Área urbanizada de cidade ou vila         Vulnerabilidade muito alta      2429
Não especial  Zona rural, exclusive aglomerado rural    Não classificado                1981
```

```
                                                             Vulnerabilidade muito baixa     1273
Subnormal     Área urbanizada de cidade ou vila              Vulnerabilidade média            968
Não especial  Zona rural, exclusive aglomerado rural         Vulnerabilidade alta             927
                                                             Vulnerabilidade baixa            700
              Área não urbanizada de cidade ou vila          Não classificado                 692
              Área urbana isolada                            Não classificado                 617
Subnormal     Área urbanizada de cidade ou vila              Vulnerabilidade baixa            405
Não especial  Área não urbanizada de cidade ou vila          Vulnerabilidade média            361
Subnormal     Área urbanizada de cidade ou vila              Não classificado                 253
Não especial  Área não urbanizada de cidade ou vila          Vulnerabilidade alta             216
              Área urbana isolada                            Vulnerabilidade média            199
              Área não urbanizada de cidade ou vila          Vulnerabilidade muito baixa      187
                                                             Vulnerabilidade baixa            174
              Zona rural, exclusive aglomerado rural         Baixíssima vulnerabilidade       150
              Área urbana isolada                            Vulnerabilidade alta             103
                                                             Vulnerabilidade muito baixa       98
              Aglomerado rural de extensão urbana            Vulnerabilidade alta              76
              Área urbana isolada                            Vulnerabilidade baixa             72
              Aglomerado rural de extensão urbana            Vulnerabilidade média             69
              Aglomerado rural isolado - outros aglomerados  Não classificado                  67
              Aglomerado rural de extensão urbana            Não classificado                  66
                                                             Vulnerabilidade baixa             45
Subnormal     Área urbanizada de cidade ou vila              Vulnerabilidade muito baixa       44
Não especial  Aglomerado rural de extensão urbana            Vulnerabilidade muito baixa       39
              Zona rural, exclusive aglomerado rural         Vulnerabilidade média             37
              Aglomerado rural isolado - povoado             Vulnerabilidade alta              36
              Aglomerado rural isolado - outros aglomerados  Vulnerabilidade muito baixa       22
              Área não urbanizada de cidade ou vila          Baixíssima vulnerabilidade        22
              Aglomerado rural isolado - povoado             Vulnerabilidade baixa             19
              Área urbana isolada                            Baixíssima vulnerabilidade        18
              Aglomerado rural isolado - núcleo              Não classificado                  18
              Aglomerado rural isolado - povoado             Vulnerabilidade muito baixa       14
              Aglomerado rural isolado - outros aglomerados  Vulnerabilidade baixa             13
              Aglomerado rural isolado - povoado             Não classificado                  13
              Aglomerado rural isolado - outros aglomerados  Baixíssima vulnerabilidade        11
                                                             Vulnerabilidade alta              10
              Aglomerado rural de extensão urbana            Baixíssima vulnerabilidade         8
Subnormal     Área não urbanizada de cidade ou vila          Vulnerabilidade muito alta         8
              Aglomerado rural de extensão urbana            Vulnerabilidade muito alta         4
Não especial  Aglomerado rural isolado - núcleo              Baixíssima vulnerabilidade         4
                                                             Vulnerabilidade muito baixa        3
                                                             Vulnerabilidade baixa              3
Subnormal     Área não urbanizada de cidade ou vila          Não classificado                   2
              Área urbana isolada                            Vulnerabilidade muito alta         2
Não especial  Aglomerado rural isolado - povoado             Baixíssima vulnerabilidade         2
Subnormal     Aglomerado rural de extensão urbana            Vulnerabilidade média              1
              Área urbana isolada                            Vulnerabilidade baixa              1
              Aglomerado rural de extensão urbana            Não classificado                   1
              Área urbanizada de cidade ou vila              Baixíssima vulnerabilidade         1
Name: count, dtype: int64
```

The chi^2 summary of association for a cross-tabulation seen above no longer works for tables in higher dimensions (i.e., when the number of variables cross-tabulated is greater than two). For an example of alternative approaches to describe tables with more than two variables, see the application of Cochran-

Mantel-Haenszel χ^2 statistic in Mella-Lira and Paez (2021).

Quantitative variables: correlation matrices

Correlations can be explored among multiple variables. The example below shows how to select the quantitative variables from a data frame and obtain a correlation matrix:

```
cntr_sp_ipvs.select_dtypes(include=['number']).corr()
```

```
          v11       v12       v13       v16       v19       v20       v21  \
v11  1.000000  0.878547  0.193525  0.059918 -0.019068 -0.051714 -0.050226
v12  0.878547  1.000000  0.262772  0.059523 -0.024789 -0.052490 -0.051395
v13  0.193525  0.262772  1.000000  0.076160 -0.021957  0.027524  0.043107
v16  0.059918  0.059523  0.076160  1.000000 -0.376486  0.101923  0.326834
v19 -0.019068 -0.024789 -0.021957 -0.376486  1.000000 -0.098330 -0.199692
v20 -0.051714 -0.052490  0.027524  0.101923 -0.098330  1.000000  0.134759
v21 -0.050226 -0.051395  0.043107  0.326834 -0.199692  0.134759  1.000000
v22 -0.099796 -0.098629  0.030677  0.583757 -0.558097  0.174360  0.418454
v23  0.078206  0.090604  0.004844  0.210822 -0.730871 -0.253502 -0.059020
v24  0.007640 -0.001853 -0.028545 -0.490263  0.854191 -0.162505 -0.261858
v25 -0.100209 -0.099257  0.034251  0.589140 -0.549662  0.180938  0.518524
v26 -0.094546 -0.096358  0.041122  0.508395 -0.383454  0.208738  0.711928
v27 -0.122935 -0.127519 -0.017360 -0.682189  0.308199 -0.107536 -0.198024
v28 -0.026130 -0.032962 -0.019906 -0.307116  0.956923 -0.107346 -0.176763
v29  0.100052  0.101001 -0.044009 -0.369814  0.427343 -0.084350 -0.349817
v30  0.072633  0.078256  0.028338  0.510229 -0.274644  0.141703  0.198201
v40  0.360502  0.386053 -0.029148 -0.079208  0.147163 -0.001964 -0.140734
v41  0.247340  0.258120 -0.027590 -0.223625  0.217448 -0.079513 -0.252931
v42  0.305123  0.318539 -0.043871 -0.042784  0.128797  0.009121 -0.132810
v43  0.082220  0.085369 -0.028035 -0.073273  0.055814 -0.091408 -0.219229

          v22       v23       v24       v25       v26       v27       v28  \
v11 -0.099796  0.078206  0.007640 -0.100209 -0.094546 -0.122935 -0.026130
v12 -0.098629  0.090604 -0.001853 -0.099257 -0.096358 -0.127519 -0.032962
v13  0.030677  0.004844 -0.028545  0.034251  0.041122 -0.017360 -0.019906
v16  0.583757  0.210822 -0.490263  0.589140  0.508395 -0.682189 -0.307116
v19 -0.558097 -0.730871  0.854191 -0.549662 -0.383454  0.308199  0.956923
v20  0.174360 -0.253502 -0.162505  0.180938  0.208738 -0.107536 -0.107346
v21  0.418454 -0.059020 -0.261858  0.518524  0.711928 -0.198024 -0.176763
v22  1.000000  0.225576 -0.731443  0.993579  0.807569 -0.447422 -0.497979
v23  0.225576  1.000000 -0.783879  0.205018  0.013712 -0.182649 -0.678195
v24 -0.731443 -0.783879  1.000000 -0.721238 -0.507517  0.396121  0.786409
v25  0.993579  0.205018 -0.721238  1.000000  0.848972 -0.445895 -0.490282
v26  0.807569  0.013712 -0.507517  0.848972  1.000000 -0.351040 -0.339859
v27 -0.447422 -0.182649  0.396121 -0.445895 -0.351040  1.000000  0.298843
v28 -0.497979 -0.678195  0.786409 -0.490282 -0.339859  0.298843  1.000000
v29 -0.646902 -0.278207  0.569139 -0.652604 -0.557009  0.210385  0.383907
v30  0.379990  0.147003 -0.345245  0.382432  0.314508 -0.783339 -0.318295
v40 -0.327827 -0.049373  0.207351 -0.325694 -0.258737 -0.010095  0.117313
v41 -0.455367 -0.051369  0.298487 -0.459323 -0.404242  0.135346  0.186147
v42 -0.296637 -0.046086  0.186230 -0.295365 -0.228049 -0.032071  0.104602
v43 -0.121072  0.033691  0.071776 -0.140425 -0.190824  0.004441  0.049137

          v29       v30       v40       v41       v42       v43
```

```
v11  0.100052  0.072633  0.360502  0.247340  0.305123  0.082220
v12  0.101001  0.078256  0.386053  0.258120  0.318539  0.085369
v13 -0.044009  0.028338 -0.029148 -0.027590 -0.043871 -0.028035
v16 -0.369814  0.510229 -0.079208 -0.223625 -0.042784 -0.073273
v19  0.427343 -0.274644  0.147163  0.217448  0.128797  0.055814
v20 -0.084350  0.141703 -0.001964 -0.079513  0.009121 -0.091408
v21 -0.349817  0.198201 -0.140734 -0.252931 -0.132810 -0.219229
v22 -0.646902  0.379990 -0.327827 -0.455367 -0.296637 -0.121072
v23 -0.278207  0.147003 -0.049373 -0.051369 -0.046086  0.033691
v24  0.569139 -0.345245  0.207351  0.298487  0.186230  0.071776
v25 -0.652604  0.382432 -0.325694 -0.459323 -0.295365 -0.140425
v26 -0.557009  0.314508 -0.258737 -0.404242 -0.228049 -0.190824
v27  0.210385 -0.783339 -0.010095  0.135346 -0.032071  0.004441
v28  0.383907 -0.318295  0.117313  0.186147  0.104602  0.049137
v29  1.000000 -0.214998  0.340567  0.392734  0.281853  0.181236
v30 -0.214998  1.000000  0.024796 -0.109597  0.028925 -0.022869
v40  0.340567  0.024796  1.000000  0.603978  0.731587  0.173820
v41  0.392734 -0.109597  0.603978  1.000000  0.517000  0.134586
v42  0.281853  0.028925  0.731587  0.517000  1.000000  0.176140
v43  0.181236 -0.022869  0.173820  0.134586  0.176140  1.000000
```

Note that the default `method` is `pearson`. It is possible to select two other correlation methods (setting the paramether `method` as `kendall` or `spearman`) that work on ranked data instead of the quantities. Ranking the values is a more robust way of calculating the association between two variables for reasons that will become clear next session when we discuss visualization approaches for EDA.

Practice

1. Join tables `cntr_sp_ipvs`, `cntr_sp_basico`, and `cntr_sp_head`.
2. Imagine that you are interested in the average age of head of household. Describe and discuss this variable.
3. How does the average age of head of household relate to other variables in the data set? Select 4 variables and discuss.
4. Propose some hypotheses about age of head of household based on your exploration of the data set. How would you propose to investigate these hypotheses?
5. Imagine now that you are interested in the vulnerability group of the census tracts (check `?cntr_sp_ipvs`). Repeat questions 3 and 4 but for this variable.
6. What can you say so far about the relationship between age of head of household and the categorical variables in the data set? Or between vulnerability group and the quantitative variables in the data set? Propose an approach to explore a combination of categorical and quantitative variables.