Introducción a Python usando Notebooks en ArcGIS Pro

EDICIÓN ESTUDIANTE

Ejercicio

30 minutos

Usando Python para acceder a datos geográficos

En este módulo revisaremos como mediante Python y Notebooks en ArcGIS Pro permiten acceder a Datos Geográficos. En este emocionante viaje, exploraremos cómo utilizar el poderoso lenguaje de programación Python para trabajar con datos geográficos y geoespaciales en ArcGIS Pro. La capacidad de manipular y analizar datos geográficos es fundamental en el campo de la SIG (Sistemas de Información Geográfica) y en diversas aplicaciones, Python, con su amplia gama de bibliotecas y herramientas, se ha convertido en una herramienta esencial para profesionales y entusiastas que trabajan con datos geoespaciales.

En este ejercicio, usted realizara las siguientes tareas:

- Crear notebooks usando ArcGIS Pro.
- Escribir líneas de comando en python para acceder y visualizar datos geográficos.

Paso 1: Abra un Notebook de Python en ArcGIS Pro

a. Abra ArcGIS Pro.

b. Cree un nuevo Proyecto con una plantilla de mapa y nómbrelo **"Notebooks_IDU"**.

c. Diríjase a la pestaña **Análisis (Analyisis).**

d. Haga clic en el botón **Python** y seleccione **Notebooks.**

e. Se creará un nuevo **Notebook**, aquí podrá escribir código Python en cada una de las celdas, lo primero que realizará será importar las librerías que se usarán en este ejercicio. Para ello podrá guiarse de la siguiente imagen:

```
In [1]:
1 import arcpy
2 import pandas as pd
3 import matplotlib.pyplot as plt
4 from matplotlib.pyplot import figure
```

Una "librería" en programación es un conjunto de funciones y herramientas predefinidas que los programadores pueden utilizar para realizar tareas comunes sin tener que escribir código desde cero.

f. Una vez escriba el código de la anterior imagen, haga clic en el botón **Ejecutar (Run)** o **Shift + Enter** para ejecutar la primera línea de código.

g. Ahora usted va a declarar una variable que contenga el Shapefile con el cual se va a trabajar, para declarar escriba en la siguiente celda el código:

ciclo_ruta = r"D:\Documentos\ciclorruta\Ciclorruta.shp"

Si la ruta del Shapefile es distinta dentro de su equipo modifíquela en la línea de código.

h. Luego ejecute la celda con el comando **Shift + Enter.**

Paso 2: Examine las propiedades de la capa

a. Escriba la siguiente línea de código para ver el tipo de elemento geométrico que vamos a manipular.

```
In [4]:
1 desc = arcpy.Describe(ciclo_ruta)

In [5]:
1 desc.shapeType
Out[5]:
'Polygon'
```

b. Para visualizar el sistema de coordenadas que tiene la capa ejecute la siguiente línea de código.

```
In [6]:  1  dcoord = desc.spatialReference.name
         2  print(dcoord)

GCS_MAGNA
```

Paso 3: Convierta el shapefile a un dataframe

a. Ahora, convertirá el Shapefile a una tabla de Excel usando la función de la librería ArcPy **Table to Excel**. Para ello haga uso de la siguiente línea de código, adaptando la ruta a su equipo:

```
ciclo_ruta_xlsx = r"D:\Documentos\cicloruta_xlsx.xls"

salida = arcpy.TableToExcel_conversion(ciclo_ruta, ciclo_ruta_xlsx)
```

Un "DataFrame" es una estructura de datos tabular utilizada en programación y análisis de datos.

b. Ahora usará la librería **Pandas** para acceder al libro de Excel creado anteriormente y convertirá a DataFrame esa tabla, en la siguiente imagen puede encontrar los comandos para crear el Dataframe y la visualizar los datos de la tabla.

```
In [9]:  1  df_ciclo_ruta = pd.read_excel(ciclo_ruta_xlsx)

In [10]: 1  df_ciclo_ruta.head()
```

Out[10]:

	FID	CicTSuperf	CicCodigo	CicCIV	SHAPE_Leng	SHAPE_Area
0	0	2	34013476	7009283	0.000129	9.796160e-10
1	1	4	35003046	13001032	0.002548	3.222010e-08
2	2	2	35003047	10010469	0.003033	2.232427e-08
3	3	2	35003096	12002289	0.000248	2.611396e-09
4	4	2	34011616	12003236	0.001754	2.391483e-08

c. Si necesita visualizar la información del DataFrame, use el método **.info()**, así podrá ver un resumen de los campos que componen la tabla y el tipo de dato que almacena cada uno de ellos.

```
In [11]: 1 df_ciclo_ruta.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5659 entries, 0 to 5658
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   FID         5659 non-null   int64
 1   CicTSuperf  5659 non-null   int64
 2   CicCodigo   5659 non-null   int64
 3   CicCIV      5659 non-null   int64
 4   SHAPE_Leng  5659 non-null   float64
 5   SHAPE_Area  5659 non-null   float64
dtypes: float64(2), int64(4)
memory usage: 265.4 KB
```

d. Para generar una lista con todos los nombres de las columnas o campos de la tabla use la función **list()** junto al método **.columns** como se muestra en la siguiente imagen:

```
In [13]: 1 list(df_ciclo_ruta.columns)

Out[13]: ['FID', 'CicTSuperf', 'CicCodigo', 'CicCIV', 'SHAPE_Leng', 'SHAPE_Area']

In [14]: 1 df_ciclo_ruta.head()
```

Out[14]:

	FID	CicTSuperf	CicCodigo	CicCIV	SHAPE_Leng	SHAPE_Area
0	0	2	34013476	7009283	0.000129	9.796160e-10
1	1	4	35003046	13001032	0.002548	3.222010e-08
2	2	2	35003047	10010469	0.003033	2.232427e-08
3	3	2	35003096	12002289	0.000248	2.611396e-09
4	4	2	34011616	12003236	0.001754	2.391483e-08

e. Para generar un resumen con las estadísticas descriptivas de los campos con datos cuantitativos que tiene el DataFrame que creamos use el método **.describe()**.

```
In [15]:  1 df_ciclo_ruta.describe()
```

Out[15]:

	FID	CicTSuperf	CicCodigo	CicCIV	SHAPE_Leng	SHAPE_Area
count	5659.000000	5659.000000	5.659000e+03	5.659000e+03	5659.000000	5.659000e+03
mean	2829.000000	1.999470	2.880315e+07	9.039590e+06	0.001424	1.544283e-08
std	1633.756918	0.445114	1.081794e+07	5.073914e+06	0.001306	1.848808e-08
min	0.000000	1.000000	1.432920e+05	0.000000e+00	0.000099	4.825460e-10
25%	1414.500000	2.000000	2.413739e+07	7.007844e+06	0.000617	5.708484e-09
50%	2829.000000	2.000000	3.401172e+07	9.003744e+06	0.001058	1.039452e-08
75%	4243.500000	2.000000	3.401348e+07	1.200037e+07	0.001781	1.836080e-08
max	5658.000000	5.000000	3.500351e+07	9.104250e+07	0.015968	4.165671e-07

Paso 4: Cree una función para convertir datos

a. Como se pudo observar en pasos anteriores el campo **CicTSuperf**, que corresponde al tipo de superficie de las ciclorrutas, es un dato numérico, ahora lo convertiremos a tipo texto para trabajarlo como un dato cualitativo.

Para esto crearemos una función en Python que se llamará **número_a_texto** con esta capacidad, use el siguiente código para crear la función:

def numero_a_texto(numero):

return str(numero)

b. A continuación, aplicará la función creada a la columna del DataFrame **CicTSuperf** usando **lambda**, que es una forma sencilla de hacer llamados de funciones en Python, puede guiarse observando el código de la siguiente imagen:

```
In [16]:  1 def numero_a_texto(numero):
          2     return str(numero)
          3 df_ciclo_ruta['CicTSuperf'] = df_ciclo_ruta['CicTSuperf'].apply(lambda x: numero_a_texto(x))

In [17]:  1 df_ciclo_ruta.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5659 entries, 0 to 5658
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   FID         5659 non-null   int64
 1   CicTSuperf  5659 non-null   object
 2   CicCodigo   5659 non-null   int64
 3   CicCIV      5659 non-null   int64
 4   SHAPE_Leng  5659 non-null   float64
 5   SHAPE_Area  5659 non-null   float64
dtypes: float64(2), int64(3), object(1)
memory usage: 265.4+ KB
```

c. Una vez convertido el tipo de dato del campo **CicTSuperf**, visualizaremos la longitud de las ciclorrutas según el tipo de superficie, guardando esta información en un

DataFrame nuevo. Observe la siguiente imagen y así podrá crear la nueva tabla con el resumen de la información.

```
In [18]:
df_agupados_Tab = df_ciclo_ruta.groupby(['CicTSuperf'])['SHAPE_Leng'].sum().reset_index()
df_agupados_Tab = df_agupados_Tab.sort_values(by = 'SHAPE_Leng', ascending=False)
df_agupados_Tab
```

Out[18]:

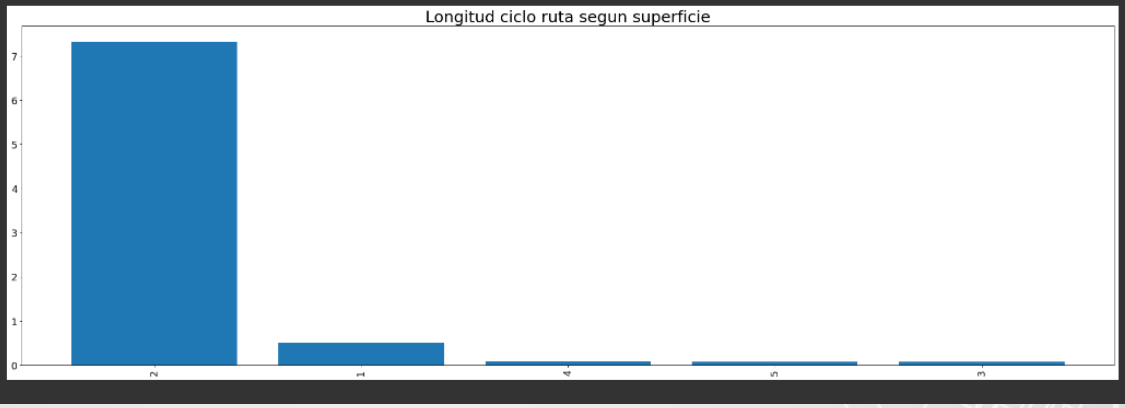
	CicTSuperf	SHAPE_Leng
1	2	7.319289
0	1	0.501383
3	4	0.088117
4	5	0.079529
2	3	0.069889

Paso 5: Creación de gráficos

a. Para crear gráficos desde Notebooks se usará la librería **Matplotlib** especializada en la generación de salidas graficas en Python. Primero generaremos un gráfico de barras visualizando la información obtenida en el paso anterior.

Para obtener el grafico de barras es necesario definir cuál será el dato categórico y el dato cuantitativo que será usado, en este caso son **CicTsuperf** y **SHAPE_Leng** respectivamente. Use la siguiente imagen como referencia para crear el gráfico.

```
In [21]:
%matplotlib inline
plt.figure(figsize=(32, 10), dpi=80)
plt.bar(df_agupados_Tab['CicTSuperf'],df_agupados_Tab['SHAPE_Leng'], label = 'Tipo Superficie')
plt.title ('Longitud ciclo ruta segun superficie', fontsize = 26 )
plt.xticks(fontsize = 16 , rotation = 90)
plt.yticks(fontsize = 16 )
plt.show()
```

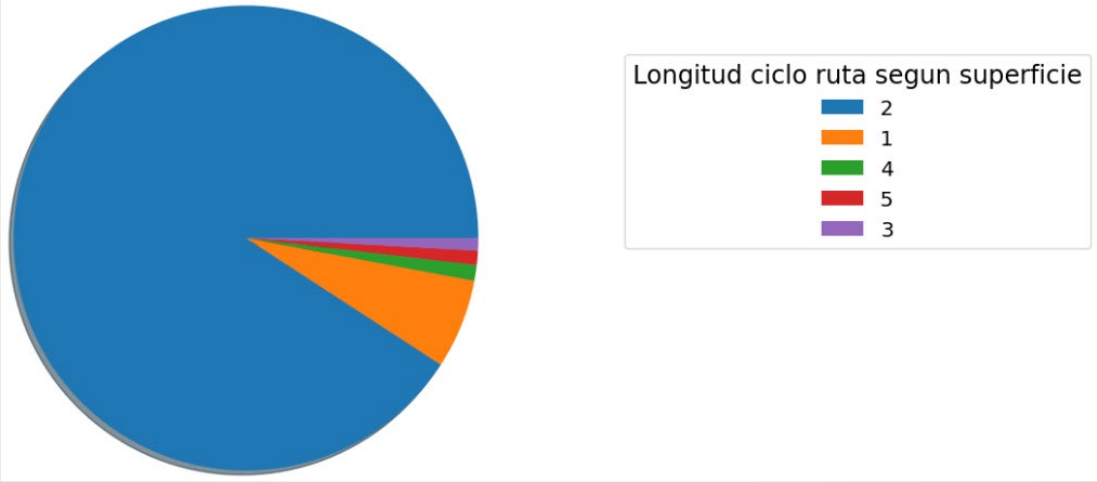


b. Para generar un gráfico circular o gráfico de torta a partir de la misma información que se encuentra en el gráfico de barras y así poder visualizar la proporción de la longitud de las vías según su superficie, utilizaremos el siguiente código:

```
In [22]:
labels_pie = df_agupados_Tab['CicTSuperf']
plt.pie(df_agupados_Tab['SHAPE_Leng'],
        labeldistance = 1.5,
        startangle = 0,
        shadow = True,
        rotatelabels = True,
        radius = 1.5)
#plt.subplot2grid((1,2),(0,0), colspan = 1)
plt.legend(title = 'Longitud ciclo ruta segun superficie',
           title_fontsize = 'xx-large',
           labels = labels_pie,
           bbox_to_anchor = (1.45,1),
           fontsize ='x-large'

          )
plt.draw()
plt.show()
```

c. Al ejecutar la anterior celda de código obtendrá el siguiente grafico circular:



Observe que la generación de gráficos con Python en un Notebook permite configurar las características propias de los elementos del gráfico, así como las características complementarias de las etiquetas de los ejes, títulos, leyendas, tipos y tamaños de letra, entre otros.

Paso 6: Consultas y exportación de resultados

a. Con los DataFrame también es posible hacer consultas para extraer información específica a partir de los datos de una o varias columnas. En esta ocasión será extraída únicamente la información del DataFrame cuyo tipo de superficie de la ciclorruta sea 1, para generar el DataFrame con esta condición use las siguientes líneas de código y ejecútelo.

```
In [24]:
1 superficie_1 = df_ciclo_ruta[df_ciclo_ruta['CicTSuperf'] == '1']
2 superficie_1.head()
3
```

Out[24]:

	FID	CicTSuperf	CicCodigo	CicCIV	SHAPE_Leng	SHAPE_Area
213	213	1	34013205	8001672	0.002741	3.341577e-08
302	302	1	34012891	5002311	0.000331	2.255409e-09
304	304	1	34012899	5008378	0.000656	4.855455e-09
306	306	1	34012896	5002289	0.000589	4.089827e-09
319	319	1	34011227	9001570	0.001170	1.119736e-08

b. El DataFrame generado debe tener menos registros, usted lo puede verificar usando el método **.info()**.

```
In [25]:
1 superficie_1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 346 entries, 213 to 5645
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   FID         346 non-null    int64
 1   CicTSuperf  346 non-null    object
 2   CicCodigo   346 non-null    int64
 3   CicCIV      346 non-null    int64
 4   SHAPE_Leng  346 non-null    float64
 5   SHAPE_Area  346 non-null    float64
dtypes: float64(2), int64(3), object(1)
memory usage: 18.9+ KB
```

c. Los resultados del DataFrame que cumple con la condición los puede exportar en distintos formatos como csv, xls, xlsx, incluso como Shapefile si este DataFrame tiene

la componente geográfica (Coordenadas). En este caso será exportado como un csv, para realizarlo use como guía la siguiente imagen:

```
In [26]:
# podemos exportar el resultado de la agrupación a excel o a csv
superficie_1.to_csv(r'D:\ncaro\OneDrive - Esri NOSA\Escritorio\idu\consulta.csv', index = True ,e
```

Paso 7: Aplique herramientas de geoprocesamiento usando python

a. Aplicaremos ahora funciones de geoprocesamiento usando la librería Arcpy, esta es la librería de ArcGIS que contiene todas las herramientas de geoprocesamiento, haremos uso de la herramienta **Buffer** directamente en la capa **Ciclorruta,** observe la siguiente imagen y use como referencia el código:

En el código de ejemplo es posible acceder a la capa “Ciclorruta” de esta manera cargándola previamente al proyecto en el cual se está trabajando, usted también puede realizarlo recurriendo a la variable ciclo_ruta anteriormente definida con el mismo Shapefile.

```
In [23]:
arcpy.analysis.PairwiseBuffer(
    in_features="Ciclorruta",
    out_feature_class=r"D:\ncaro\OneDrive - Esri NOSA\Documentos\ArcGIS\Packages\Notebook_IDU_a3d
    buffer_distance_or_field="1000 Meters",
    dissolve_option="NONE",
    dissolve_field=None,
    method="PLANAR",
    max_deviation="0 DecimalDegrees"
)
```

b. Ejecute la celda donde escribió el código para aplicar la herramienta buffer con la configuración realizada.

c. Finalmente, para revisar el resultado haga clic en la pestaña mapa y visualizará la nueva capa generada a partir de la ejecución de la herramienta, en esta evidenciará los círculos que definen el **área de influencia (Buffer)** dada por la distancia ingresada para cada ciclorruta que se encuentra en el Shapefile de entrada.

d. Guarde el proyecto de ArcGIS Pro.

Bogotá l (1) 650 1550 l Cll. 90 # 13 - 40
Esri.co

Más información:

En Colombia: entrenamiento@esri.co
https://esri.co/entrenamiento/cursos/

esri Colombia | ALCALDÍA MAYOR DE BOGOTÁ D.C. | INSTITUTO DE DESARROLLO URBANO | BOGOTÁ