Geoprocesamiento en python y qgis

Versión 1.0

Víctor Hernández

19 de marzo de 2021

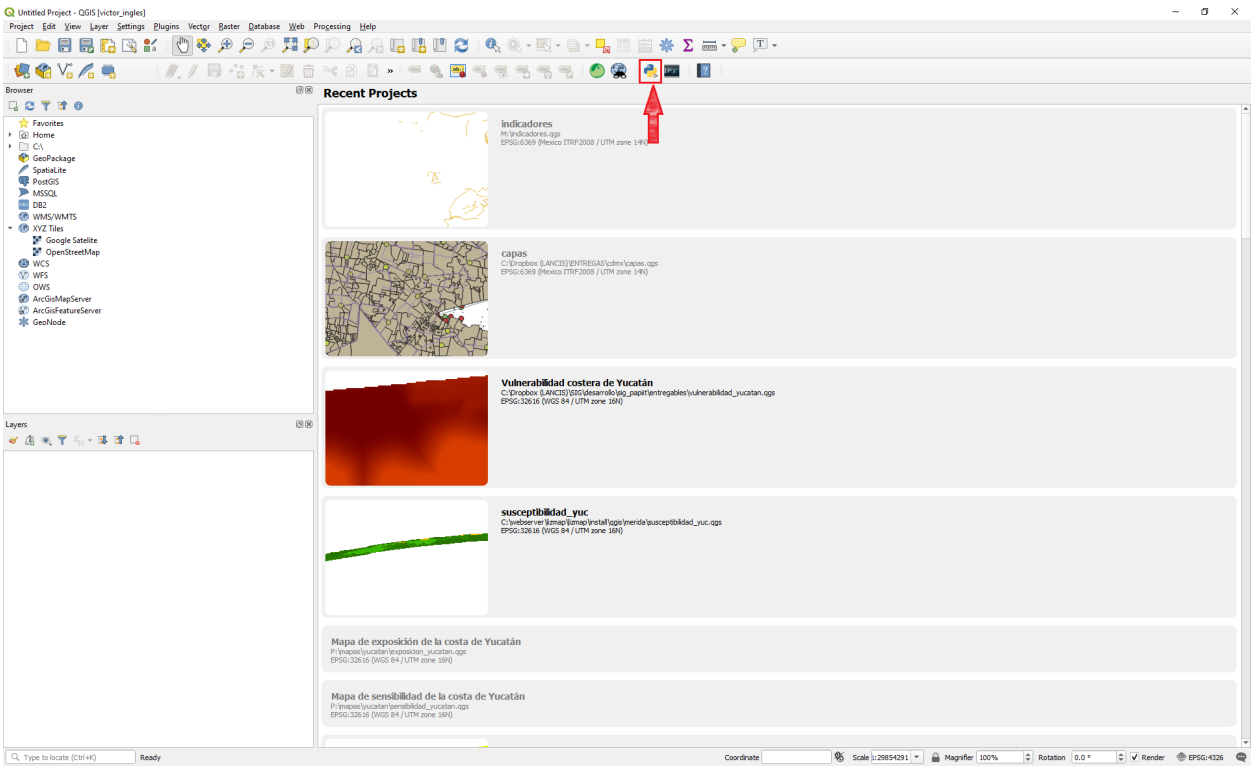
Guias de uso:

CAPÍTULO 1

¿Cómo ejecutar un código en Qgis?

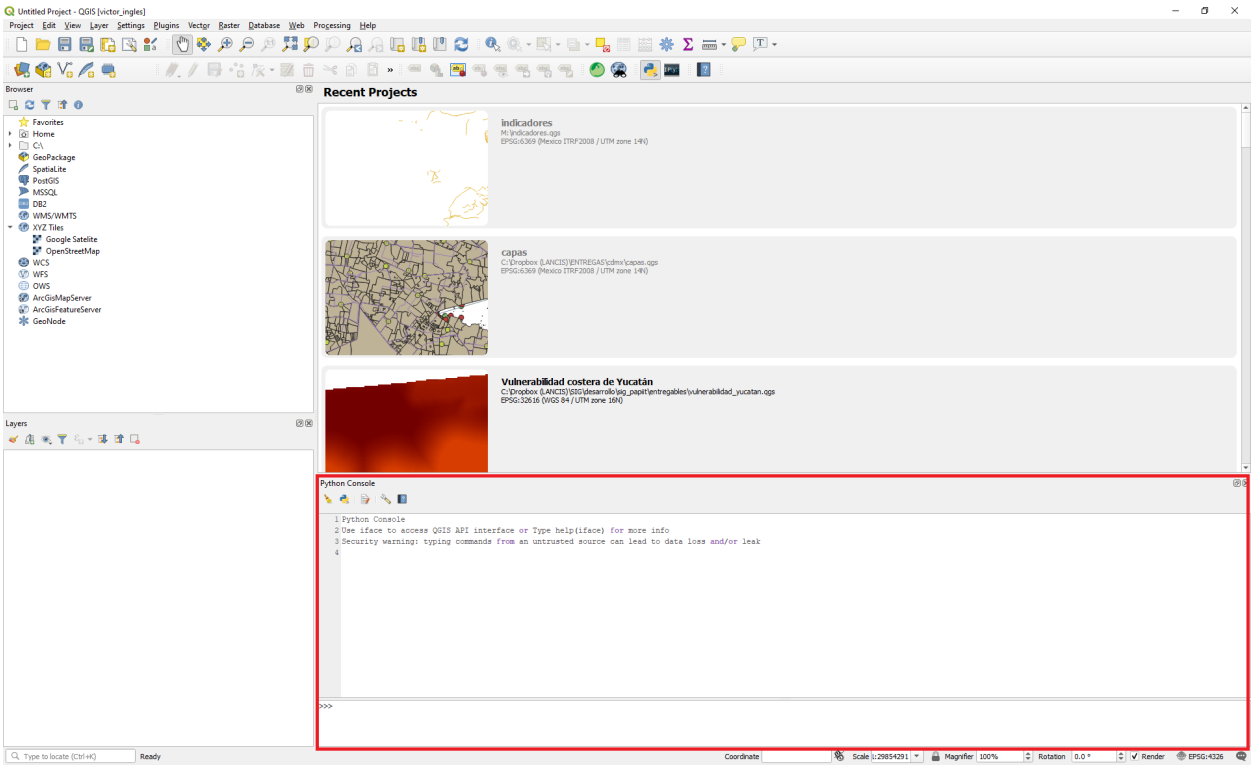
1.1 Paso #1

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



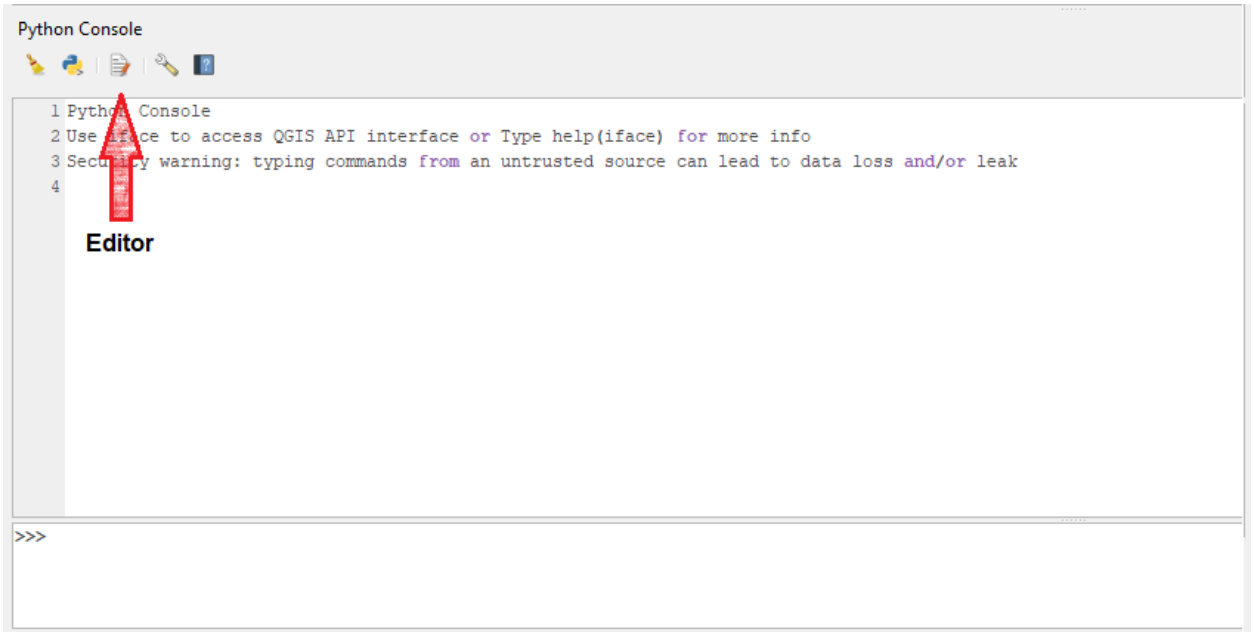
1.2 paso #2

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



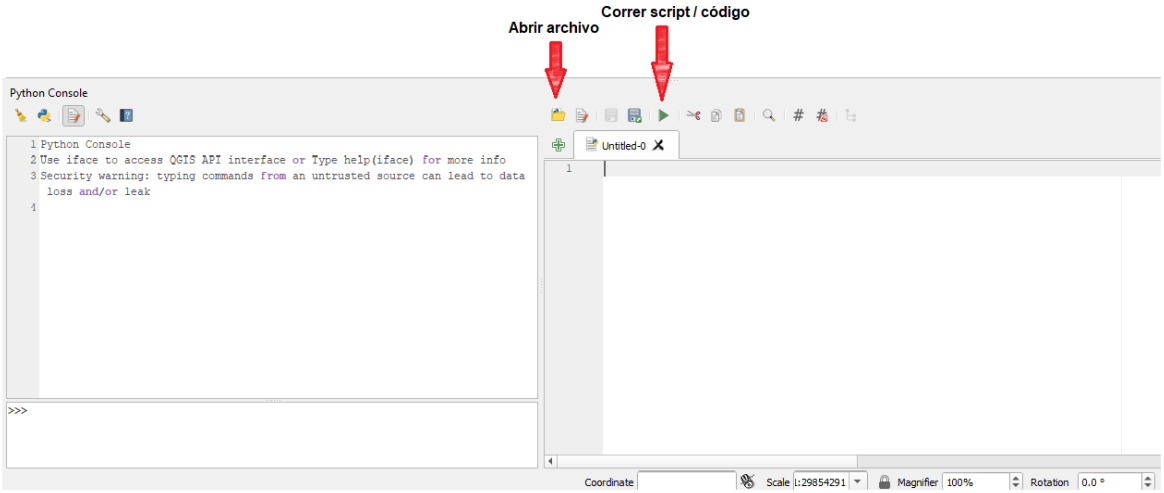
1.3 paso #3

Hacer clic en el ícono de **Editor**



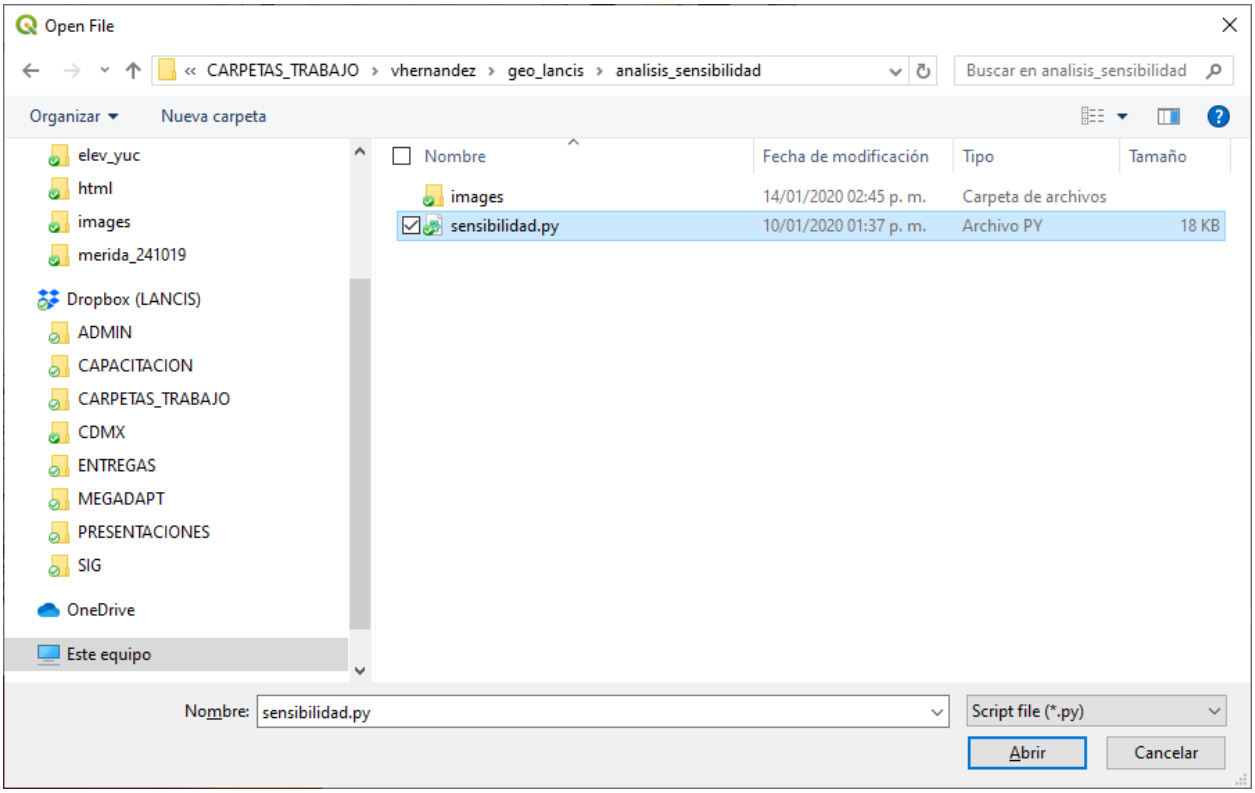
1.4 paso #4

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.


Open File
« CARPETAS_TRABAJO › vhernandez › geo_lancis › analisis_sensibilidad
Buscar en analisis_sensibilidad
Organizar
Nueva carpeta
elev_yuc
html
images
merida_241019
Dropbox (LANCIS)
ADMIN
CAPACITACION
CARPETAS_TRABAJO
CDMX
ENTREGAS
MEGADAPT
PRESENTACIONES
SIG
OneDrive
Este equipo
Nombre
Fecha de modificación
Tipo
Tamaño
images
14/01/2020 02:45 p. m.
Carpeta de archivos
sensibilidad.py
10/01/2020 01:37 p. m.
Archivo PY
18 KB
Nombre:
sensibilidad.py
Script file (*.py)
Abrir
Cancelar

CAPÍTULO 2

Libreria de funciones para el análsis 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 libreria

`apcsig.py`.

2.3 Documentación

Autor: Víctor Hernández D.

Correo: victor.hernandez@iecologia.unam.mx

user_github: @vichdzgeo

Colaboladores: LANCIS - UNAM

apcsig.**agregar_categorias**(*path_v*, *campo*, *nuevo_int_cats='categorias'*, *cont=1*)

Esta función reclasifica una capa en enteros consecutivos en función de las categorias únicas de un campo en especificico, como subproducto genera un archivo csv con las categorias.

Parámetros

- **path_v** (*str*) – ruta de capa vectorial
- **campo** (*str*) – nombre del campo que contiene las categorias
- **nuevo_int_cats** (*str*) – nombre del campo a crear, el cúal contendrá las categorias en enteros
- **cont** (*int*) – contador para empezar la numeración en el número indicado, por defecto es 1

apcsig.**alinear_raster**(*path_raster*, *region*, *resolucion*, *path_salida*, *crs_destino=''*, *tipo='int'*)

Esta función alinea un raster dada una región y el tamaño de pixel :param path_raster: ruta de la capa a alinear :type path_raster: str

Parámetros

- **region** (*str*) – coordenadas de la región del estudio xmin,xmax,ymin,ymax
- **resolucion** (*int*) – tamaño de pixel
- **path_salida** (*str*) – ruta de la capa de salida con extension tif
- **crs_destino** (*str*) – nombre del código EPSG,
- **tipo** (*str*) – tipo de dato, use "int" para entero o "float" para flotante, por default es entero

apcsig.**aplica_mascara**(*path_mascara*, *path_capa*, *path_salida*, *region*)

Esta función aplica la máscara de la zona de estudio a una capa raster, es importante que la capa a la cual se aplicará la máscara este previamente alineada

Parámetros

- **path_mascara** (*str*) – ruta de la mascara en formato tiff
- **path_capa** (*str*) – ruta de la capa a la cual se requiere aplicar la máscara
- **path_salida** (*str*) – ruta de la capa resultado de aplicar la máscara
- **region** (*str*) – coordenadas de la región del estudio xmin,xmax,ymin,ymax

apcsig.**asignar_nulls**(*map*, *output*, *valor_huecos=0*)

Esta función asigna un valor a los no_data de la capa

Parámetros

- **map** (*str*) – ruta de la capa raster
- **region** (*str*) – coordenadas de la región del estudio xmin,xmax,ymin,ymax

:param output:ruta de la capa resultante :type output: str

Parámetros **valor_huecos** (*int*) – número que tendrán los pixeles nulos

apcsig.**calculadora_grass**(*path_capa*, *ecuacion*, *path_salida*)
Esta función aplica la máscara de la zona de estudio

Parámetros

- **path_mascara** (*str*) – ruta de la mascara en formato tiff
- **path_capa** (*str*) – ruta de la capa a la cual se requiere aplicar la máscara
- **path_salida** (*str*) – ruta de la capa resultado de aplicar la máscara
- **region** (*str*) – coordenadas de la región del estudio xmin,xmax,ymin,ymax

apcsig.**calculadora_grass_2capas**(*path_capa_a*, *path_capa_b*, *ecuacion*, *path_salida*)
Esta función aplica la máscara de la zona de estudio

Parámetros

- **path_mascara** (*str*) – ruta de la mascara en formato tiff
- **path_capa** (*str*) – ruta de la capa a la cual se requiere aplicar la máscara
- **path_salida** (*str*) – ruta de la capa resultado de aplicar la máscara
- **region** (*str*) – coordenadas de la región del estudio xmin,xmax,ymin,ymax

apcsig.**calculadora_grass_3capas**(*path_capa_a*, *path_capa_b*, *path_capa_c*, *ecuacion*, *path_salida*)
Esta función aplica la máscara de la zona de estudio

Parámetros

- **path_mascara** (*str*) – ruta de la mascara en formato tiff
- **path_capa** (*str*) – ruta de la capa a la cual se requiere aplicar la máscara
- **path_salida** (*str*) – ruta de la capa resultado de aplicar la máscara
- **region** (*str*) – coordenadas de la región del estudio xmin,xmax,ymin,ymax

apcsig.**campos_mayusculas**(*path_shape*)
Esta función renombra los campos en mayusculas

Parámetros **path_shape** (*str*) – Ruta de la capa vectorial

apcsig.**campos_minusculas**(*path_shape*)
Esta función renombra los campos en minúsculas

Parámetros **path_shape** (*str*) – Ruta de la capa vectorial

apcsig.**capa_binaria**(*path_v*, *campo_cat='presencia'*, *valor=1*)
Esta función crea un campo llamado presencia y asigna a cada elemento el valor de 1

Parámetros

- **path_v** (*str*) – ruta de la capa vectorial
- **campo_cat** (*str*) – nombre del campo a crear, por default es presencia

apcsig.**categorias_campo_csv**(*path_shape*, *campo*)
Esta función extrae las categorias únicas de un campo dado de una capa vectorial, el archivo csv se guarda en la misma ruta de la capa vectorial y es nombrado como : **categorias_campo_nombre_capa.csv**

Nota: En dado caso que el nombre de la categoria contenga el símbolo de "," está función la remplaza por ";" para evitar errores en la escritura del archivo csv

Parámetros

- **path_shape** (*str*) – ruta de la capa vectorial
- **campo** (*str*) – nombre del campo que contiene las categorias

apcsig.**clasificar_shape**(*path_v*, *clasificador*, *l_field*, *campo_cat=''*, *fp=2*, *categories=5*)
Funcion integradora para clasificar la capa vectorial

Parámetros

- **path_v** (*str*) – ruta de la capa vectorial
- **clasificador** (*str*) – nombre del clasificador
- **fp** (*float*) – factor de progresión
- **categories** (*int*) – número de categorias

apcsig.**crea_capa_raster**(*ecuacion*, *rasters_input*, *salida*, *decimales=3*)
Esta función crea una capa mediante la calculadora raster de GDAL, esta función esta limitada hasta 14 variables en la ecuación.

Parámetros

- **ecuacion** (*str*) – ecuación expresada en formato gdal,
- **rasters_input** (*list*) – lista de los paths de los archivos rasters
- **salida** (*str*) – ruta con extensión tiff de la salida

Devuelve Capa raster de tipo flotante, los valores de la capa son redondeados a 3 decimales

apcsig.**crear_campo**(*path_vector*, *nombre_campo*, *tipo*)
Esta funcion crea un campo segun el tipo especificado. Parametros: :param path_vector: La ruta del archivo shapefile al cual se le quiere agregar el campo :type path_vector: String

Parámetros

- **nombre_campo** (*Sting*) – Nombre del campo nuevo
- **tipo** – es el tipo de campo que se quiere crear

Int: para crear un campo tipo entero Double: para crear un campo tipo doble o flotante String: para crear un campo tipo texto Date: para crear un campo tipo fecha :type tipo: String

apcsig.**cuantiles_s**(*path_v*, *quantil*, *field*, *min*, *max*)
Esta función regresa la lista de cortes según el cualtil deseado de los valores de un campo de la capa vectorial de entrada

Parámetros

- **path_v** (*str*) – ruta de la capa vectorial
- **quantil** (*int*) – cuantil
- **field** (*str*) – nombre del campo
- **min** (*float*) – valor mínimo de la capa
- **max** (*float*) – valor máximo de la capa

apcsig.**distancia_caminos_lugar**(*layer_raster_lugar*, *path_caminos*, *campo_caminos*, *path_mascara*, *path_salida*, *region_ext*, *ancho_ext=2292*, *alto_ext=2284*, *remover=1*)

Esta función genera una capa de distancia a caminos y agrega distancia cero a aquellos pixeles que se sobreponen con el área del lugar

apcsig.**ecuacion_clp**(*pesos*)

Esta función recibe una lista de pesos para regresar la ecuación en la estructura requerida por gdal para la combinación lineal ponderada.

ejemplo:

$$ecuacion = A * 0{,}40 + B * 0{,}25 + C * 0{,}15 + D * 0{,}20$$

Parámetros **pesos** (*lista*) – lista de los pesos de las capas, salida de la función

Devuelve ecuación en formato gdal para ser ingresada a la funcion crea_capa_raster

apcsig.**equidistantes**(*categories=5*, *min=0*, *max=1*)

Esta función regresa la lista de cortes equidistantes según el número de categorias y el valor minimo y maximo ingresados.

Parámetros

- **categories** (*int*) – número de categorias
- **min** (*float*) – valor mínimo de la capa
- **max** (*float*) – valor máximo de la capa

apcsig.**get_region**(*path_layer*)

Esta función extrae la región o extensión de una capa raster así como el número de columnas y renglones

Parámetros **path_layer** (*str*) – ruta de la capa raster

Devuelve en forma de lista [1,2,3] [1] las coordenadas de la extensión de una capa raster xmin,xmax,ymin,ymax ; [2]. ancho de una capa raster (número de columnas) y [3]. alto de una capa raster (número de renglones)

apcsig.**integra_localidades_caminos**(*path_lugar_n*, *w_lugar*, *path_d_camino_n*, *w_d_camino*, *d_max_lugar*, *salida*)

Esta función aplica la máscara de la zona de estudio

Parámetros

- **path_mascara** (*str*) – ruta de la mascara en formato tiff
- **path_capa** (*str*) – ruta de la capa a la cual se requiere aplicar la máscara
- **path_salida** (*str*) – ruta de la capa resultado de aplicar la máscara
- **region** (*str*) – coordenadas de la región del estudio xmin,xmax,ymin,ymax

apcsig.**llenar_campos_nulos**(*path_vector*, *valor=-9999*)

Parámetros

- **path_vector** (*str*) – Ruta de la capa vectorial
- **valor** – valor numérico para rellenar los campos vacios por default se establece -9999

apcsig.**max_min_vector**(*layer*, *campo*)

Esta función regresa el maximo y minimo del campo elegido de la capa vectorial de entrada

Parámetros

- **layer** (*QgsLayer*) – capa vectorial

- **campo** (*str*) – nombre del campo

apcsig.**normailiza**(*path_raster*, *path_raster_n*, *modo='ideales'*)

Esta función normaliza una capa raster, se puede elegir entre dos tipos de normalización.

1) ideales: La capa ráster se divide entre el valor máximo como resultado se tiene una capa con un máximo de 1 pero el valor mínimo no necesariamente será 0

$$ideales = \frac{A}{A.max}$$

2) lineal: la capa ráster resultante tendra como valor máximo 1 y como valor mínimo 0 el 1 representará el maximo de la capa de entrada y el 0 representa el valor mínimo de la capa de entrada

$$lineal = \frac{A - A.min}{A.max - A.min}$$

apcsig.**nulls**(*map*, *output*, *valor_huecos=0*)

Esta función asigna un valor a los no_data de la capa

Parámetros

- **map** (*str*) – ruta de la capa raster
- **region** (*str*) – coordenadas de la región del estudio xmin,xmax,ymin,ymax

:param output:ruta de la capa resultante :type output: str

Parámetros **valor_huecos** (*int*) – número que tendrán los pixeles nulos

apcsig.**raster_min_max**(*path_raster*)

Esta funcion regresa los valores maximos y minimos de una capa raster

Parámetros **path_raster** (*str*) – ruta de la capa raster

apcsig.**rasterizar_vector**(*path_vector*, *n_campo*, *region*, *path_salida*, *tipo='int'*, *ancho=0*, *alto=0*)

Esta función rasteriza una capa vectorial a partir de un campo de tipo numérico y dada una región y el numero de columnas (ancho) y el numero de renglones (alto)

Parámetros

- **path_vector** (*str*) – ruta de la capa vectorial
- **n_campo** (*srt*) – nombre del campo que contiene los id de las categorias
- **region** (*str*) – coordenadas de la región del estudio xmin,xmax,ymin,ymax
- **path_salida** (*str*) – ruta de la capa de salida con extension tif
- **tipo** (*str*) – tipo de dato, use "int" para entero o "float" para flotante, por default es entero

apcsig.**reclasifica_capa**(*capa*, *region*, *reglas*, *salida*)

Esta función permite reclasificar una capa raster y genera un archivo xml el cúal contiene el nombre de las categorias.

Parámetros

- **capa** (*str*) – ruta de capa de entrada
- **region** (*str*) – coordenadas de la región del estudio xmin,xmax,ymin,ymax
- **reglas** (*str*) – ruta del archivo txt que contiene las reglas de clasificación
- **salida** (*str*) – ruta de la capa de salida reclasificada

Retuns Capa raster clasificada y archivo xml

apcsig.**redondea_raster**(*path_raster*, *salida*, *no_decimales=3*)
Esta función redondea una capa raster de tipo flotante en el número de decimales indicado

Parámetros

- **path_raster** (*str*) – ruta de la capa raster
- **no_decimales** – número de decimales a los que se va a redondear la capa, por default es 3
- **salida** (*str*) – ruta de la capa de salida

apcsig.**remove_raster**(*path_r*)
Esta función elimina una capa del sistema

Parámetros **path_r** (*str*) – ruta de la capa

apcsig.**tipo_clasificador_s**(*clasificador*, *path_v*, *l_field*, *campo_cat=''*, *fp=2*, *categories=5*, *min=0*, *max=1*)

Esta función integra los modos de clasificación, weber-fechner, progresiva, cuartiles, quintiles, deciles o equidistante

param clasificador: tipo de clasificador (progresiva, cuartiles, quintiles, deciles, equidistante) type clasificador: str

param path_v ruta de la capa vectorial

type path_v str

param l_field nombre del campo

type l_field str

param fp factor de progresión

type fp float

param categories número de categorias

type categories int

param min valor mínimo de la capa

type min float

param max valor máximo de la capa

type max float

apcsig.**vcopia**(*path_vector*, *path_salida*)
Crea una copia de la capa a partir de la ruta de la capa, la capa es creada con el mismo sistema de referencia que el origen.

Parámetros

- **path_vector** (*String*) – ruta de la capa original
- **path_salida** (*String*) – ruta de donde sera almacenada la capa

CAPÍTULO 3

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:

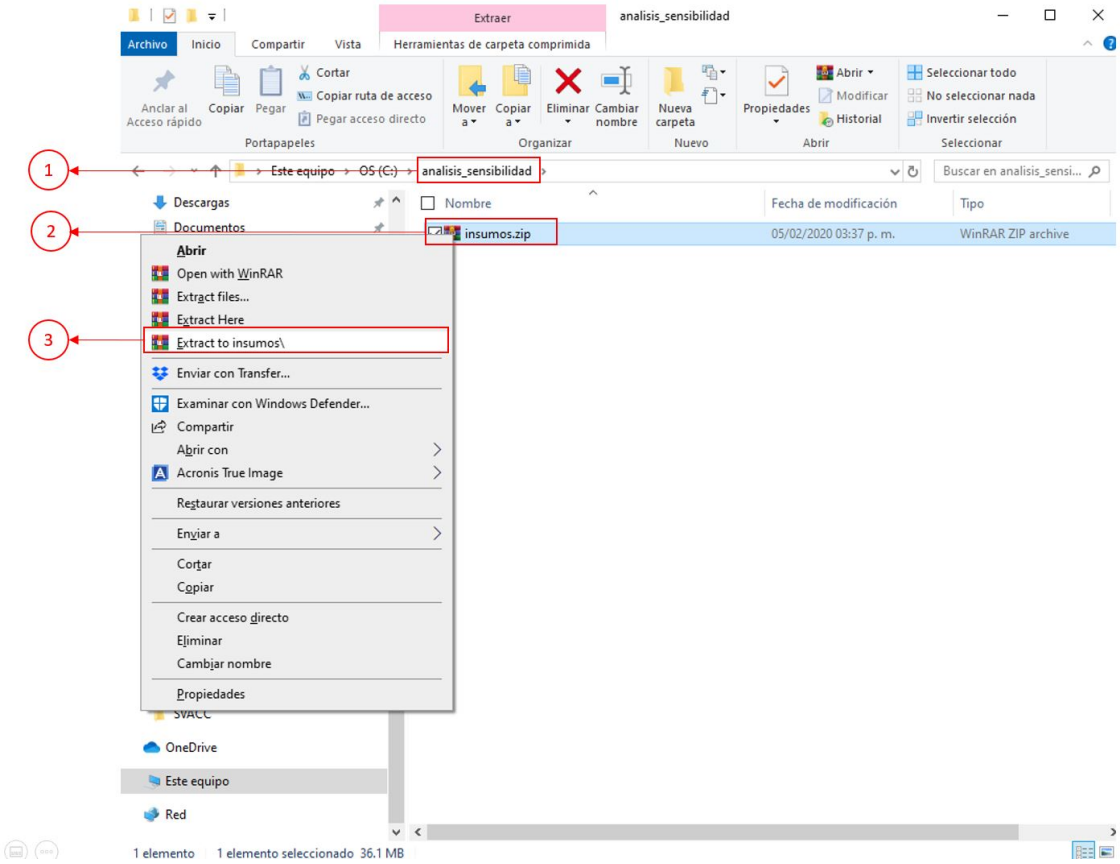
- Misma proyección cartográfica
- Mismo tamaño de pixel

- Misma extensión de capa
- Mismo valor de NoData

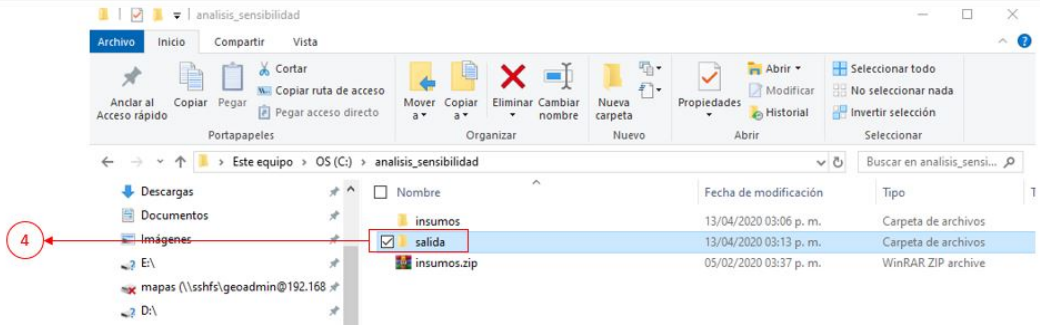
3.3 Ejemplo

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 `aqui` (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**



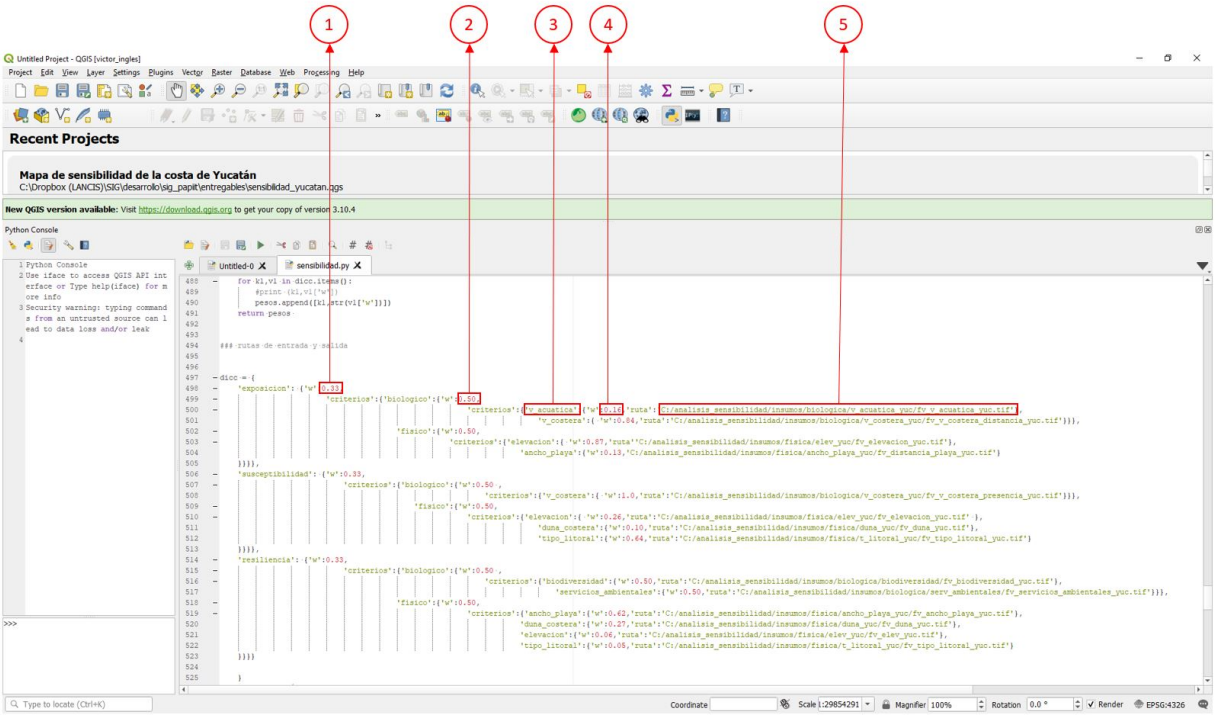
3.3.2 Procedimiento

1. Abrir el código

Abrir el código **sensibilidad.py** en Qgis 3.4 o superior, Para resolver cualquier duda al respecto, consultar la guia

2. Actualizar el diccionario

Ingresar la (1) ponderación del compontente 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.



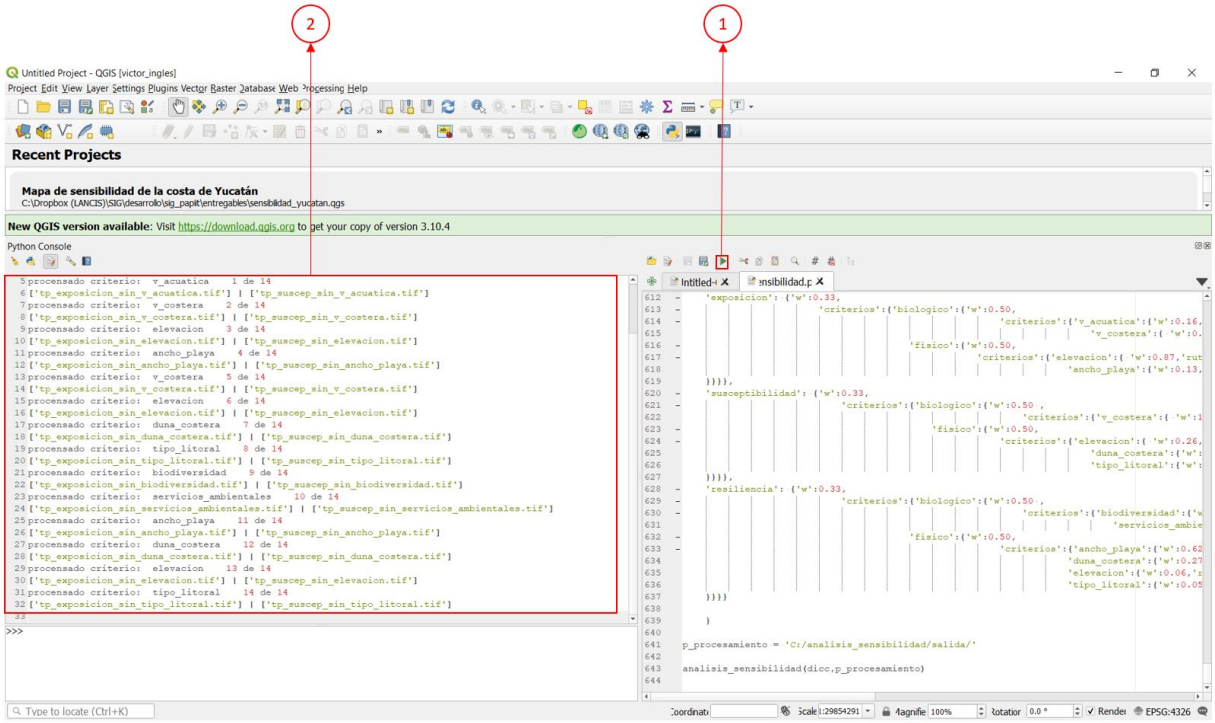
3. Indicar el direcctorio de salida

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

Autores: LANCIS -APC, Fidel Serrano,Victor Hernandez

Qgis 3.4 o superior

sensibilidad_por_remocion_capas.**crea_capa**(*ecuacion*, *rasters_input*, *salida*)

Esta función crea una capa mediante la calculadora raster de GDAL, esta función esta limitada hasta 14 variables en la ecuación.

Parámetros

- **ecuacion** (*String*) – ecuación expresada en formato gdal, es este caso es la salida de la funcion *ecuacion_clp*
- **rasters_input** (*lista*) – lista de los paths de los archivos rasters, salida de la función *separa_ruta_pesos*
- **salida** (*String*) – ruta con extensión tiff de la salida

sensibilidad_por_remocion_capas.**ecuacion_clp**(*pesos*)

Esta función recibe una lista de pesos para regresar la ecuación en la estructura requerida por gdal para la combinación lineal ponderada.

Parámetros **pesos** (*lista*) – lista de los pesos de las capas, salida de la función *separa_ruta_pesos*

sensibilidad_por_remocion_capas.**ecuacion_vulnerabilidad**(*n*)

Esta función expresa la ecuación para el cálculo de la vulnerabilidad

$$vulnerabilidad = \exp^{(1-sus)^{(1+ca)}}$$

$$|exp = Exposicin|sus = Susceptibilidad|ca = Capacidadadaptativa$$

Devuelve str ecuacion

sensibilidad_por_remocion_capas.**get_region**(*path_layer*)
Esta función regresa en forma de cadena de texto las coordenadas de la extensión de una capa raster

param path_layer: ruta de la capa raster type path_layer: str

sensibilidad_por_remocion_capas.**lista_criterios**(*dicc*)
Esta función regresa una lista de los criterios de un diccionario

Parámetros **dicc** – Diccionario que contiene nombres, rutas y pesos para el

análisis de vulnerabilidad / sensibilidad :type dicc: diccionario python

sensibilidad_por_remocion_capas.**lista_pesos_ruta**(*dicc*)
Funcion para sacar listas por subcriterio

sensibilidad_por_remocion_capas.**media_raster**(*path_raster*)
Esta función regresa el promedio de todos los pixeles válidos de un archivo raster

Parámetros **path_raster** (*String*) – Ruta del archivo raster

sensibilidad_por_remocion_capas.**nombre_capa**(*path_capa*)
Esta función regresa el nombre de una capa sin extensión

Parámetros **path_capa** (*str*) – ruta de la capa

sensibilidad_por_remocion_capas.**quita**(*dicc*, *key*)
Esta función retira un elemento del diccionario y regresa un nuevo diccionario sin dicho elemento <<dicc_q>>.

Parámetros

- **dicc** (*diccionario*) – Diccionario con la estructura requerida
- **key** (*String*) – nombre de la variable a quitar

sensibilidad_por_remocion_capas.**quita_reescala**(*dicc*, *key*)
Función que integra las funciones quita y reescala y regresa un diccionario sin la variable y con los pesos reescalados.

Parámetros

- **dicc** (*diccionario*) – Diccionario con la estructura requerida
- **key** (*String*) – nombre de la variable a quitar

sensibilidad_por_remocion_capas.**raster_min_max**(*path_raster*)
Esta funcion regresa los valores maximos y minimos de una capa raster

Parámetros **path_raster** (*str*) – ruta de la capa raster

sensibilidad_por_remocion_capas.**reescala**(*dicc_q*)
Esta función rescala un diccionario que se le a quitado un criterio y regresa el diccionario con los pesos rescalados

Parámetros **dicc_q** – salida de la función *quita*

CAPÍTULO 4

Índice Lee-Sallee

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

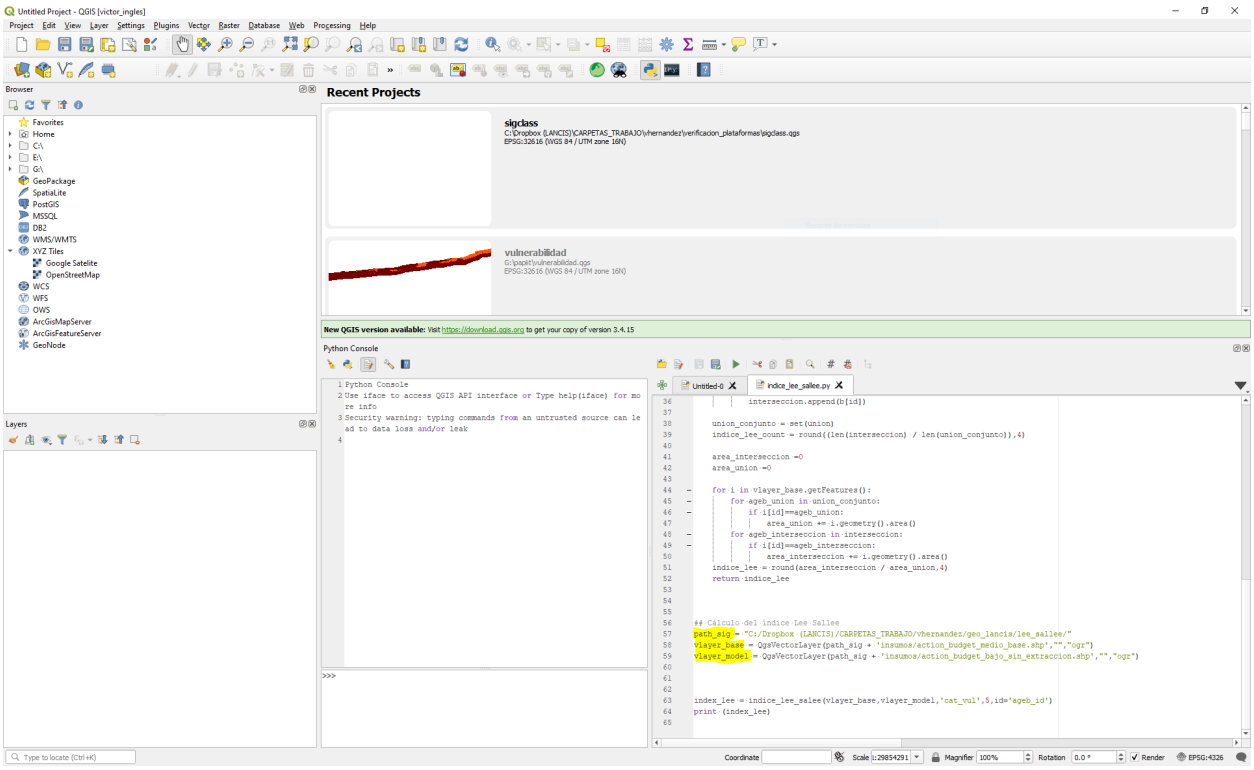
4.1.1 Insumos

Descarga los insumos para este ejemplo `aqui`

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 guia
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



indice_lee_sallee.**indice_lee_salee**(*vlayer_base*, *vlayer_model*, *campo_categoria*, *categoria*, *id='ageb_id'*)

Esta función regresa el índice Lee-Sallee

$$lee\_sallee\_index =$$

rac{Acap B}{Acup B}

param vlayer_base vector base

type vlayer_base QgsVectorLayer

param vlayer_model Vector modelo

type vlayer_model QgsVectorLayer

param campo_categoria Nombre del campo que tiene las categorias

type campo_categoria String

param categoria Número de categoria

type categoria int

param id nombre del campo identificador

type id String

indice_lee_sallee.**lista_shp**(*path_carpeta*)

Parámetros **path_carpeta** (*String*) – ruta que contiene los archivos shape a procesar

CAPÍTULO 5

OWA

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_ij = 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

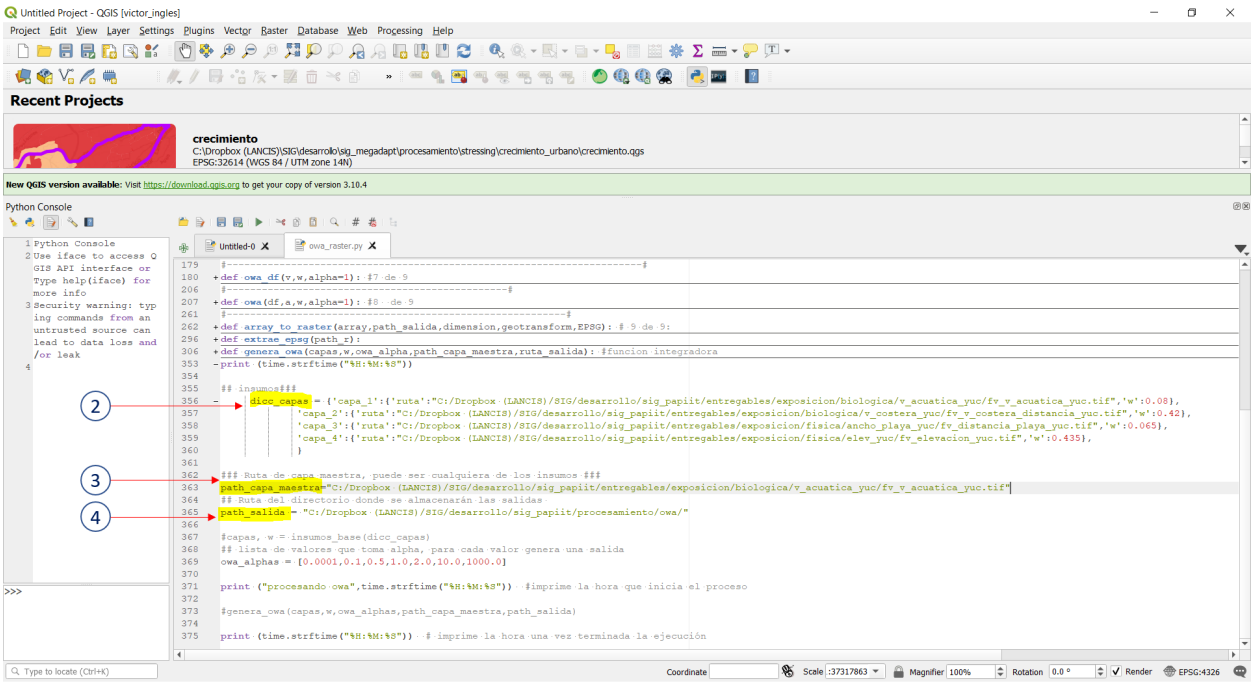
5.3.1 Insumos

Descargar los insumos para este ejemplo `aqui`

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 guia



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/ancho_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 extraidos 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 direcctorio 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

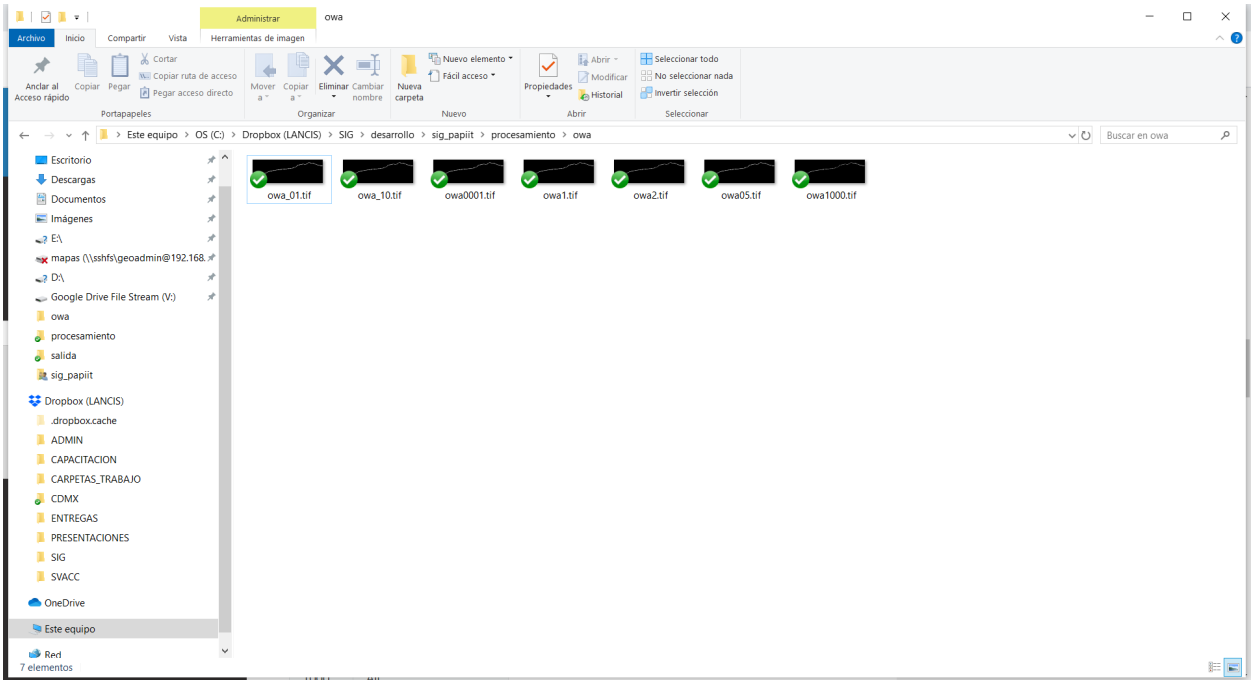
Nota: Para más información respecto a los valores de alpha consulte la bibliografía

---

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

α	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 Geoin-formation, 8 ,270-277.

5.5 Documentación dentro del código

CAPÍTULO 6

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 declardos como texto
6. Metadatos Que exista el archivo xml asociado a los metadatos geográficos

Descargar el código de ejemplo

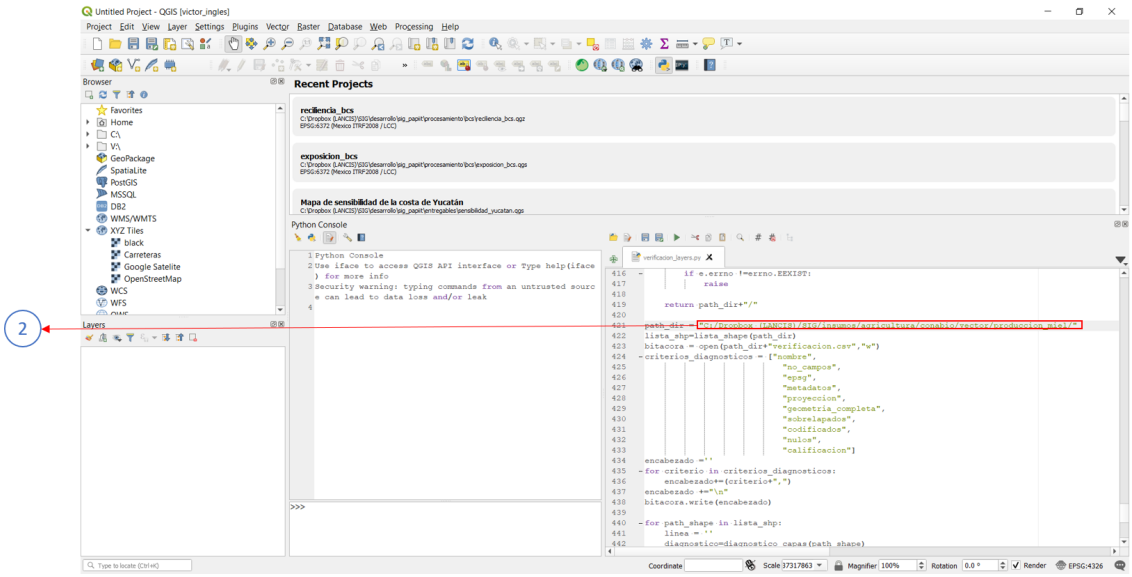
`verificacion_layers.py`.

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 guia



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**

estan nombrados de la siguiente manera:

nombrecapa_error.shp

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

nombrecapa_invalido.shp

Capa de poligonos que índica geométria inválida de la capa

nombrecapa_valido.shp

Capa de poligonos 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**

CAPÍTULO 7

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 geométria

7.2.2 Entidad B

- Que cuente con un campo dónde se especifiquen las categorias 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

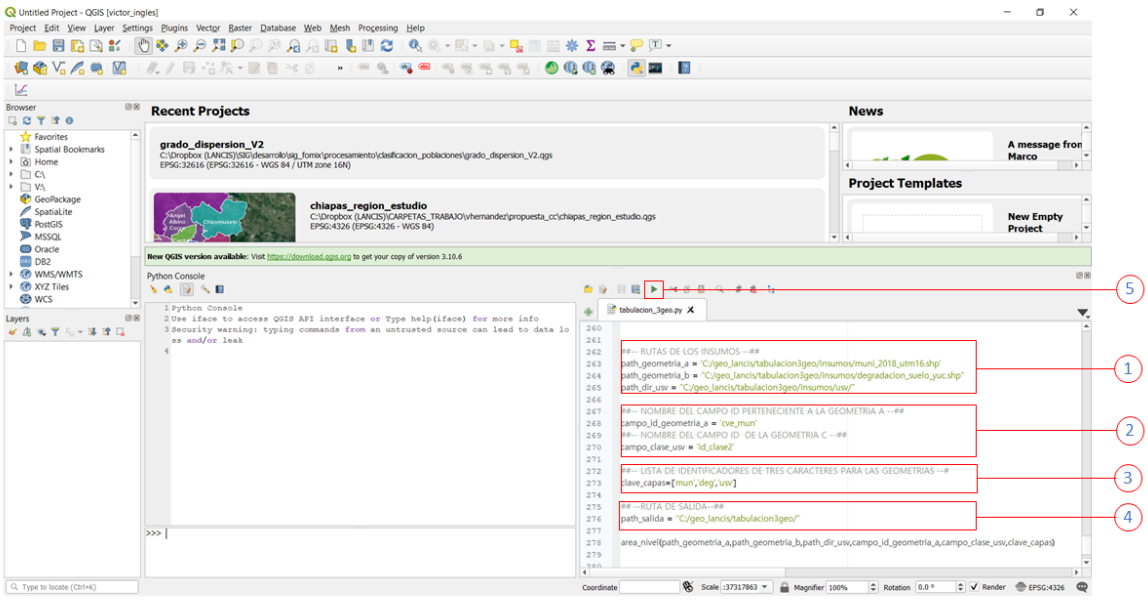
7.3.1 Datos de prueba

Descargar los datos de prueba para este ejemplo `aqui`

insumo	Descripción
muni_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 separadador de espacio en lugar de "'

```
path_geometria_a = 'C:/geo_lancis/tabulacion3geo/insumos/muni_2018_utm16.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 cúal 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 categorias de tres caracteres.

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

4. Indica el directorio de salida

En esta ruta se escribirá la tabla resultado de la cruza adicionalmente se crea una carpeta llamada **tmp** en la cúal se almacenán las cruzas ralizadas en el proceso.

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

5. Ejecuta el script

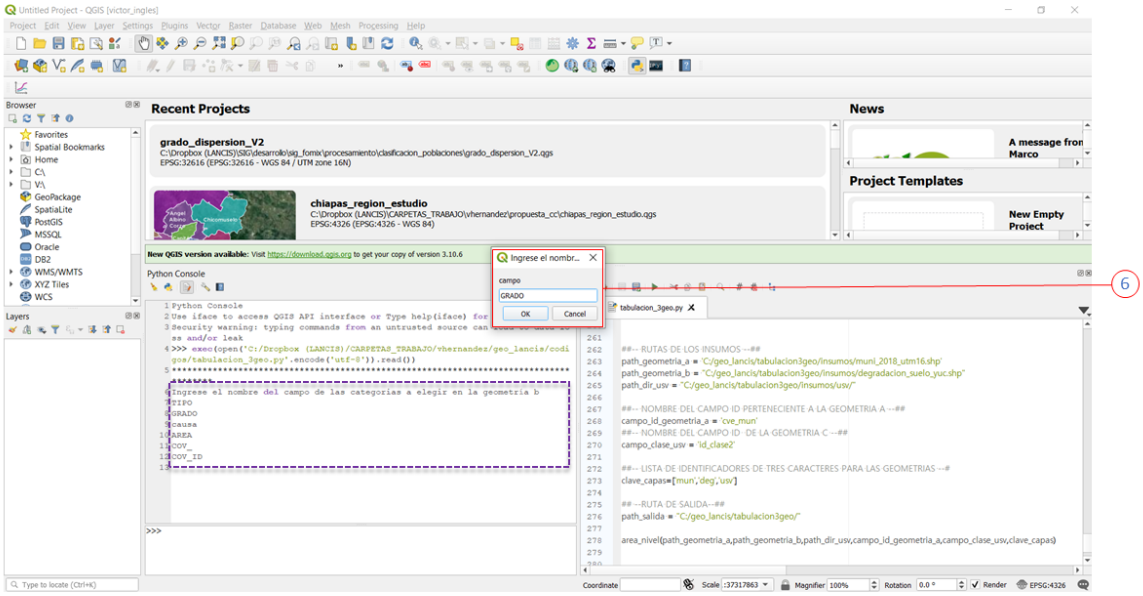
Hacer clic en el botón de ejecutar y permanece atento a la consola.

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

Se mostrará en la consola la siguiente instrucción:

Ingrese el nombre del campo de las categorias 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**



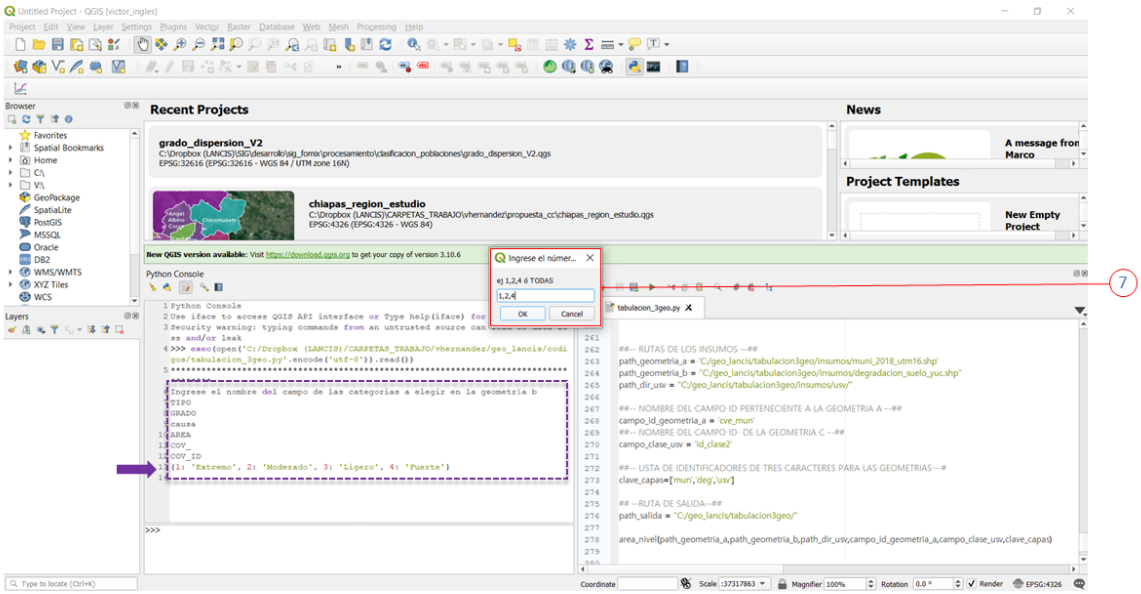
7. Ingresar el número de las categorías a considerar

El script permite ingresar todas las categorías o solo algunas, las categorias 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 geométrias con la categoria 0, una vez finalizado ese proceso, realiza la cruza solo con las categorias consideredas.

Ingrese las categorias que desea considerar conforme se muestra en la consola, para este ejemplo solo consideramos las categorias “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**

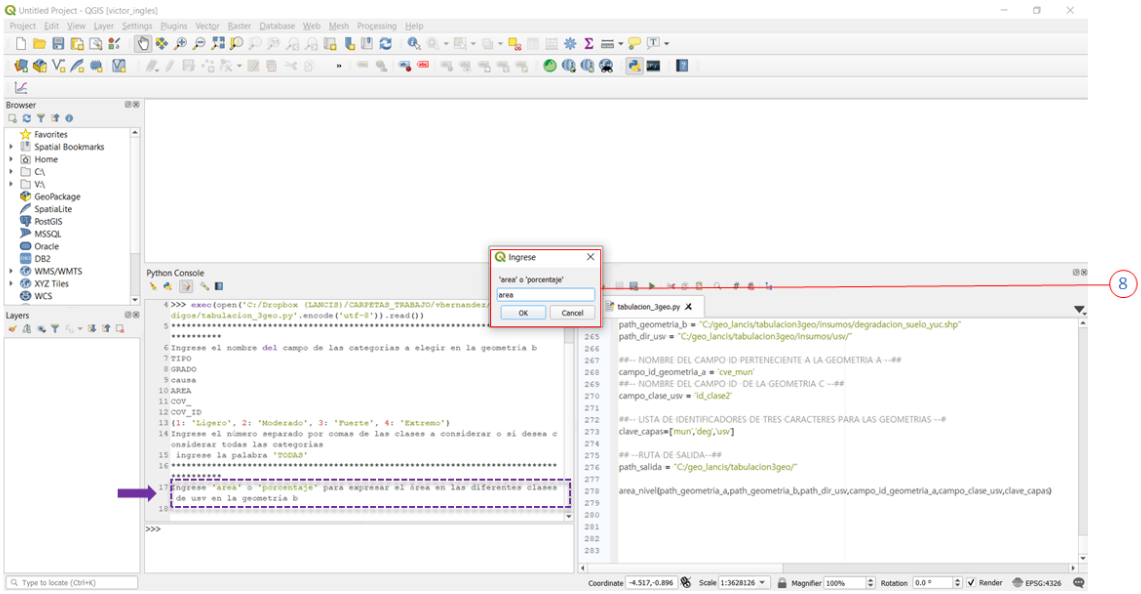
Nota: puede ingresar solo una categoria escribendo por ejemplo: 1,

Puede ingresar todas las categorias escribiendo la palabra «TODAS»



8.- Elegir entre área o porcentaje

La cuantificación de las categorias 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”



CAPÍTULO 8

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** (geometria 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 topólogicos

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 categorias 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

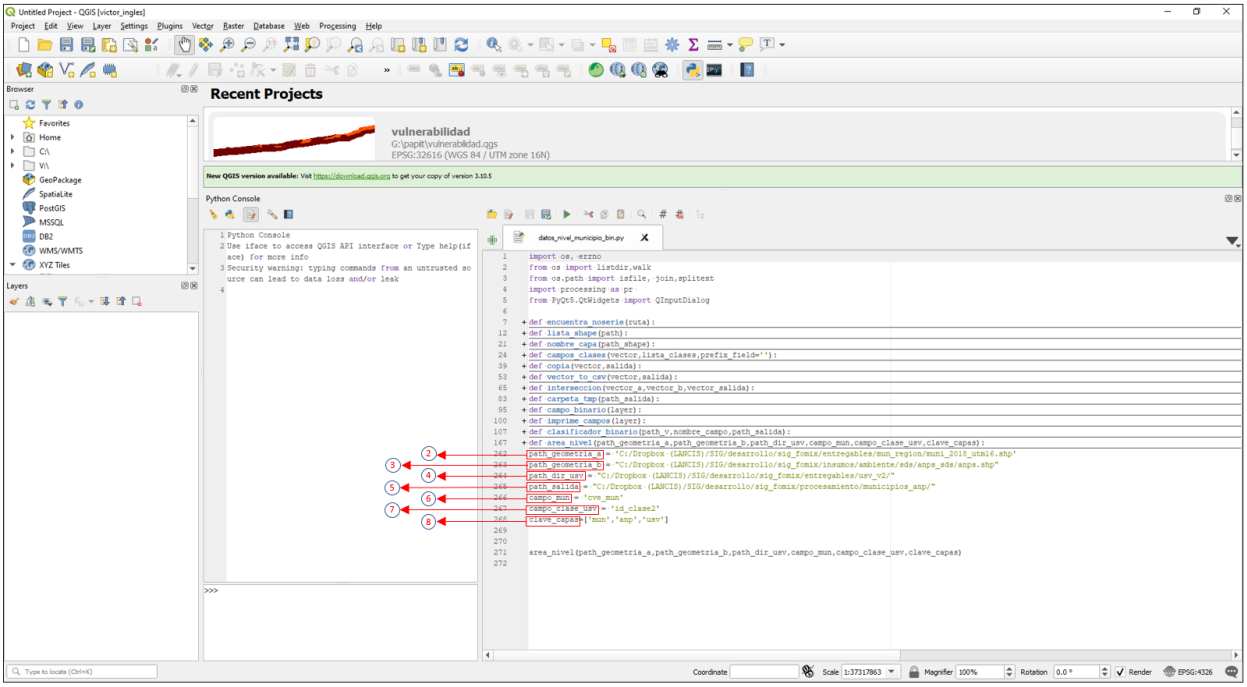
8.3 Ejemplo

8.3.1 Insumos

Descargar los insumos para este ejemplo `aqui`

1. Abrir el código

Abrir el código **owa_raster.py** en Qgis 3.4 o superior, Para resolver cualquier duda al respecto, consultar la guia



2. Ingresar la ruta de la geometria A

Se ingresa la ruta completa de la capa de **municipios** en la variable **path_geometria_a**

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

3. Ingresar la ruta de la geometria B

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"
```

4. Ingresar el directorio de las series de USV (geometría C)

Se ingresa la ruta del directorio donde se encutran 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/"
```

5. Ingresar el directorio de salida

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_geomatria_a = 'cve_mun'
```

7. Ingresar el nombre del campo de las categorias USV

Se ingresa el nombre del campo que contiene las categorias 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

CAPÍTULO 9

Cobertura de uso y tipo de suelo a nivel municipal

Objetivo:

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

Insumos

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

Procedimiento

Se unificarón las categorias para las seis series publicadas de la siguiente manera: (Solo aplica para el estado de Yúcatan)

id_clase - Categoria 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** mendiante 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**, mendiante el for que es la ruta y nombre del resultado de la intersección de municipios y USV
- Se crea una lista municipios mendiante el campo **cve_mun** de la capa **municipios**
- Se crea una lista de las categorias 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 funcion **campos_clases** pasando como parametros 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 numero 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 elmentos de la seleccion mediante la funcion geometry().area() y se va guardando en **area**
    - **Se realiza un iterdor (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 parametros 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

Índice de Módulos Python

Índice

S

T

V