

An exploratory analysis of monitored products in México with Pandas. PROFECO

Author: Alejandra Elizabeth Moreno Morales

Abstract

In this report, it is presented an exploratory analysis of a database of monitored products used by PROFECO (Mexican Consumer Protection Agency) aiming found key information to make reasonable decisions.

In order to achieve this, it will be used Pandas, a fast, powerful, flexible and easy to use open-source data analysis and manipulation tool, built on top of the Python programming language.¹ This tool fulfills the requirements for manage large databases and also gives the opportunity to be used as manipulation data tool to found useful information.

Introduction

PROFECO (Mexican Consumer Protection Agency), is a government agency tasked with enforcing the law and investigating possible violations. They respond to complaints from residents and tourists, perform random inspections on Mexican businesses and have the authority to fine or close businesses who are in violation.² In an effort to found discrepancies need to analyze products, commercial chains, consumers and its interactions.

As Data Analyst, our task is processing the data stored in a database, into an Google Drive on a csv file of about 20 Gb, that contain the information of monitored products by commercial chains. Python pandas functions was applied to solve the proposed questions and as I do not have IT dedicated resources, a personal computer was used with the following specifications (Processor Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz, RAM 12 Gb, Windows 10 Home) to use this tool. So, it will be required to use some basis concepts of the primary panda's structure, dataframes, its parameters and attributes and some methods such as columns, count, sum, add and group by. Also, to work with datasets that are much larger than memory, chunk operation will be used to split the data into small pieces to be able to process it.

Report Overview

Jupyter Notebook will be used to share live code and comments by questions, due to its characteristics that tie the technologies used in this exploratory analysis.

By question, We are going to notice the same base code structure and ordered in the following sections.

Section. - Variables and declarations. Show the variables used to read de csv file and the used to run the queries needed.

Section. - Processing query by chunks. Show the iteration with the workloads to process que data by queries.

Section. - Fit result to show. Show how dataframes methods are applied to the processed data in order to present as an answer of the requirements.

Section. - Result output file. Show the data return exported into a csv file.

Section. - Memory use of each column along with the index. Show the return of memory usage of each element to evaluated performance.

Having explained the key information about this report, let's go into our questions.

How many commercial chains are monitored, and therefore, included in this database?

To answer this question, I'm going to read our csv file by chunks to then run an operation `drop_duplicates()` in column "cadenaComercial", since its function is removing duplicate rows, and I need to know unique values of "cadenaComercial". It will be used the parameter `keep='first'` to maintain the first occurrence. Then, I'm going to buffer this information in a variable called `queryTemp`, that is going to save the query results by chunks.

When the FOR instruction used to read the file by chunks be concluded, result in variable `queryTemp` be concatenated for then apply other operation of `drop_duplicates()` to remove occurrences of values between chunks pieces.

Then, result will be sorted alphabetically to finally send it to a csv file for a better reading.

```
import pandas as pd

# Variables and declarations
file = ('all_data.csv')
chunk_size = 100000
queryTemp = []
query = pd.DataFrame()
result = pd.DataFrame()

# Processing query by chunks
for chunk in pd.read_csv(file, chunksize=chunk_size, iterator=True, low_memory=False):
    query = chunk['cadenaComercial'].drop_duplicates(keep='first')
    queryTemp.append(query)

# Fit result to show
query = pd.concat(queryTemp).drop_duplicates()
result = query.to_frame()
result.sort_values(by='cadenaComercial', ascending=True, inplace=True)

# Result output file
result.to_csv('result_q1.csv', index=False)

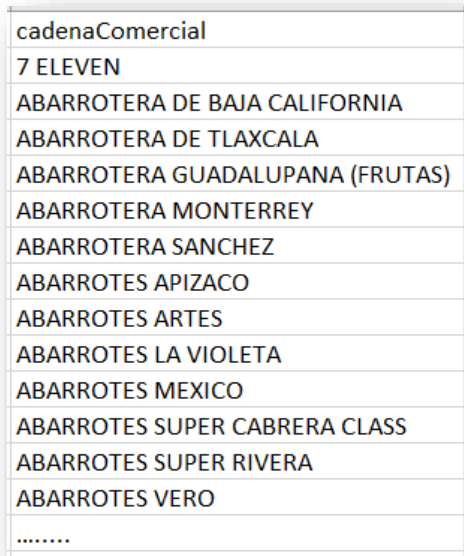
# Memory use of each column along with the index
print(result.memory_usage(index = True))
```

Image 1. Code Question 1

Index	5648
cadenaComercial	5648
dtype: int64	

Image 2.- Memory usage to run code in image 1

As shown in csv file, 704 commercial chains are included in this database with the names displayed in file.



cadenaComercial
7 ELEVEN
ABARROTERA DE BAJA CALIFORNIA
ABARROTERA DE TLAXCALA
ABARROTERA GUADALUPANA (FRUTAS)
ABARROTERA MONTERREY
ABARROTERA SANCHEZ
ABARROTES APIZACO
ABARROTES ARTES
ABARROTES LA VIOLETA
ABARROTES MEXICO
ABARROTES SUPER CABRERA CLASS
ABARROTES SUPER RIVERA
ABARROTES VERO
.....

Image 3.- Result in csv file, question 1

What are the top 10 monitored products by State?

I have been required to show which products are more frequent in each State and to know this information I need to know which products and how many registers are in each State, no matter its presentation, brand or in which commercial chain are sold.

First at all, file will be read by chunks to then apply an operation "group by" with columns "estado" and "producto" to find the products present by State, then I'm going to count the occurrences and save in a variable called "query".

Since occurrences may appear in different chunk pieces and I need to know the final result among all chunks, IF operation will be used to append. In the first round of reading, the query values will be passed to "result" variable and then for the remaining rounds it will execute an "add()" operation to sum the new occurrences found in the current chunk reading.

Having conclude the FOR operation used to read all the file, I'm going to use a groupby operation to get the information by State and so I can get the 10 products more frequent with the method "nlargest()". Finally, I will export my final query to obtain the information required.

```

import pandas as pd

# Variables and declarations
file = ('all_data.csv')
chunk_size = 100000
query = pd.DataFrame()
result = None

# Processing query by chunks
for chunk in pd.read_csv(file, chunksize=chunk_size, iterator=True, low_memory=False):
    query = chunk[['estado', 'producto', 'marca']].groupby(['estado', 'producto']).count()
    if result is None:
        result = query
    else:
        result = result.add(query, fill_value=0)

# Fit result to show
result = result.rename(columns={'marca' : 'count'})
result = result.groupby('estado')['count'].nlargest(10)

# Result output file
result.to_csv('result_q2.csv')

# Memory use of each column along with the index
print(result.memory_usage(index = True))

```

Image 4. Code Question 2

7118

Image 5.- Memory usage to run code in image 4

As has been seen in csv file, I get the top 10 monitored product grouped by State in descending order requested on demand.

estado	producto	count
AGUASCALIENTES	FUD	12005
AGUASCALIENTES	DETERGENTE P/ROPA	10188
AGUASCALIENTES	LECHE ULTRAPASTEURIZADA	9824
AGUASCALIENTES	SHAMPOO	9654
AGUASCALIENTES	REFRESCO	9481
AGUASCALIENTES	DESODORANTE	8859
AGUASCALIENTES	JABON DE TOCADOR	8517
AGUASCALIENTES	CHILES EN LATA	7946
AGUASCALIENTES	YOGHURT	7401
AGUASCALIENTES	MAYONESA	7173
BAJA CALIFORNIA	REFRESCO	37243
BAJA CALIFORNIA	DETERGENTE P/ROPA	23395
BAJA CALIFORNIA	FUD	19967
BAJA CALIFORNIA	SHAMPOO	19123
BAJA CALIFORNIA	JABON DE TOCADOR	18348
BAJA CALIFORNIA	CHILES EN LATA	16676
BAJA CALIFORNIA	GALLETAS	15873
BAJA CALIFORNIA	PANTALLAS	15703
BAJA CALIFORNIA	CEREALES	15398
BAJA CALIFORNIA	DESODORANTE	14748
.....		

Image 6.- Result in csv file, question 2

Which is the commercial chain with the highest number of monitored products?

For this request, I need to know how many different products are present in each commercial chain, no matter if the commercial chain is in one or in other state, or which branch office is, neither which branch or presentation have the products.

To be able to count how many product are in each commercial chain, I will read my database by chunks and group by columns "cadenaComercial" and "producto", in this way I am going to get the an unique product value by commercial chain, so then I am ready to apply a "sum()" method in level 0, which is my column "cadenaComercial", to sum all the products found in each commercial chain.

Since occurrences may appear in different chunk pieces and I need to know the final result among all chunks, IF operation will be used to append. In the first round of reading, the query values will be passed to "result" variable and then for the remaining rounds it will execute an "add()" operation to sum the new occurrences found in the current chunk reading.

Having conclude the FOR operation used to read all the file, I'm going to apply the "nlargest()" method to the dataframe to obtain the commercial chain with the highest number of monitored products. Lastly, I will print my result and also export it in a csv file to store my result.

```
import pandas as pd

# Variables and declarations
file = ('all_data.csv')
chunk_size = 100000
query = pd.DataFrame()
result = None
col_list = ['cadenaComercial', 'nombreComercial', 'producto']

# Processing query by chunks
for chunk in pd.read_csv(file, chunksize=chunk_size, usecols = col_list, iterator=True, low_memory=False):
    query = chunk.groupby(by=['cadenaComercial', 'producto']).all().groupby(level=0).sum()
    if result is None:
        result = query
    else:
        result = result.add(query, fill_value=0)

# Fit result to show
result = result.rename(columns={'nombreComercial' : 'count'})
result = result.nlargest(1, 'count')

# Result output file
result.to_csv('result_q3.csv')

# Result
print('The commercial chain with the highest number of monitored product is: ', result.iloc[:,0])

# Memory use of each column along with the index
print(result.memory_usage(index = True))
```

Image 7. Code Question 3

```
Index      8
count      8
dtype: int64
```

Image 8.- Memory usage to run code in image 7

As shown below, WAL-MART has the highest number of different products monitored in this database.

```
The commercial chain with the highest number of monitored product is: cadenaComercial
WAL-MART      46523.0
```

Image 9.- Result in print method

cadenaComercial	count
WAL-MART	46523

Image 10.- Result in csv file, question 3

Use the data to find an interesting fact.

As well PROFECO has to know the behavior of commercial chains, also It have to be aware of brands or trademarks. So, it will be important to know which products of them are more present in Mexican business. In order to that, I will show the top 3 leading products by trademark.

First at all, I will read our database by chunks and group the information by "marca", "producto" and "presentacion", since I want to find the different versions of products are on sale in the commercial chains. After that, I will count them with a "count()" method to find how many different commercial chains, offices, states have this product presentation in their stocks.

As in question 2 and 3, I will use an IF statement to store the results obtained in each chunk pieces. In the first round of reading, the query values will be passed to "result" variable and then for the remaining rounds it will execute an "add()" operation to sum the new occurrences found in the current chunk reading.

Having finish the FOR statement and read all the file and in consequence obtained the list of different presentation of products by brand and its occurrences, I'm going to use this same aggrupation to get the 3 more frequent with the "nlargest()" method. Last, I will export this ordered information it in a csv file in an effort to found outstanding facts about the presentation of products.

```

import pandas as pd

# Variables and declarations
file = ('all_data.csv')
chunk_size = 100000
query = pd.DataFrame()
result = None

# Processing query by chunks
for chunk in pd.read_csv(file, chunksize=chunk_size, iterator=True, low_memory=False):
    query = chunk[['marca', 'producto', 'presentacion', 'categoria']].groupby(['marca', 'producto', 'presentacion']).count()
    if result is None:
        result = query
    else:
        result = result.add(query, fill_value=0)

# Fit result to show
result = result.rename(columns={'categoria': 'count'})
result = result.groupby(['marca', 'producto'])['count'].nlargest(3)
# Result output file
result.to_csv('result_q4.csv')

# Memory use of each column along with the index
print(result.memory_usage(index = True))

```

Image 11. Code Question 4

509327

Image 12.- Memory usage to run code in image 11

As shown in the csv file, I found interesting fact about Across brand, for example, because now I know which is the lead stove product and realized "AF 1850 800" presentation have a lot of acceptance in Mexico. Also, in washing machines of Across brand, We have the "ALD 1625 AF" as the lead presentation. In brief, the information ordered and show in a purpose way, it could be very useful to make decision about in what should you focus on.

marca	producto	presentacion	count
ACROS	ESTUFAS	AF 1850 800. FRENTE 30 PLGS. 6 QUEMADORES. EN	20416
ACROS	ESTUFAS	AF 5304 M00 O AF 5304 M01. FRENTE 30 PLGS. 6 Q	7844
ACROS	ESTUFAS	AF 7323 D00. FRENTE 30. 6 QUEMADORES. ENCEND	6298
ACROS	LAVADORAS	ALD 1625 AF. 16KGS. IMPULSOR. CENTRIFUGADO (2	4405
ACROS	LAVADORAS	LAPC 2235 BR 3" LAPC 2235 BR1. 22KGS. AGITADOR	2739
ACROS	LAVADORAS	ALP 1515 3" ALP 1515 YR. 15KGS. AGITADOR	2049
ACROS	REFRIGERADORES	AS8950 G (PLATA). 227 DM3. 1 PUERTA VERTICAL. D	3988
ACROS	REFRIGERADORES	AT 9501 G (PLATA). 250 DM3. 2 PUERTAS HORIZONT	3227
ACROS	REFRIGERADORES	AT 9007 G (PLATA). 250 DM3. 2 PUERTAS HORIZONT	1239
ACR%RCATE A LA FÍSICA	LIBRO DE TEXTO DE FISICA	GUTIERREZ ARANZETA CARLOS Y ALICIA ZARZOSA PI	145
ADES	JUGO DE FRUTA	CAJA 946 ML. NARANJA	73010
ADIDAS	DESODORANTE	BARRA 56 GR. FRESH POWER 24 H. ICE DIVE	10042
ADIDAS FRESH IMPACT	DESODORANTE	BARRA 56 GR. 24 HORAS	4171
AJAX AMONIA	LIMPIADOR LIQUIDO P/PISC	BOTELLA 1 LT. MULTIUSOS	70701
AJAX. EXPEL	LIMPIADOR LIQUIDO P/PISC	BOTELLA 1 LT. LIQUIDO. CONCENTRADO	208
AL-DIA	LECHE PASTEURIZADA	ENTERA. BOTELLA 1 LT.	1711
ALBERTO. VO5	SHAMPOO	BOTELLA 800 ML. PASION DE MANGO	20679
ALERT	SHAMPOO	BOTELLA 400 ML. SALUDABLE. NORMAL A GRASO	15996
ALERT	SHAMPOO	BOTELLA 400 ML. CITRUS CABELLO GRASO	13418
ALERT	SHAMPOO	BOTELLA 400 ML. HIDRATANTE	6163
ALL-BRAN KELLOGG S	CEREALES	CAJA 775 GR. ORIGINAL	48217
ALL-BRAN KELLOGG'S	CEREALES	CAJA 620 GR. FLAKES. ORIGINAL	4182
ALPHARMA	METAMIZOL SODICO	CAJA 10 TABLETAS DE 500 MG.	17
ALPINO	JAMON	1 KG. GRANEL. DE PIERNA. EXTRAFINO	143
ALPURA	CREMA	BOTE 450 ML.	54144
ALPURA	CREMA	BOTE 450 ML. REDUCIDA EN GRASA	49023
ALPURA	CREMA	VASO 200 ML.	44836
ALPURA	LECHE EN POLVO	ENTERA. LATA 1,800 KG.	50767

Image 13.- Result in csv file, question 3

What are the lessons learned from this exercise?

The main lessons learned with this exercise are related to know, not only what are you studying in your database or what useful information want to find to a specific purpose, but also it is know your available tools, your hardware and software assets to accomplish the task.

For me, as first challenge, I have my personal computer specifications because since it is not a basic computer, the amount of information and the way I compiled into a query was a relevant aspect in my performance. Also, the pandas, as tool, was a very gratified experience because it allows you to manipulation large amount of data with a very powerful basic functions and methods with the opportunity to enjoy the benefits of a very powerful structures like dataframes and arrays. As well, Pandas offer to many ways to achieve your tasks because for example I could be useless data, or change my datatypes, use series, use multiindexes or pivot tables and if I was not enough, it allows you use many other different libraries like matplotlib, NumPy, SciPy, to very specific purposes.

Can you identify other ways to approach this problem?

Focused on available resources, come on my mind other approaches to face this problem, for example:

1. Dask API, as parallel computing library, using `dask.dataframe` and work with multiple threads to process data in parallel.
2. Distributed file systems like Hadoop and Spark as frameworks to process data in parallel across clusters on single computer.
3. Use compute services of any of Cloud providers in order to get the resources needed to perform queries more efficiently

Conclusion

Exploratory data analysis (EDA), as an approach to analyze data to summarize and deepen into its structure and main characteristics, allows Analysts know the data and how to work with it. This could be the very first step to build more complex analysis in order to make decisions and built predictions.

So, taking in mind Pandas, as a very powerful and easy to use tool, to face this first approach, it is a great advantage in order to get a quality big picture.

Referencias

- [1]Pandas. <https://pandas.pydata.org/>
- [2]What You Should Know About Consumer Rights in Mexico. <https://mexlaw.com/know-consumer-rights-mexico/>