INSTITUTO PROVINCIAL DE LA ADMINISTRACIÓN PÚBLICA (IPAP)

2022

USO DE NUEVAS TECNOLOGÍAS PARA EL ANÁLISIS DE DATOS EN EL ESTADO

Tomás Barbieri
2021

Clase 3

Objetivos de la clase 3

Segmentar la información, qué información nos parece más relevante que otra. Como podemos hacerle “preguntas” a nuestro conjunto de datos, es decir, filtrar los datos por algunos criterios.

Comprender la transformación de la información.Entender cómo crear a partir de los datos existentes, algunos nuevos, y podemos cambiar el tipo de los mismos, borrar o quitar columnas o filas, etc.

Segmentar la información

Una vez que ya conocemos nuestro dataset, es decir, ya sabemos cuántos datos tenemos, de qué tipo son y cómo están organizados (a grandes rasgos), empezamos a preparar los datos para empezar a sacar conclusiones.

- Obtener un subconjunto de columnas
- Obtener un subconjunto de filas
- Realizar consultas al dataset

Obtener un subconjunto de columnas

Obtener un subconjunto de columnas nos permite trabajar con datos más reducidos, es decir, para ciertos análisis no es necesario toda la información.

Por ejemplo si solo quisiéramos trabajar con las obras y la fecha, podríamos filtrar las columnas correspondientes:

```
dataframe_obras_viales[['nombre_obra','fecha_adjudicacion']]
```

	nombre_obra	fecha_adjudicacion
0	Autopista 24 de marzo	2007-03-13
1	Aeropuerto San Martin	2007-04-13
2	Ruta 22	2009-12-19
3	Ruta Rural 42	2003-07-29
4	Ruta 28	2004-04-06
5	Ruta 2	2007-06-25
6	Autopista 30 de octubre	2008-10-06
7	Ruta 58	2007-01-23
8	Ruta 14	2003-11-09

Obtener un subconjunto de filas

También podemos filtrar un conjunto de filas determinadas, es más complejo el criterio ya que tendría que estar “ordenado” nuestro dataset, pero es una opción valida. En este caso filtramos solamente las filas de la 5 a la 10.

```
[7]  dataframe_obras_viales.loc[5:10,['nombre_obra','fecha_adjudicacion']]
```

	nombre_obra	fecha_adjudicacion
5	Ruta 2	2007-06-25
6	Autopista 30 de octubre	2008-10-06
7	Ruta 58	2007-01-23
8	Ruta 14	2003-11-09
9	Autopista 1 de diciembre	2004-03-17
10	Ruta Rural 420	2008-10-28

Hacerle consultas al dataset

Una de las herramientas que tenemos para obtener esas conclusiones es hacerle consultas al dataset. Pensemos como si le hiciéramos preguntas:

¿Cuáles obras hay en La Plata?

Buscaríamos manualmente en la columna localidad y nos quedamos con las filas que coincidan con La Plata.

Pero veamos un poco cómo esta herramienta nos ayuda a hacerle preguntas al data set.

Hacerle consultas al dataset

Con el comando **query ()** podemos verificar una o más condiciones

```
[10] dataframe_obras_viales.query('localidad_origen == "La Plata"')
```

	nombre_obra	tipo_obra	cant_km	localidad_origen	punto_mapa	fecha_adjudicacion	meses_pautados	monto obra
0	Autopista 24 de marzo	autopista	62	La Plata	-34.9216247017708, -57.95403921946463	2007-03-13	8	85000000
4	Ruta 28	ruta	45	La Plata	-34.96822598862444, -57.88003846002003	2004-04-06	4	48000000
7	Ruta 58	ruta	12	La Plata	-34.97999852966609, -57.931717833242416	2007-01-23	6	12000000
12	Ruta Rural 56	ruta rural	8	La Plata	-34.95775883244286, -58.00415533206554	2004-09-30	9	80000000

Y si le agregamos a la pregunta, que sea una obra de más de 7 meses...

```
[ ] dataframe_obras_viales.query('localidad_origen == "La Plata" & meses_pautados > 7')
```

	nombre_obra	tipo_obra	cant_km	localidad_origen	punto_mapa	fecha_adjudicacion	meses_pautados	monto obra
0	Autopista 24 de marzo	autopista	62	La Plata	-34.9216247017708, -57.95403921946463	2007-03-13	8	85000000
12	Ruta Rural 56	ruta rural	8	La Plata	-34.95775883244286, -58.00415533206554	2004-09-30	9	80000000

Operadores lógicos

Para realizar comparaciones entre valores, tenemos los operadores lógicos

AND &
OR

SENTENCIA 1	SENTENCIA 2	AND	OR
VERDADERO	VERDADERO	VERDADERO	VERDADERO
VERDADERO	FALSO	FALSO	VERDADERO
FALSO	VERDADERO	FALSO	VERDADERO
FALSO	FALSO	FALSO	FALSO

| NOT
~
MAYOR
MENOR
IGUAL

elem1 > elem2 MAYOR O IGUAL elem1 >= elem2

elem1 < elem2 MENOR O IGUAL elem1 <= elem2

elem1 == elem2

Transformación de los datos

Muchas veces necesitamos cambiar los tipos de datos de las variables (columnas) que venían originalmente, generar nuevas columnas en base a las existentes o también algunas de ellas si no son necesarias para nuestro análisis.

Como **transformación de los datos** entendemos a modificar nuestro dataset inicial, veamos algunos ejemplos de lo que podemos hacer.

Transformación de los datos

Por ejemplo el tipo de dato **fecha**, originalmente el procesador de pandas lo reconoce como un texto, pero sería interesante convertirlo a tipo de dato fecha.

Y así obtener del mismo dato otros atributos, como el mes, el año, el dia.

También ocurre que los datos muchas veces están compuestos, es decir, hay más de un dato en una columna, entonces lo que tenemos que hacer es a través de una función. Esta parte suele ser un poco compleja y artesanal, pero lo importante es quedarse con la idea de que se puede modificar el formato de los datos de cada columna.

Transformación de los datos

Transformemos la fecha: con la función **to_datetime**

```
dataframe_obras_viales['fecha_adjudicacion'] = pd.to_datetime(dataframe_obras_viales['fecha_adjudicacion'])
```

```
dataframe_obras_viales.dtypes
```

```
nombre_obra                 object
tipo_obra                   object
cant_km                      int64
localidad_origen            object
punto_mapa                  object
fecha_adjudicacion    datetime64[ns]
meses_pautados               int64
monto obra                   int64
dtype: object
```

ahora tenemos el tipo de dato fecha

Transformación de los datos

Como ahora tenemos un formato de dato fecha, podemos hacerle consultas como por ejemplo, el año.

```
dataframe_obras_viales['fecha_adjudicacion'].dt.year
```

```
0    2007
1    2007
2    2009
3    2003
4    2004
5    2007
6    2008
7    2007
8    2003
```

ahora veremos cómo podemos añadir ese año en una nueva columna

Creación de columnas

Creación de columnas en base a otras: en este caso podríamos crear nuevas columnas (otra forma de ver los datos) y que nos sirva para el análisis.

Continuando con el tema de la fecha. El formato de la fecha de la columna es (día-mes-año) y nosotros queremos tener en otra columna solamente el año.

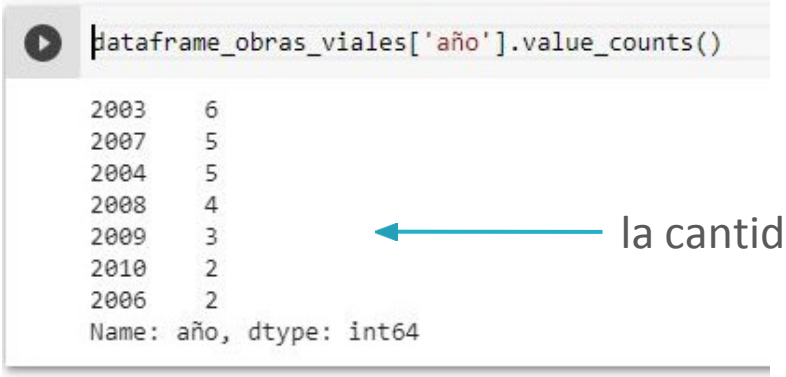
columna nueva

```
dataframe_obras_viales['año'] = dataframe_obras_viales['fecha_adjudicacion'].dt.year
dataframe_obras_viales.head(3)
```

	nombre_obra	tipo_obra	cant_km	localidad_origen	punto_mapa	fecha_adjudicacion	meses_pautados	monto obra	año
0	Autopista 24 de marzo	autopista	62	La Plata	-34.9216247017708, -57.95403921946463	2007-03-13	8	85000000	2007
1	Aeropuerto San Martin	aeropuerto	4	Junin	-34.632090156133394, -60.92658792156374	2007-04-13	14	140025000	2007
2	Ruta 22	ruta	80	Pehuajo	-35.70219477411215, -61.52857252912656	2009-12-19	9	45000000	2009

Creación de columnas

Y ahora podríamos totalizar los valores por año y saber rápidamente cuántas obras se ejecutaron en un año determinado.



Borrar elementos

También se pueden borrar columnas o filas que no se deseen. Una buena práctica por ejemplo es generar copias de un conjunto de datos y ahi hacer este tipo de operaciones.

Con el comando **drop** se pueden borrar columnas o filas, en este caso, borre la columna que había agregado anteriormente.

```
[19] dataframe_obras_viales.drop('año', axis=1, inplace=True)
```

```
[20] dataframe_obras_viales.head(3)
```

	nombre_obra	tipo_obra	cant_km	localidad_origen	punto_mapa	fecha_adjudicacion	meses_pautados	monto obra
0	Autopista 24 de marzo	autopista	62	La Plata	-34.9216247017708, -57.95403921946463	2007-03-13	8	85000000
1	Aeropuerto San Martin	aeropuerto	4	Junin	-34.632090156133394, -60.92658792156374	2007-04-13	14	140025000
2	Ruta 22	ruta	80	Pehuajo	-35.70219477411215, -61.52857252912656	2009-12-19	9	45000000

Links de interés

- Seleccionar filas y columnas
- Query
- Operadores lógicos
- Agregando nuevas columnas
- Transformación de datos

Pregunta/Debate

Para comentar en el Foro:

¿Qué piensan sobre la versatilidad de esta herramienta?

¿Les resulta complejo a simple vista?

ipap.gba.gob.ar