

---

# **Geoprocesamiento en python y qgis Documentation**

***Versión 1.0***

**Víctor Hernández**

**19 de marzo de 2021**



---

## Guías de uso:

---

<b>1. ¿Cómo ejecutar un código en Qgis?</b>	<b>1</b>
<b>2. Librería de funciones para el análisis espacial multicriterio</b>	<b>5</b>
<b>3. Análisis de sensibilidad</b>	<b>7</b>
<b>4. Índice Lee-Sallee</b>	<b>11</b>
<b>5. OWA</b>	<b>13</b>
<b>6. Verificación de capas</b>	<b>19</b>
<b>7. Tabular intersección entre 3 geometrías</b>	<b>23</b>
<b>8. Cobertura por niveles areas</b>	<b>29</b>
<b>9. Cobertura de uso y tipo de suelo a nivel municipal</b>	<b>33</b>

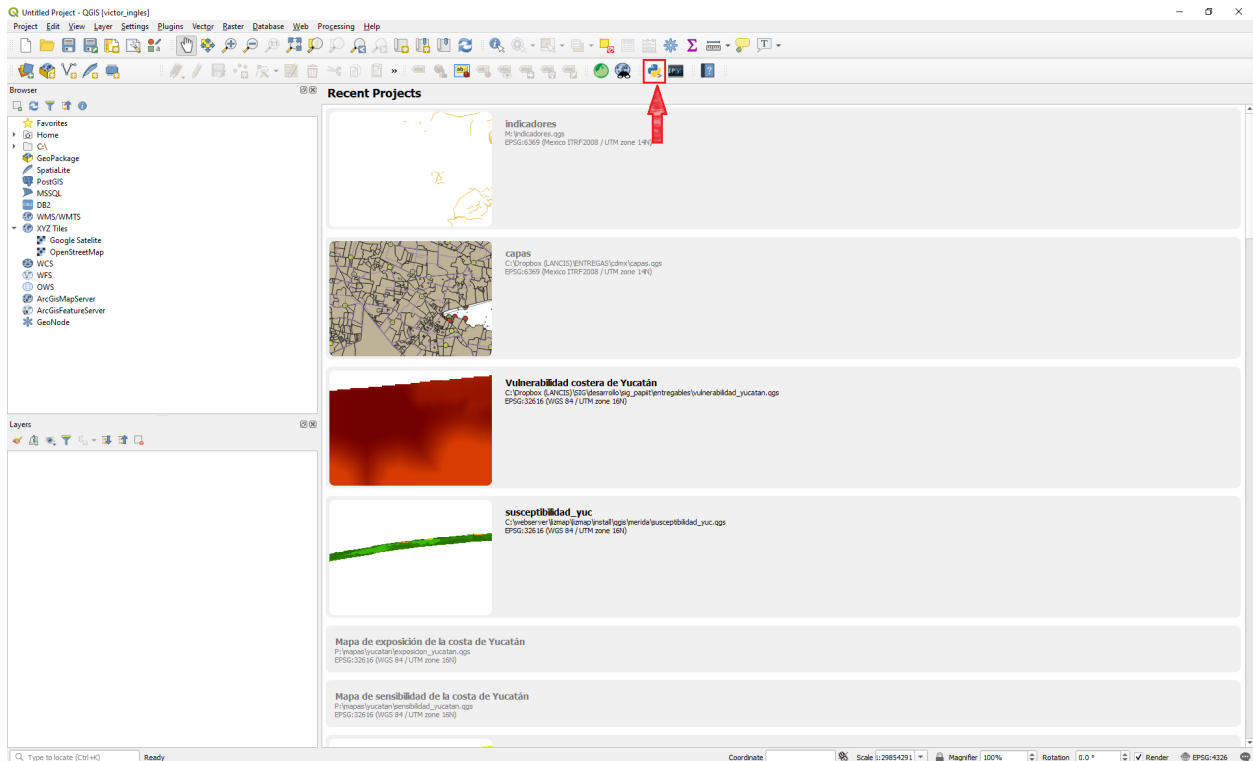


# CAPÍTULO 1

## ¿Cómo ejecutar un código en Qgis?

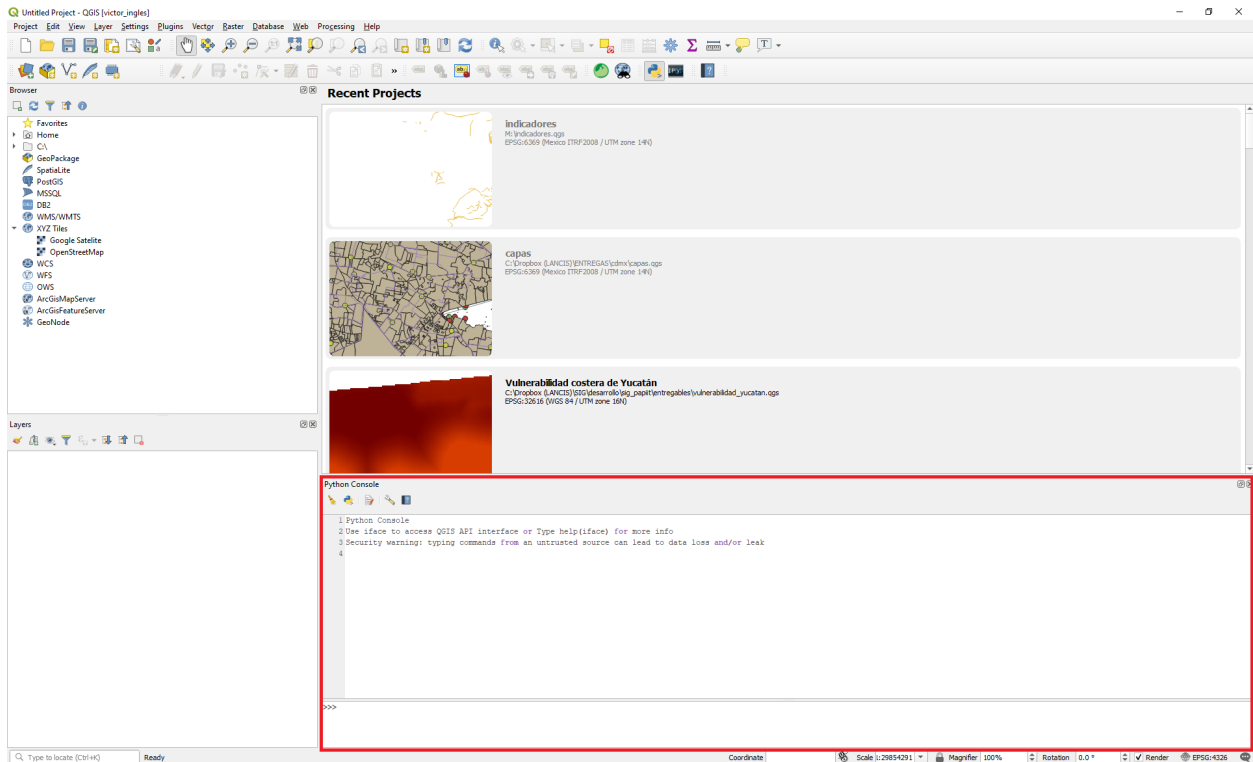
### 1.1 Paso #1

Ejecutar Qgis Desktop 3.XX, la ventana que se muestra es la que corresponde a la interfaz gráfica del programa, en la barra de tareas hacer clic en el ícono correspondiente a **python** para abrir la consola



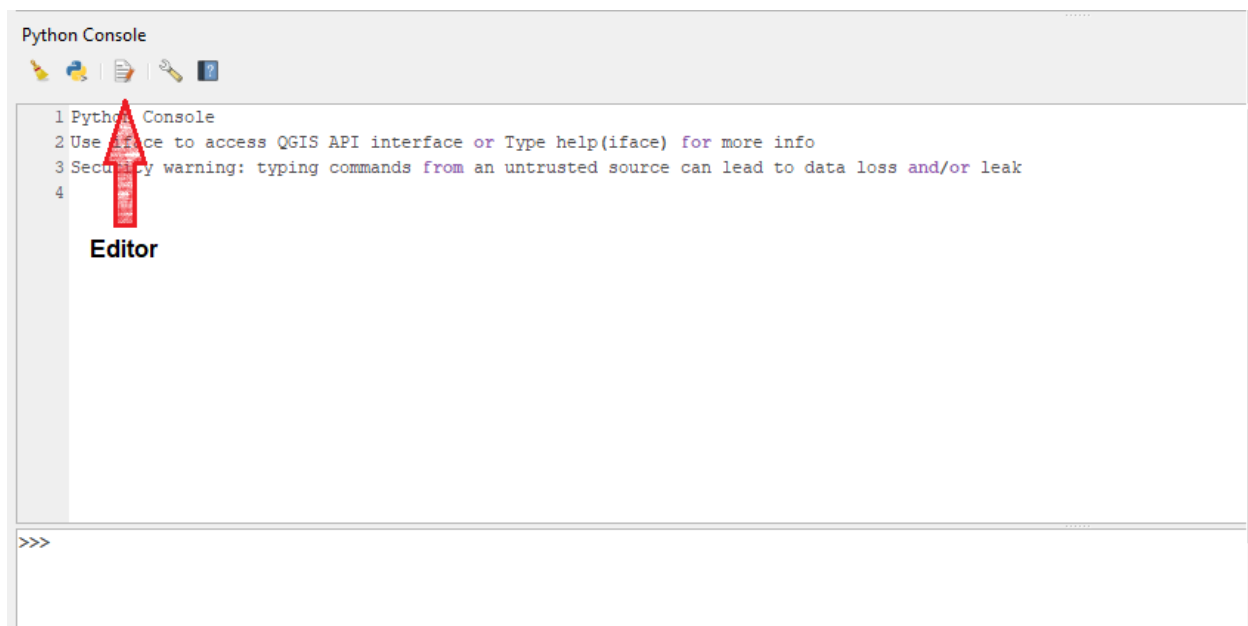
## 1.2 paso #2

En la parte inferior de la ventana se mostrará la consola de python



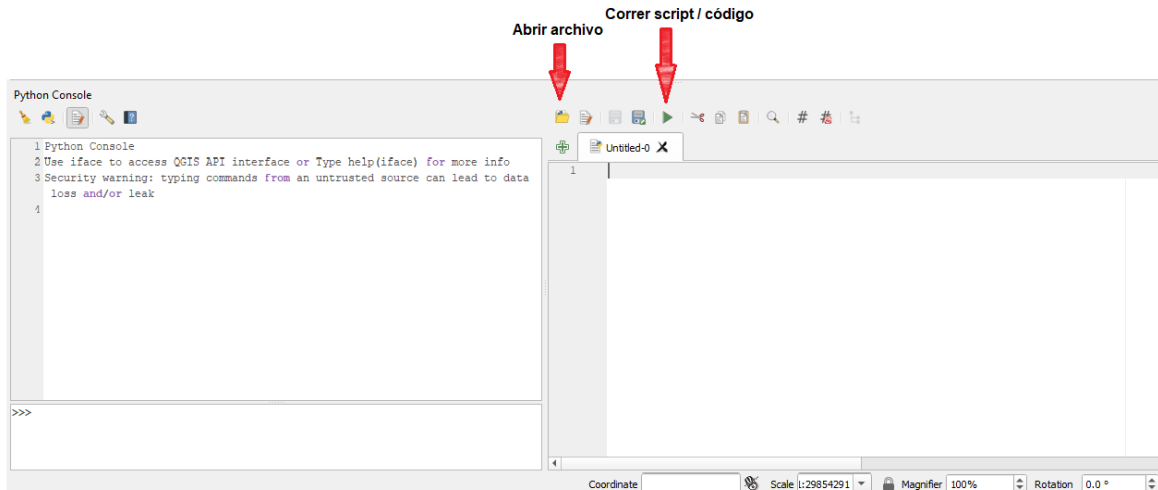
## 1.3 paso #3

Hacer clic en el ícono de **Editor**



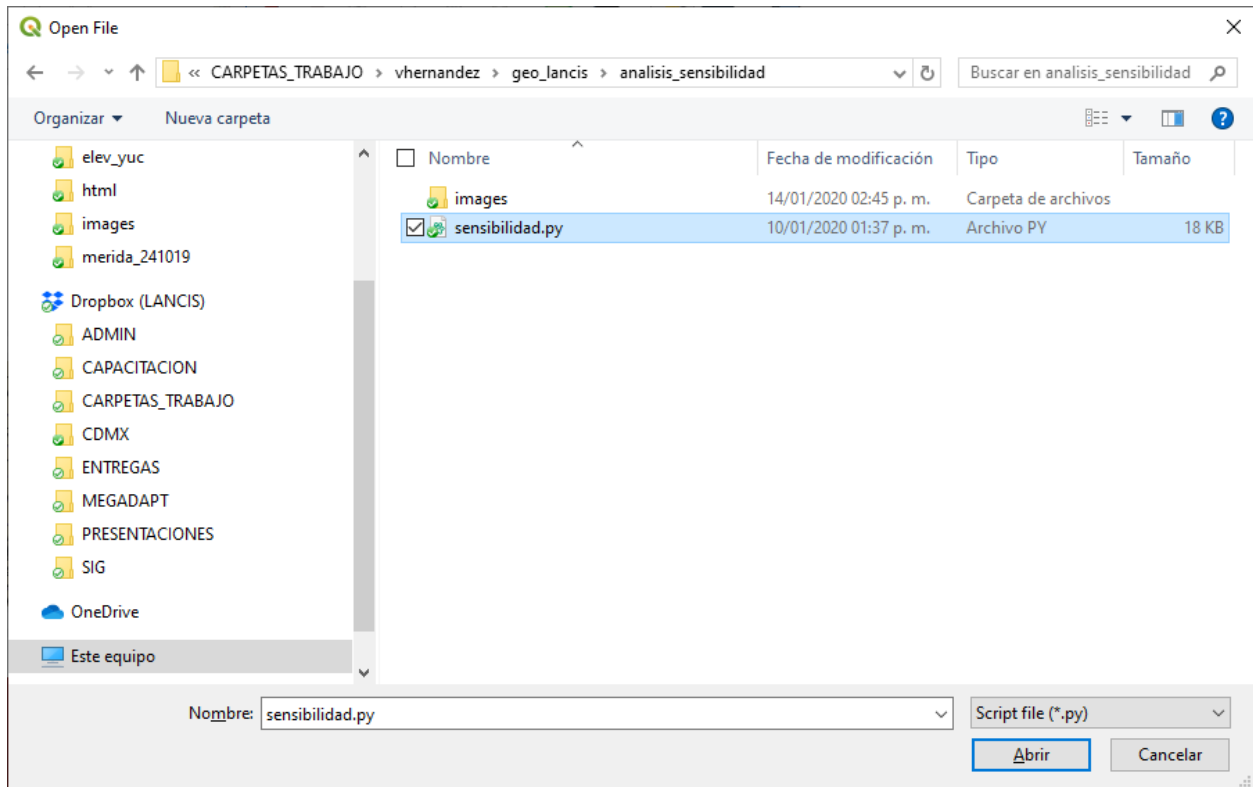
## 1.4 paso #4

Se despliega en el lado izquierdo un panel, el cual es el editor de código, cuenta con una barra de tareas, para abrir un script dar clic en el ícono de **abrir archivo**



## 1.5 paso #5

Se abrirá una ventana que te permite usar el explorador de archivos para navegar y encontrar el archivo **.py**, elegir el script deseado y dar clic en abrir.





---

### Librería de funciones para el análisis espacial multicriterio

---

En esta página encontrarás la documentación de las funciones que he creado en python y qgis 3.10 o superior

#### 2.1 Requerimientos generales

Para asegurar la ejecución correcta del código es importante verificar la instalación y funcionamiento de los siguientes elementos:

- Qgis 3.10 o superior con Grass y librerías de Osgeo4W
- Librerías python:
  - Numpy
  - Pandas
  - GDAL
  - reduce
  - os
  - copy
  - pprint
  - string

#### 2.2 Descarga la librería

`apcsig.py`.

#### 2.3 Documentación



---

### Análisis de sensibilidad

---

La prueba de sensibilidad por remoción de capas mide la importancia de cada mapa que se utiliza en un índice cartográfico como el que resulta de la aplicación de la combinación lineal ponderada.

Descargar el código de ejemplo

`sensibilidad.py`.

### 3.1 Requerimientos generales

Para asegurar la ejecución correcta del código es importante verificar la instalación y funcionamiento de los siguientes elementos:

- Qgis 3.4 o superior y librerías de Osgeo4W
- Librerías python:
  - copy
  - pprint
  - string
  - osgeo/gdal
  - gdal\_calc
  - os

### 3.2 Requerimientos generales de los insumos

Es importante que todas las capas raster cumplan con las siguientes condiciones:

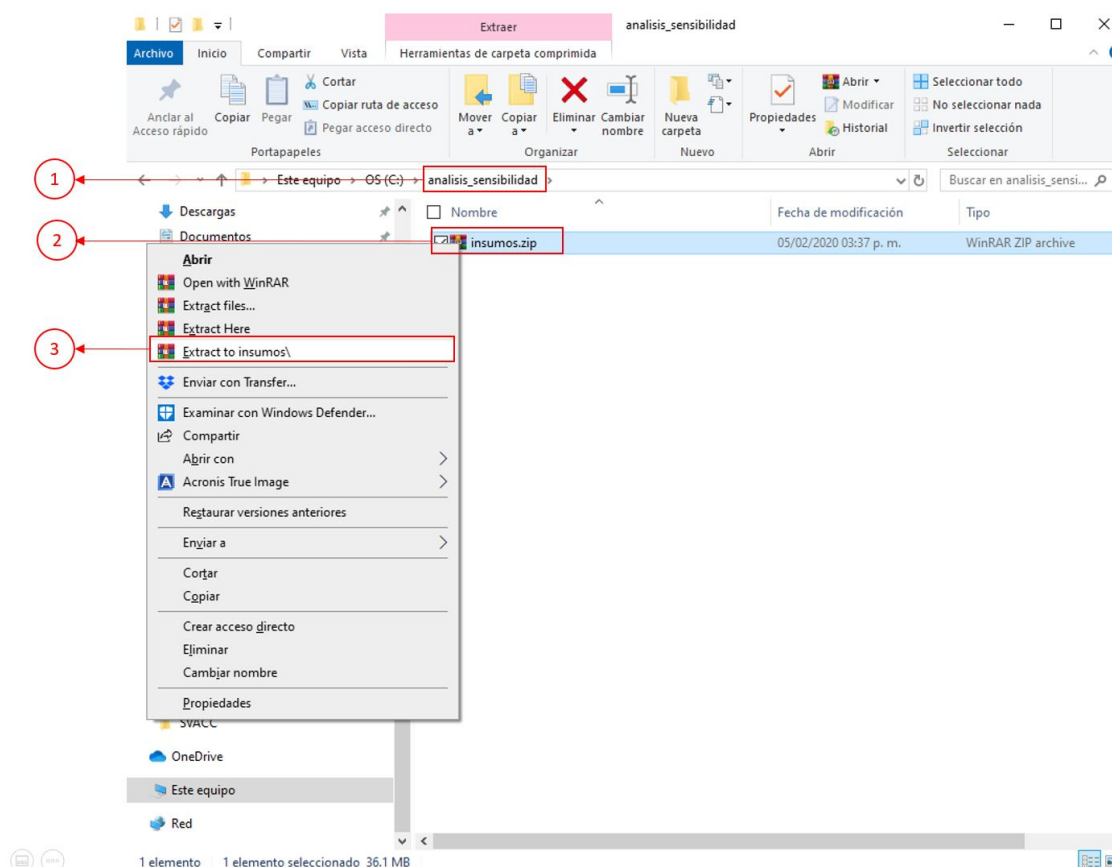
- Misma proyección cartográfica
- Mismo tamaño de pixel

- Misma extensión de capa
- Mismo valor de NoData

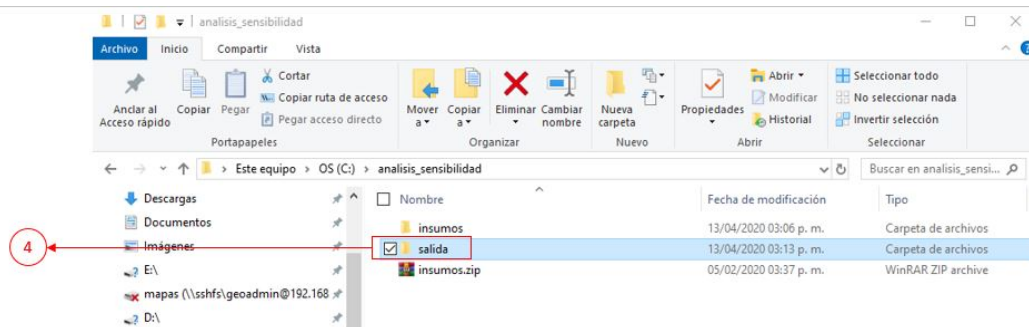
## 3.3 Ejemplo

### 3.3.1 Insumos

Crear una (1) carpeta en el directorio raíz o en la unidad C que se llame **analisis\_sensibilidad**, para descargar los insumos hacer clic [aquí](#) (2) guarde el archivo **insumos.zip** en la carpeta **analisis\_sensibilidad**, posteriormente hacer clic derecho sobre el archivo y elegir la opción (3) **Extract to insumos\**



Una vez terminado el proceso, crear en la carpeta **analisis\_sensibilidad** una (4) carpeta con el nombre **salida**



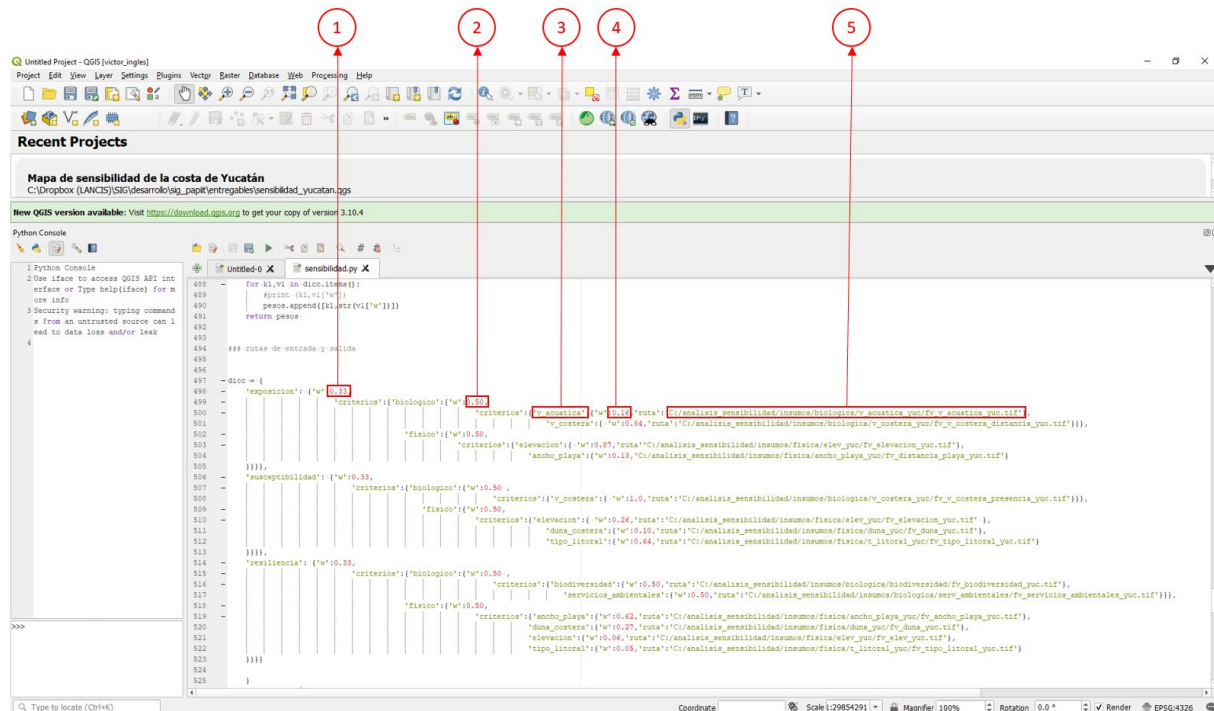
### 3.3.2 Procedimiento

#### 1. Abrir el código

Abrir el código **sensibilidad.py** en Qgis 3.4 o superior, Para resolver cualquier duda al respecto, consultar la [guía](#)

#### 2. Actualizar el diccionario

Ingresar la (1) ponderación del componente según corresponda (Exposición, Susceptibilidad, Resiliencia), posteriormente ingresar la (2)ponderación del subcomponente (biológico,físico), Ingresar el (3)nombre de la capa raster de entrada con su respectiva (4)ponderación y su (5)ruta repita los pasos siguiendo la estructura y hasta ingresar cada una de las capas.



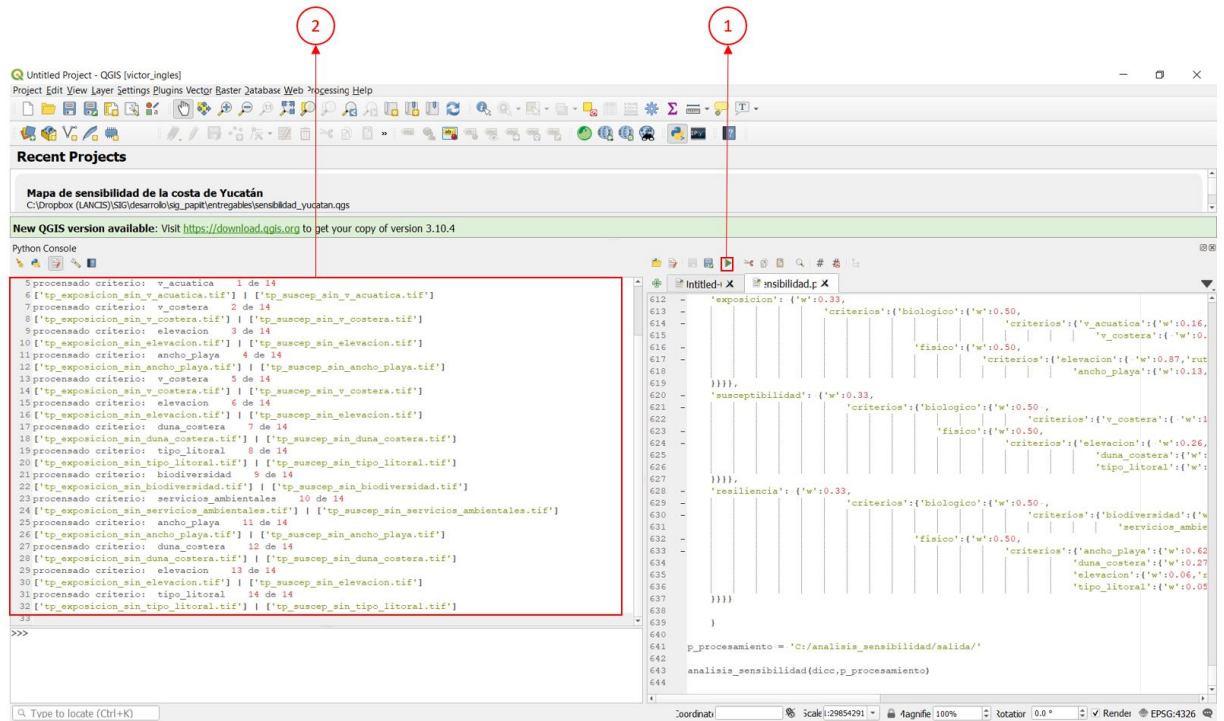
#### 3. Indicar el directorio de salida

Indicar el directorio donde guardarán los archivos necesarios para realizar el análisis de sensibilidad y el archivo **analisis\_sensibilidad.csv** que contendrá los resultados.

```
p_procesamiento = 'C:/analisis_sensibilidad/salida/'
```

#### 4. Ejecutar el código

hacer clic en el (1) botón de ejecutar código, puede demorar 10 minutos o más dependiendo el procesador y memoria RAM que tenga el equipo en donde se ejecute, al concluir aparecerá en la (2) consola una lista que indica que ha procesado cada una de las capas.



### 3.3.3 Bibliografía

### 3.3.4 Documentación dentro del código

Para capas vectoriales

$$lee\_sallee\_index = \frac{A \cap B}{A \cup B}$$

descarga el código de ejemplo

`indice_lee_sallee.py`.

## 4.1 Ejemplo

### 4.1.1 Insumos

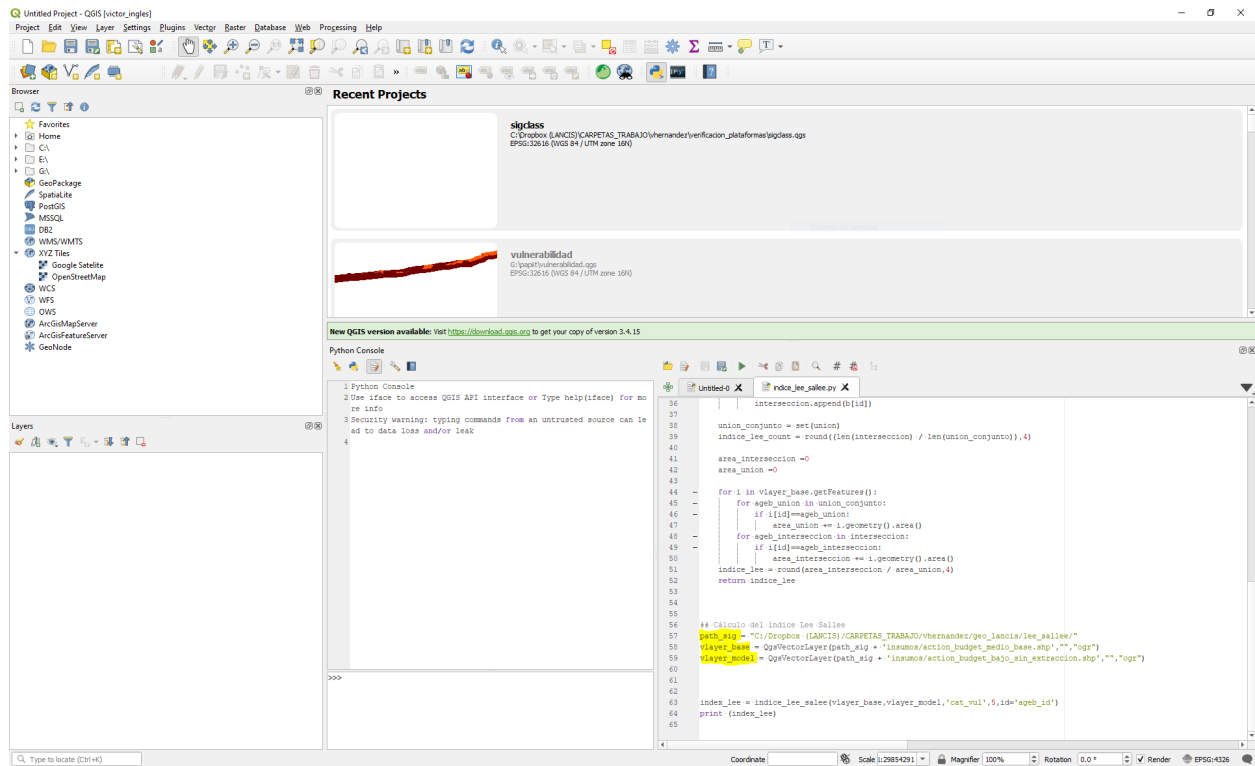
Descarga los insumos para este ejemplo [aquí](#)

### 4.1.2 Procedimiento

Abre el código **indice\_lee\_sallee.py** en Qgis 3.4 o superior, Si tienes dudas de como hacerlo visualiza la [guía](#)

Modifica las rutas donde se encuentran los insumos

- `vlayer_base` es la capa vectorial que se ocupa como base
- `vlayer_model` es la capa vectorial que se ocupa como modelo





OWA (Ordered Weighted Average) es un análisis de aptitud territorial basado en procedimientos de Sistemas de Información Geográfica (SIG) y evaluación multicriterio (Malczewski, 2006). El análisis OWA genera un amplio rango de escenarios de aptitud territorial cambiando únicamente un parámetro lingüístico (alpha), relacionado con la rigidez en el cumplimiento de criterios preestablecidos.

OWA está definido por la siguiente ecuación:

$$OWA = \sum_{j=1}^n \left( \left( \sum_{k=1}^j u_k \right)^{\alpha} - \left( \sum_{k=1}^{j-1} u_k \right)^{\alpha} \right) z_{ij}$$

Donde:

j = Criterio

uk = Peso ordenado del criterio j

k= Orden asignado al peso del criterio j (renglón)

i = Pixel

z<sub>ij</sub> = Valor ordenado del criterio j en el pixel i

α = Cuantificador lingüístico

Descargar el código de ejemplo

`owa_raster.py`.

## 5.1 Requerimientos generales

Para asegurar la ejecución correcta del código es importante verificar la instalación y funcionamiento de los siguientes elementos:

- Qgis 3.4 o superior y librerías de Osgeo4W
- Librerías python:

- Numpy
- Pandas
- GDAL
- reduce

## 5.2 Requerimientos generales de los insumos

Es importante que todas las capas raster cumplan con las siguientes condiciones:

- Misma proyección cartográfica
- Mismo tamaño de pixel
- Misma extensión de capa
- Mismo valor de NoData

## 5.3 Ejemplo

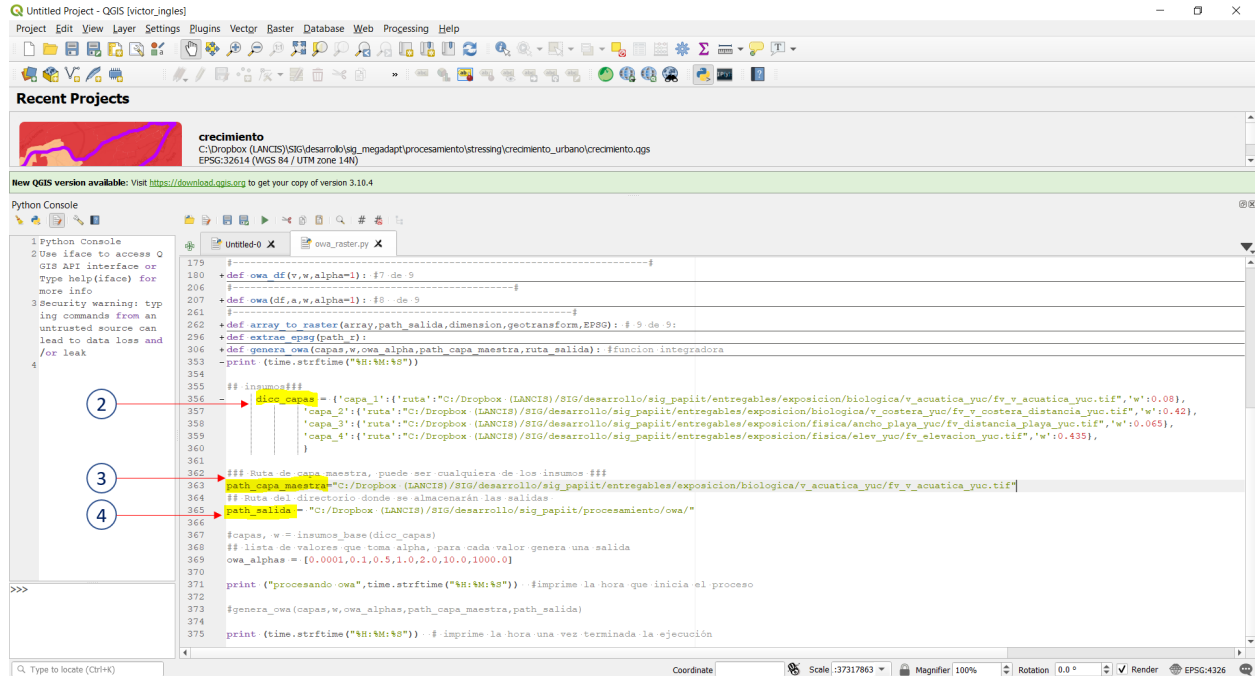
### 5.3.1 Insumos

Descargar los insumos para este ejemplo [aquí](#)

### 5.3.2 Procedimiento

#### 1. Abrir el código

Abrir el código `owa_raster.py` en Qgis 3.4 o superior, Para resolver cualquier duda al respecto, consultar la [guía](#)



## 2. Actualizar el diccionario

Ingresar las capas raster de entrada con sus respectivos pesos a la función mediante un diccionario. Es importante seguir la estructura del siguiente ejemplo:

```
dicc_capas = {'capa_1':{'ruta':'C:/Dropbox (LANCIS)/SIG/desarrollo/sig_papiit/entregables/exposicion/biologica/v_acuatica_yuc/fv_v_acuatica_yuc.tif','w':0.08},
              'capa_2':{'ruta':'C:/Dropbox (LANCIS)/SIG/desarrollo/sig_papiit/entregables/exposicion/biologica/v_costera_yuc/fv_v_costera_distancia_yuc.tif','w':0.42},
              'capa_3':{'ruta':'C:/Dropbox (LANCIS)/SIG/desarrollo/sig_papiit/entregables/exposicion/fisica/ancha_playa_yuc/fv_distancia_playa_yuc.tif','w':0.065},
              'capa_4':{'ruta':'C:/Dropbox (LANCIS)/SIG/desarrollo/sig_papiit/entregables/exposicion/fisica/elev_yuc/fv_elevacion_yuc.tif','w':0.435},
              }
```

Donde:

- **capa\_#**: Corresponde a la capa en el orden en que se agregó al diccionario,
- **ruta** : Corresponde a la ruta o path de la capa
- **w** : Corresponde al peso asociado a esa capa o criterio

**Nota:** Para adicionar una capa, agregar el consecutivo a la llave de la capa (en este caso capa\_5). La línea quedaría de la siguiente forma:

```
“capa_5”:{“ruta”:path_tiff,”w”:#.###}, }
```

### 3. Indicar la capa maestra

Para generar la salida en formato tiff se requiere conocer aspectos técnicos como número de columnas y renglones, tamaño de pixel, coordenadas del extent, entre otros.

Estos datos son extraídos por el código mediante la variable **path\_capa\_maestra**, en ella, se indica la ruta de **cualquier** capa raster ingresada en el diccionario del paso #2.

como ejemplo se toma la ruta de la **capa\_1**

```
path_capa_maestra = "C:/Dropbox (LANCIS)/SIG/desarrollo/sig_papiit/entregables/  
→exposicion/biologica/v_acuatica_yuc/fv_v_acuatica_yuc.tif"
```

### 4. Indicar el directorio de salida

Indicar el directorio donde guardarán los mapas de salida.

por ejemplo:

```
path_salida = "C:/Dropbox (LANCIS)/SIG/desarrollo/sig_papiit/procesamiento/owa/"
```

### 5. Los valores de alpha

El código tiene valores predeterminados de alpha

---

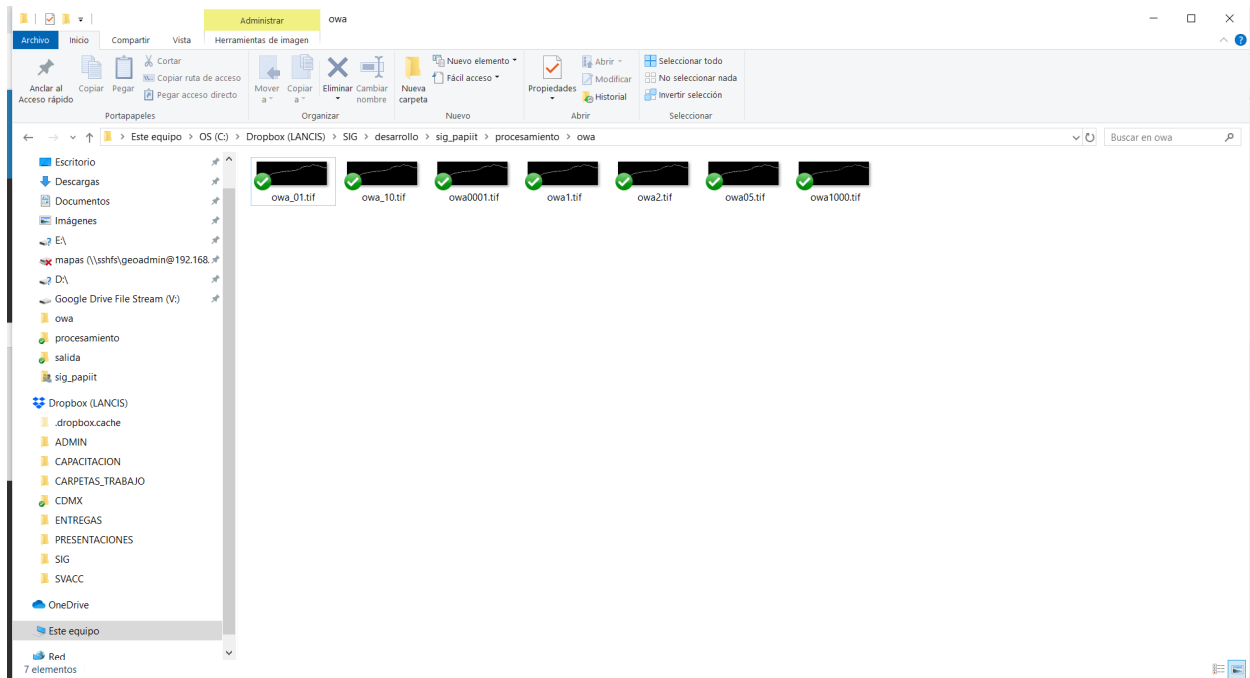
**Nota:** Para más información respecto a los valores de alpha consulte la bibliografía

---

```
owa_alphas = [0.0001, 0.1, 0.5, 1.0, 2.0, 10.0, 1000.0]
```

$\alpha$	Quantifier (Q)
0.0001	At least one
0.1	At least a few a a
0.5	A few
1.0	Half (identity)
2.0	Most
10.0	Almost all
1000	All

para cada valor en la lista, el código generará un mapa en el directorio de salida



## 5.4 Bibliografía

Malczewski, J. (2006). Ordered weighted averaging with fuzzy quantifiers: GIS-based multicriteria evaluation for land-use suitability analysis. *International Journal of Applied Earth Observation and Geoinformation*, 8, 270-277.

## 5.5 Documentación dentro del código



---

### Verificación de capas

---

Los puntos a verificar son los siguientes:

1. Proyección Que exista el archivo prj asociado
2. Geometría completa Que exista los mismos elementos geométricos que los contenidos en la tabla de atributos
3. Sobrelapados Que la capa no cuente con errores topológicos
4. Nulos Que no existan campos vacios en la tabla de atributos
5. Codificados Que no existan campos con caracteres espeaciales o extraños en su contenido o que datos numéricos esten declarados como texto
6. Metadatos Que exista el archivo xml asociado a los metadatos geográficos

Descargar el código de ejemplo

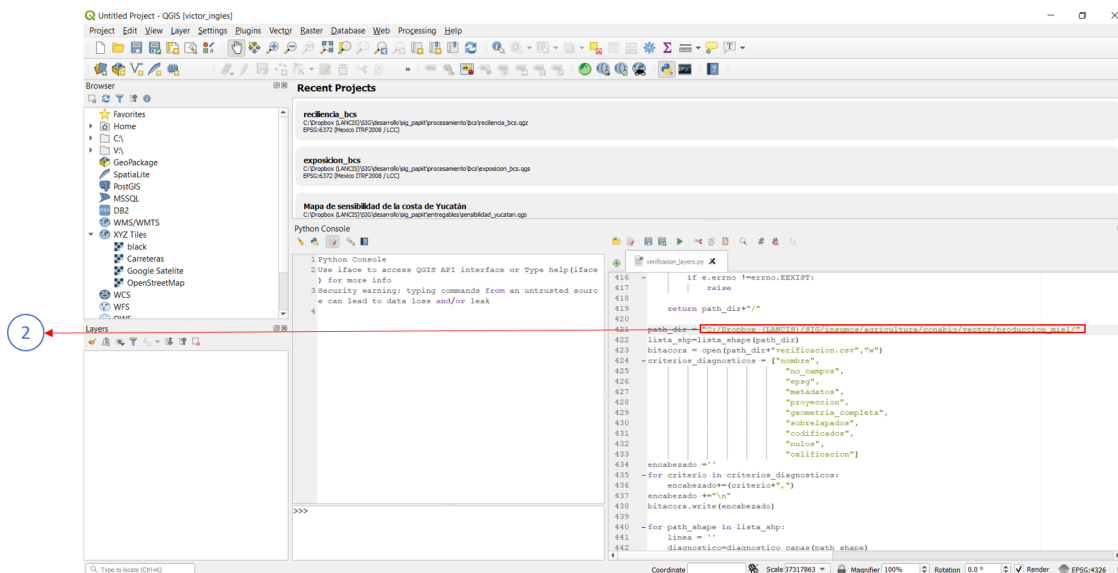
`verificacion_layers.py`.

## 6.1 Requerimientos generales

## 6.2 Ejemplo:

### 6.2.1 1. Abrir el código

Abrir el código **verificación\_layers.py** en Qgis 3.4 o superior, para resolver cualquier duda al respecto, consultar la [guía](#)



## 6.2.2 2. Indicar el directorio

El código verifica todas las capas vectoriales contenidas en el directorio indicado en la variable **path\_dir**

```
path_dir = "C:/Dropbox (LANCIS)/SIG/insumos/agricultura/conabio/vector/produccion_
miel/"
```

## 6.2.3 Salidas

### Capas de topología

el código crea una carpeta llamada **temp**, dentro de ella otra carpeta llamada **topologia** en esta carpeta se guardan los 3 archivos shapefile resultantes de la función **topologia**

están nombrados de la siguiente manera:

#### **nombrecapa\_error.shp**

Capa de puntos que indica la posición en donde 2 o más polígonos no colindan adecuadamente o que su geometría puede causar problemas en algún análisis espacial

#### **nombrecapa\_invalido.shp**

Capa de polígonos que indica geometría inválida de la capa

#### **nombrecapa\_valido.shp**

Capa de polígonos que indica la geometría válida de la capa

### Imagen de la capa



---

**Nota:** El código genera una imagen de la capa con el mapa base de openstreetmap sí y solo sí la capa tiene un puntaje de 10 en el criterio de **proyección**

---



---

## Tabular intersección entre 3 geometrías

---

Calcula la intersección de 3 capas y realiza una tabulación cruzada del área

La capa A es al nivel geográfico que se reporta La capa B es el nivel geométrico intermedio del cual se cuantifica el área total perteneciente a la entidad A y se cuantifica el área por clase de USV. La capa C es la serie de uso de suelo y vegetación de INEGI

### 7.1 Requerimientos generales

Para asegurar la ejecución correcta del código es importante verificar la instalación y funcionamiento de los siguientes elementos:

- Qgis 3.4 o superior

Descargar el código de ejemplo

`tabulacion_3geo.py`.

### 7.2 Requerimientos generales de los insumos

Es importante que todas las capas vectoriales cumplan con las siguientes condiciones:

- Misma proyección cartográfica
- Sin problemas topológicos

#### 7.2.1 Entidad A

- Que cuente con un campo que contenga un identificador único para cada geometría

## 7.2.2 Entidad B

- Que cuente con un campo dónde se especifiquen las categorías o clases

## 7.2.3 Entidad C

- Que cuente con un campo que contenga el número de clase correspondiente
- Que el campo mencionado en el inciso anterior se llame igual para todas las series

## 7.3 Ejemplo

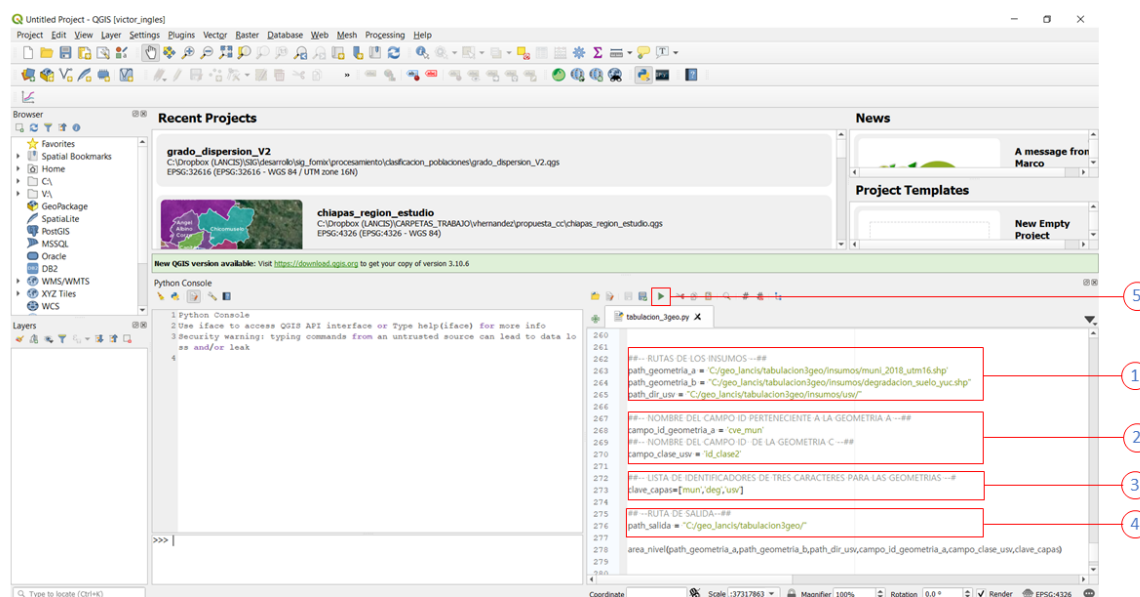
### 7.3.1 Datos de prueba

Descargar los datos de prueba para este ejemplo [aquí](#)

insumo	Descripción
mu-ni_2018_utm16.shp	Capa de municipios del estado de Yucatán, esta capa representa la entidad A
degradacion_suelo_yuc.shp	Capa de áreas de degradación del suelo clasificadas en ligero, moderado, alto y extremo, esta capa representa la entidad B
usv/	Directorio que contiene capas de USV de las 6 series de INEGI para el estado de Yucatán, estas capas representan la entidad C

### Abrir el código

Abrir el código **tabulacion\_3geo.py** en Qgis 3.4 o superior, Para resolver cualquier duda al respecto, consultar la [guia](#)



## 1. Indicar la ruta de los insumos

Indicar la ruta completa de los insumos según corresponda:

**Advertencia:** verifica que se use “/” como separador de espacio en lugar de “\”

```
path_geometria_a = 'C:/geo_lancis/tabulacion3geo/insumos/muni_2018_utml6.shp'
path_geometria_b = "C:/geo_lancis/tabulacion3geo/insumos/degradacion_suelo_yuc.shp"
path_dir_usv = "C:/geo_lancis/tabulacion3geo/insumos/usv/"
```

## 2. Indicar los nombres de los campos id

Para la geometría A se declara el nombre del campo identificador o clave

```
campo_id_geometria_a = 'cve_mun'
```

**Nota:** El nombre del campo para la geometría B se preguntará más adelante por medio de una ventana emergente. ver paso 6

Para la geometría C se declara el nombre del campo el cual contiene el identificador de clase para las series de USV

```
campo_clase_usv = 'id_clase2'
```

## 4. Agrega identificadores según el tipo de geometría

En la variable **clave\_capas** agragar un identificador para cada una de las categorías de tres caracteres.

```
clave_capas=['mun', 'deg', 'usv']
```

## 4. Indica el directorio de salida

En esta ruta se escribirá la tabla resultado de la cruce adicionalmente se crea una carpeta llamada **tmp** en la cual se almacenarán las cruces realizadas en el proceso.

```
path_salida = "C:/geo_lancis/tabulacion3geo/"
```

## 5. Ejecuta el script

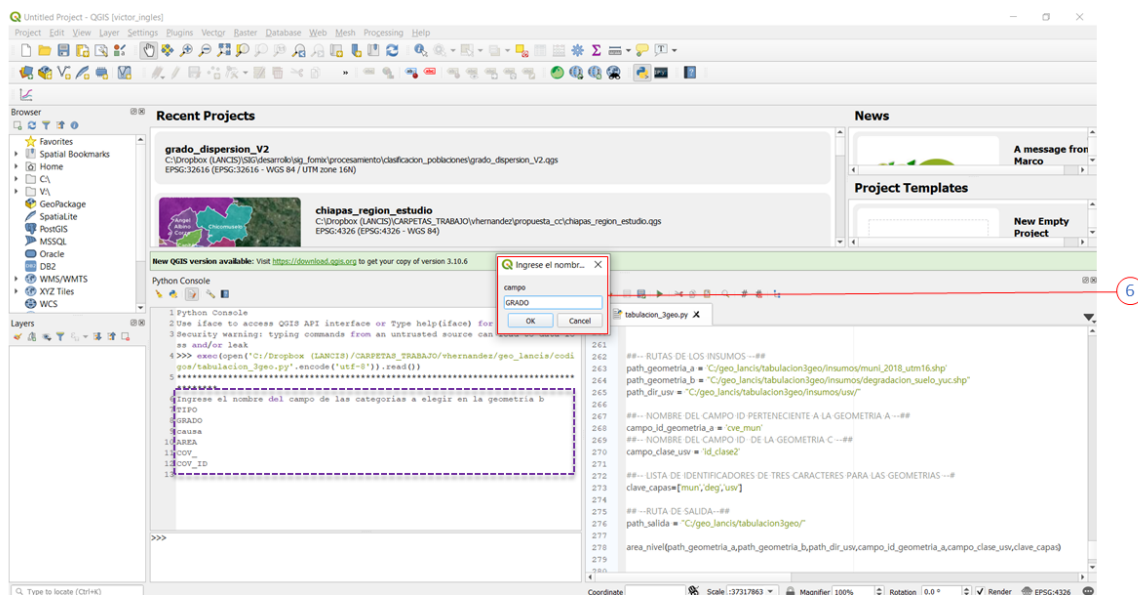
Hacer clic en el botón de ejecutar y permanece atento a la consola.

## 6. Ingresa el nombre del campo identificador de la geometría B

Se mostrará en la consola la siguiente instrucción:

*Ingresa el nombre del campo de las categorías a elegir en la geometría B*

enseguida del nombre de todos los campos que tiene la capa, escriba en la ventana de texto el nombre del campo tal cual se muestra en la consola, para este ejemplo el campo que sirve como identificador es **GRADO** una vez escrito hacer clic en **OK**



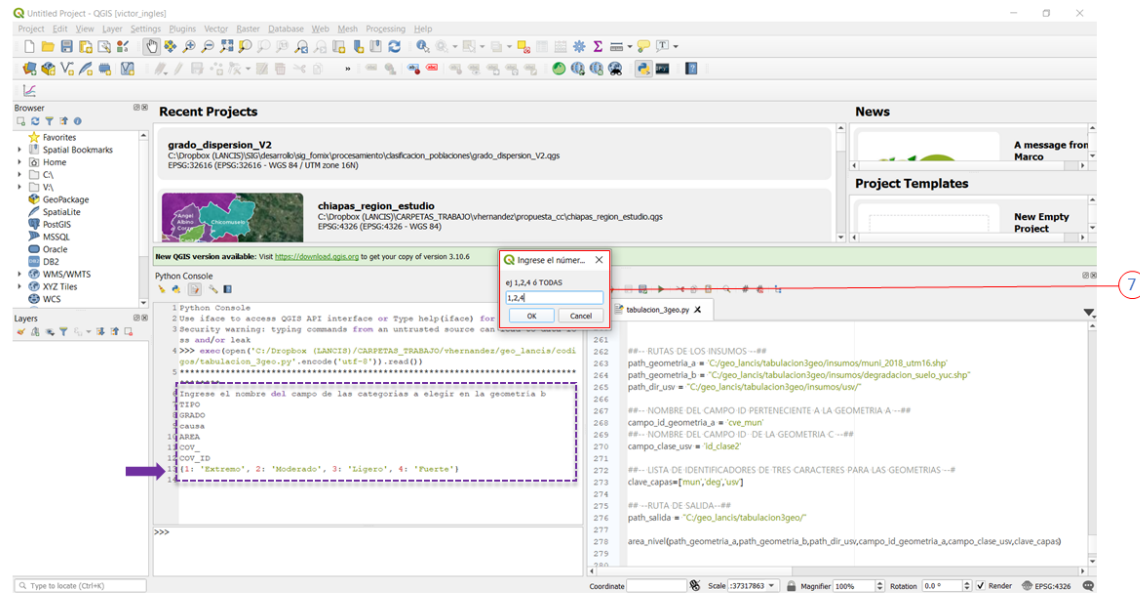
### 7. Ingresar el número de las categorías a considerar

El script permite ingresar todas las categorías o solo algunas, las categorías indicadas serán reclasificadas como binario donde 1 es que fue considerada y 0 cero que no, posteriormente realiza una limpieza de la capa eliminando las geometrías con la categoría 0, una vez finalizado ese proceso, realiza la cruza solo con las categorías consideradas.

Ingrese las categorías que desea considerar conforme se muestra en la consola, para este ejemplo solo consideramos las categorías “Moderado”, “Fuerte” y “Extremo”, por lo cual en la ventana que se muestra se escribe: 1,2,4, una vez escrito dar clic en **OK**

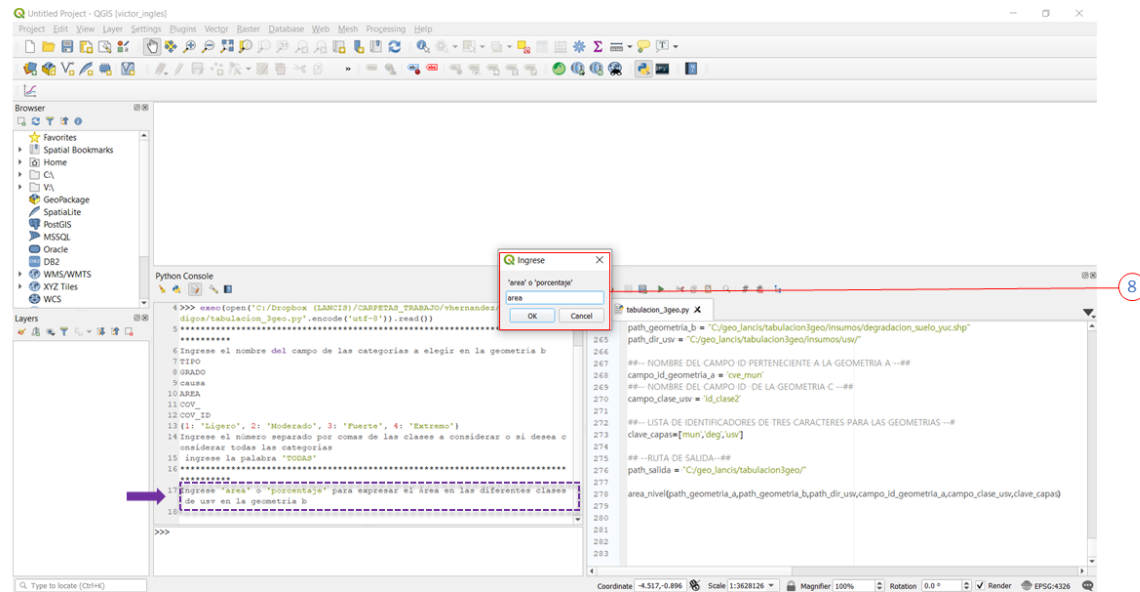
**Nota:** puede ingresar solo una categoría escribiendo por ejemplo: 1,

Puede ingresar todas las categorías escribiendo la palabra «TODAS»



## 8.- Elegir entre área o porcentaje

La cuantificación de las categorías de USV en el área de la geometría B perteneciente a una identidad de la geometría A, puede expresarse en área (hectáreas) o en porcentaje. para este ejemplo se elige “area”







---

### Cobertura por niveles areas

---

El objetivo de esta herramienta es cuantificar el área según el tipo de clase de uso de suelo vegetación asociado a una geometría intermedia.

para comprender mejor lo anterior se expresa el siguiente caso

se quiere conocer a nivel de municipio (geometría A), el espacio designado como **Área Natural Protegida** (geometría B), en dicho espacio, se requiere cuantificar el tipo de cobertura por clase de la serie de uso de suelo y vegetación de INEGI (Geometría C).

Por lo tanto, se tendrá como resultado una capa geografica a nivel municipio que en su tabla de atributos cuente con:

- Todos los campos de la capa original de municipios
- un campo del área total (expresado en hectáreas) del espacio designado como **Área Natural Protegida** perteneciente al municipio,
- campos nombrados como **clase\_#** donde # corresponde al número de clase o cobertura asociado a la capa de uso de suelo y vegetación. Estos campos pueden contener el área por clase expresada en hectáreas, o bien, el porcentaje correspondiente al área total del espacio designado como **Área Natural Protegida** perteneciente al municipio.

### 8.1 Requerimientos generales

Para asegurar la ejecución correcta del código es importante verificar la instalación y funcionamiento de los siguientes elementos:

- Qgis 3.4 o superior

### 8.2 Requerimientos generales de los insumos

Es importante que todas las capas vectoriales cumplan con las siguientes condiciones:

- Misma proyección UTM

- Sin problemas topológicos

### 8.2.1 Geometría A

- Tener un campo que contenga un identificador único para cada geometría (puede ser de tipo texto o entero)

### 8.2.2 Geometría B

- Tener un campo que contenga las diferentes categorías o tipos de geometria (puede ser de tipo texto o entero)

### 8.2.3 Geometría C

- Tener el siguiente tipo de nombrado

**usv\_serie#\_aaa.shp**

Donde:

- **#** representa el número de serie (1,2,3,4,5 o 6) **obligatorio**
- **\_aaa** representa una abreviatura del lugar, para este ejemplo se ocupa **\_yuc**
- Tener un campo de tipo entero que contenga las diferentes clases de cobertura empezando por 1, este campo debe estar presente en todas las capas de USV y debe llamarse de la misma forma

## 8.3 Ejemplo

### 8.3.1 Insumos

Descargar los insumos para este ejemplo [aquí](#)

#### 1. Abrir el código

Abrir el código **owa\_raster.py** en Qgis 3.4 o superior, Para resolver cualquier duda al respecto, consultar la [guía](#)

```
path_geometria_a = 'C:/Dropbox (LANCIS)/SIG/desarrollo/sig_fomix/entregables/mun_
↳region/muni_2018_utm16.shp'
```

Se ingresa la ruta completa de la capa de **Áreas Naturales Protegidas** en la variable **path\_geometria\_b**

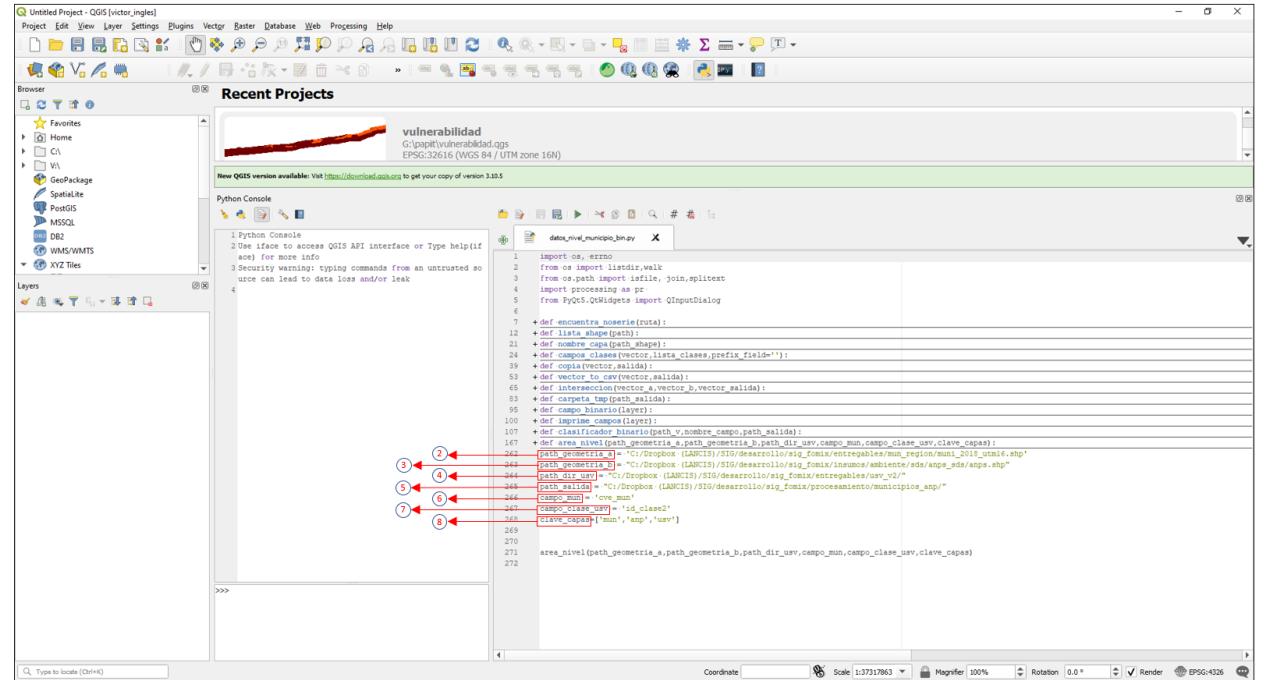
```
path_geometria_b = "C:/Dropbox (LANCIS)/SIG/desarrollo/sig_fomix/insumos/ambiente/sds/  
↳ anps_sds/anps.shp"
```

Se ingresa la ruta del directorio donde se encuentran las capas de **Uso de suelo y Vegetación** \*\* en la variable **\*\*path\_dir\_usv**

```
path_dir_usv = "C:/Dropbox (LANCIS)/SIG/desarrollo/sig_fomix/entregables/usv_v2/"
```

Se ingresa la ruta del directorio de salida de los datos en la variable **\*\* path\_salida\*\***

```
path_salida = "C:/Dropbox (LANCIS)/SIG/desarrollo/sig_fomix/procesamiento/municipios_
↳ anp/"
```



en esta carpeta estarán los resultados del script, tambien se conservan los datos intermedios o productos de las intersecciones realizadas para la consulta de las áreas en una subcarpeta llamada **tmp**

### 6. Ingresar el nombre del campo ID de la capa A

se Ingresa el nombre del campo que contiene el identificador único para las geometrías en la variable **campo\_id\_geometria\_a**, en este caso el nombre del campo es **cve\_mun**

```
campo_id_geometria_a = 'cve_mun'
```

### 7. Ingresar el nombre del campo de las categorías USV

Se ingresa el nombre del campo que contiene las categorías de USV en la variable **\*\* campo\_clase\_usv\*\***

```
campo_clase_usv = 'id_clase2'
```

### 8. Ingresar claves de las tres geometrías

Se declaran en una lista tres claves de las tres geometrías involucradas en la variable **clave\_capas**

las claves son de tres caracteres y separadas por comas

```
clave_capas=['mun', 'anp', 'usv']
```

## 8.4 Bibliografía

## 8.5 Documentación dentro del código

---

### Cobertura de uso y tipo de suelo a nivel municipal

---

Objetivo:

Generar bases de datos a nivel municipal que cuantiquen el área en hectáreas por clase de uso de suelo y vegetación

# Insumos

- Series I - VI de uso de suelo y vegetación INEGI
- Municipios del estado de Yucatán (2018)

## Procedimiento

Se unificarán las categorías para las seis series publicadas de la siguiente manera: (Solo aplica para el estado de Yucatán)

id\_clase - Categoría 1 - Agricultura de riego 2 - Agricultura de temporal 3 - Cuerpo de agua 4 - Manglar 5 - Pastizal 6 - Selva baja 7 - Selva mediana 8 - Sin vegetación 9 - Asentamiento humano 10 - Vegetación de duna costera 11 - Vegetación de petén 12 - Vegetación secundaria de selva baja 13 - Vegetación secundaria de selva mediana 14 - Vegetación secundaria de manglar 15 - Acuícola 16 - Bosque cultivado/Palmar inducido 17 - Tular 18 - Vegetación halófila hidrófila 19 - Sábana

Se genera el script **datos\_nivel\_municipio.py** el cual genera realiza los siguientes pasos:

- Se declara **path\_mun** la ruta de la capa de municipios
- se realiza un iterador (for) del 1 al 6 para procesar las 6 series de USV
- Se declara **path\_usv** mediante el for la ruta de la capa usv\_serie\_i\_yuc.shp (donde i, va del 1 al 6)
- Se declara **path\_mun\_usv** mediante el for la ruta del archivo que resultará de la intersección de municipios y USV (agregados)
- Se declara **path\_mun\_usv\_csv** mediante el for la ruta del archivo csv que contendrá las áreas (Ha) por clase por municipio
- se declara **path\_mun** como la capa **municipios**
- se declara **path\_usv** como la cap **usv**
- Se crea una copia de la capa de municipio

- Se declara **path\_interseccion**, mediante el for que es la ruta y nombre del resultado de la intersección de municipios y USV
- Se crea una lista municipios mediante el campo **cve\_mun** de la capa **municipios**
- Se crea una lista de las categorías mediante el campo **id\_clase** de la capa **usv**
- Se realiza la intersección **path\_mun\_usv** y **path\_usv** se indica la ruta y nombre de salida con **path\_interseccion**
- se declara **path\_mun\_usv** como la capa **mun\_usv**
- Se llama a la función **campos\_clases** pasando como parámetros la cap **mun\_usv** y la lista\_clases, esta función creará los campos en la capa vectorial

como «clase\_i» donde i es el número de id de la clase - se declara **path\_interseccion** como la capa **consulta\_intersect**  
- Se inicia la edición de la capa **mun\_usv** - se realiza un iterador (for) de la lista\_clases

- Se inicializa la variable **area = 0**
- **Se realiza un iterador (for) de la lista\_mun**
  - se restablece la variable **area = 0**
  - se realiza un filtro mediante una consulta por municipio y por número de clase **request\_mun**
  - se realiza un filtro mediante una consulta por municipio **request\_mun\_o**
  - **Se realiza un iterador (for) de los elementos de la capa consulta\_interect pasando la consulta request\_mun**
    - Se realiza la suma de los elementos de la selección mediante la función **geometry().area()** y se va guardando en **area**
  - **Se realiza un iterador (for) de los elementos de la capa mun\_usv pasando la consulta request\_mun**
    - Se escribe en el campo **clase\_i** (donde i es el id de la clase) el valor del área dividido entre 10,000 y redondeado a 2 dígitos
  - Se actualizan los elementos de la capa **mun\_usv**
- al finalizar la serie de iteradores se guardan los cambios en **mun\_usv**
- Se manda a llamar a la función **vector\_to\_usv** donde se recibe como parámetros la capa **mun\_usv** y la variable **path\_mun\_usv\_csv** que es la ruta y el nombre del archivo csv

al finalizar se obtiene

- 6 capas vectoriales a nivel municipal, una por serie **mun\_usv\_si.shp** (donde i va del 1 al 6)
- 6 capas vectoriales resultado de las intersecciones **tp\_inters\_mun\_usv\_si.shp** (donde i va del 1 al 6), una por serie **\*\*mun\_usv\_si.shp** donde i va del 1 al 6)

se entrega como producto final

- 6 archivos csv a nivel municipal, uno por serie **mun\_usv\_si.csv** (donde i va del 1 al 6) que son copia de los atributos de la capa vectorial correspondiente

ruta : SIGdesarrollosig\_fomixentregablesmunicipios\_usv