Seminario de proyectos II

Unidad 2. Actividad 2A. Reporte. Aplicación de las herramientas de preprocesamiento

Claudio Daniel Pacheco Castro

Introducción

El presente documento muestra el desarrollo de la descarga de información así como de la limpieza de la misma para poder contar con datos que, posteriormente, se puedan utilizar para la definición de un modelo de clasificación.

Desarrollo de la actividad

Importar las librerías necesarias

In [ ]:
```
#Importar librerías
import pandas as pd
import os
import zipfile
import shutil
import ee
import requests
import zipfile
import io
from matplotlib import pyplot as plt
```

El primer paso consiste en definir el directorio de trabajo y, dentro de este, el de descarga de información.

In [ ]:
```
#Definir directorio de trabajo
#Directorio de trabajo. Si existe directorio D:/, se selecciona si no, C:/
try:
    os.chdir('D:/github/proyecto_infotec')
except:
    os.chdir('C:/users/claudio.pacheco/Documents/Github/proyecto_infotec')
#if os.path.exists('D:/'):
#    os.chdir('D:/github/proyecto_infotec')
#else:
#    os.chdir('C:/users/claudio.pacheco/Documents/Github/proyecto_infotec')
```

In [ ]:
```
#crear directorio de salida
if not os.path.exists('datos'):
    os.makedirs('datos')
```

Descarga de información a nivel manzana

Una vez definido el directorio y creada la carpeta de descarga, se procede a descargarla la información del INEGI. Para este caso, se utilizan los datos del Censo de Población y Vivienda 2020 a nivel manzana.

In [ ]:
```
#URL básica para descarga de archivos
url_basica="https://www.inegi.org.mx/contenidos/programas/ccpv/2020/datosabiertos/a
```

Con la URL básica, se obtiene la información de cada una de las entidades federativas:

In [ ]:
```
#Descargar archivos y descomprimir archivos

for entidad in range(1,33):
    #Si existe el archivo, no se descarga
    if os.path.exists("datos/conjunto_de_datos_ageb_urbana_01_cpv2020.csv"):
        pass
    else:
        if entidad<10:
            url=url_basica+"0"+str(entidad)+"_cpv2020_csv.zip"
        else:
            url=url_basica+str(entidad)+"_cpv2020_csv.zip"

#Descargar archivo
        os.system("curl -o datos/entidad_"+str(entidad)+".zip "+url)
#Descomprimir archivo
        with zipfile.ZipFile("datos/entidad_"+str(entidad)+".zip","r") as zip_ref:
            zip_ref.extractall("datos")
#Eliminar archivo zip
        os.remove("datos/entidad_"+str(entidad)+".zip")
```

In [ ]:
```
#Sacar de la carpeta conjunto_de_datos el archivo csv de cada entidad
for entidad in range(1,33):
    if entidad<10:
        shutil.move("datos/ageb_mza_urbana_0"+str(entidad)+"_cpv2020/conjunto_de_da
    else:
        shutil.move("datos/ageb_mza_urbana_"+str(entidad)+"_cpv2020/conjunto_de_dat
```

In [ ]:
```
#Eliminar las carpetas del directorio de trabajo si existen
for entidad in range(1,33):
    if entidad<10:
        if os.path.exists("datos/ageb_mza_urbana_"+"0"+str(entidad)+"_cpv2020"):
            shutil.rmtree("datos/ageb_mza_urbana_"+"0"+str(entidad)+"_cpv2020")
    else:
        if os.path.exists("datos/ageb_mza_urbana_"+str(entidad)+"_cpv2020"):
            shutil.rmtree("datos/ageb_mza_urbana_"+str(entidad)+"_cpv2020")
```

Al finalizar la descarga y descompresión de la información, se procede a unir los archivos en uno solo:

In [ ]:
```
#Pegar todos los archivos csv en una sola data frame
df=pd.DataFrame()
for entidad in range(1,33):
    if entidad<10:
        df_temp=pd.read_csv("datos/conjunto_de_datos_ageb_urbana_"+"0"+str(entidad)
    else:
        df_temp=pd.read_csv("datos/conjunto_de_datos_ageb_urbana_"+str(entidad)+"_c
    df=pd.concat([df,df_temp],axis=0)
    print("Se ha agregado la entidad "+str(entidad)," a la dataframe")

#Imprimir el número de registros con separadores de miles
print("Se concluyó la concatenación. El conjunto de datos cuenta con "+str(df.shape
df.columns=df.columns.str.lower()
#Eliminar todos los archivos csv de la carpeta datos
#archivos=[x for x in os.listdir('datos') if x.endswith('.csv')]
#for archivo in archivos:
#    os.remove("datos/"+archivo)
```

```
Se ha agregado la entidad 1  a la dataframe
Se ha agregado la entidad 2  a la dataframe
Se ha agregado la entidad 3  a la dataframe
Se ha agregado la entidad 4  a la dataframe
Se ha agregado la entidad 5  a la dataframe
Se ha agregado la entidad 6  a la dataframe
Se ha agregado la entidad 7  a la dataframe
Se ha agregado la entidad 8  a la dataframe
Se ha agregado la entidad 9  a la dataframe
Se ha agregado la entidad 10  a la dataframe
Se ha agregado la entidad 11  a la dataframe
Se ha agregado la entidad 12  a la dataframe
Se ha agregado la entidad 13  a la dataframe
Se ha agregado la entidad 14  a la dataframe
Se ha agregado la entidad 15  a la dataframe
Se ha agregado la entidad 16  a la dataframe
Se ha agregado la entidad 17  a la dataframe
Se ha agregado la entidad 18  a la dataframe
Se ha agregado la entidad 19  a la dataframe
Se ha agregado la entidad 20  a la dataframe
Se ha agregado la entidad 21  a la dataframe
Se ha agregado la entidad 22  a la dataframe
Se ha agregado la entidad 23  a la dataframe
Se ha agregado la entidad 24  a la dataframe
Se ha agregado la entidad 25  a la dataframe
Se ha agregado la entidad 26  a la dataframe
Se ha agregado la entidad 27  a la dataframe
Se ha agregado la entidad 28  a la dataframe
Se ha agregado la entidad 29  a la dataframe
Se ha agregado la entidad 30  a la dataframe
Se ha agregado la entidad 31  a la dataframe
Se ha agregado la entidad 32  a la dataframe
Se concluyó la concatenación. El conjunto de datos cuenta con 230 columnas y 1,68
3,504 registros
```

In [ ]:
```
#Mostrar todos los nombres de las columnas
print(df.columns.tolist())
```

```
['entidad', 'nom_ent', 'mun', 'nom_mun', 'loc', 'nom_loc', 'ageb', 'mza', 'pobto
t', 'pobfem', 'pobmas', 'p_0a2', 'p_0a2_f', 'p_0a2_m', 'p_3ymas', 'p_3ymas_f', 'p_
3ymas_m', 'p_5ymas', 'p_5ymas_f', 'p_5ymas_m', 'p_12ymas', 'p_12ymas_f', 'p_12ymas
_m', 'p_15ymas', 'p_15ymas_f', 'p_15ymas_m', 'p_18ymas', 'p_18ymas_f', 'p_18ymas_
m', 'p_3a5', 'p_3a5_f', 'p_3a5_m', 'p_6a11', 'p_6a11_f', 'p_6a11_m', 'p_8a14', 'p_
8a14_f', 'p_8a14_m', 'p_12a14', 'p_12a14_f', 'p_12a14_m', 'p_15a17', 'p_15a17_f',
'p_15a17_m', 'p_18a24', 'p_18a24_f', 'p_18a24_m', 'p_15a49_f', 'p_60ymas', 'p_60ym
as_f', 'p_60ymas_m', 'rel_h_m', 'pob0_14', 'pob15_64', 'pob65_mas', 'prom_hnv', 'p
nacent', 'pnacent_f', 'pnacent_m', 'pnacoe', 'pnacoe_f', 'pnacoe_m', 'pres2015',
'pres2015_f', 'pres2015_m', 'presoe15', 'presoe15_f', 'presoe15_m', 'p3ym_hli', 'p
3ym_hli_f', 'p3ym_hli_m', 'p3hlinhe', 'p3hlinhe_f', 'p3hlinhe_m', 'p3hli_he', 'p3h
li_he_f', 'p3hli_he_m', 'p5_hli', 'p5_hli_nhe', 'p5_hli_he', 'phog_ind', 'pob_afr
o', 'pob_afro_f', 'pob_afro_m', 'pcon_disc', 'pcdisc_mot', 'pcdisc_vis', 'pcdisc_l
eng', 'pcdisc_aud', 'pcdisc_mot2', 'pcdisc_men', 'pcon_limi', 'pclim_csb', 'pclim_
vis', 'pclim_haco', 'pclim_oaud', 'pclim_mot2', 'pclim_re_co', 'pclim_pmen', 'psin
d_lim', 'p3a5_noa', 'p3a5_noa_f', 'p3a5_noa_m', 'p6a11_noa', 'p6a11_noaf', 'p6a11_
noam', 'p12a14noa', 'p12a14noaf', 'p12a14noam', 'p15a17a', 'p15a17a_f', 'p15a17a_
m', 'p18a24a', 'p18a24a_f', 'p18a24a_m', 'p8a14an', 'p8a14an_f', 'p8a14an_m', 'p15
ym_an', 'p15ym_an_f', 'p15ym_an_m', 'p15ym_se', 'p15ym_se_f', 'p15ym_se_m', 'p15pr
i_in', 'p15pri_inf', 'p15pri_inm', 'p15pri_co', 'p15pri_cof', 'p15pri_com', 'p15se
c_in', 'p15sec_inf', 'p15sec_inm', 'p15sec_co', 'p15sec_cof', 'p15sec_com', 'p18ym
_pb', 'p18ym_pb_f', 'p18ym_pb_m', 'graproes', 'graproes_f', 'graproes_m', 'pea',
'pea_f', 'pea_m', 'pe_inac', 'pe_inac_f', 'pe_inac_m', 'pocupada', 'pocupada_f',
'pocupada_m', 'pdesocup', 'pdesocup_f', 'pdesocup_m', 'psinder', 'pder_ss', 'pder_
imss', 'pder_iste', 'pder_istee', 'pafil_pdom', 'pder_segp', 'pder_imssb', 'pafil_
ipriv', 'pafil_otrai', 'p12ym_solt', 'p12ym_casa', 'p12ym_sepa', 'pcatolica', 'pro
_crieva', 'potras_rel', 'psin_relig', 'tothog', 'hogjef_f', 'hogjef_m', 'pobhog',
'phogjef_f', 'phogjef_m', 'vivtot', 'tvivhab', 'tvivpar', 'vivpar_hab', 'vivparh_c
v', 'tvivparhab', 'vivpar_des', 'vivpar_ut', 'ocupvivpar', 'prom_ocup', 'pro_ocup_
c', 'vph_pisodt', 'vph_pisoti', 'vph_1dor', 'vph_2ymasd', 'vph_1cuart', 'vph_2cuar
t', 'vph_3ymasc', 'vph_c_elec', 'vph_s_elec', 'vph_aguadv', 'vph_aeasp', 'vph_agua
fv', 'vph_tinaco', 'vph_cister', 'vph_excsa', 'vph_letr', 'vph_drenaj', 'vph_nodre
n', 'vph_c_serv', 'vph_ndeaed', 'vph_dsadma', 'vph_ndacmm', 'vph_snbien', 'vph_ref
ri', 'vph_lavad', 'vph_hmicro', 'vph_autom', 'vph_moto', 'vph_bici', 'vph_radio',
'vph_tv', 'vph_pc', 'vph_telef', 'vph_cel', 'vph_inter', 'vph_stvp', 'vph_spmvpi',
'vph_cvj', 'vph_sinrtv', 'vph_sinltc', 'vph_sincint', 'vph_sintic']
```

In [ ]:
```
df
```

Out[ ]:

	entidad	nom_ent	mun	nom_mun	loc	nom_loc	ageb	mza	pobtot	pob
0	1	Aguascalientes	0	Total de la entidad Aguascalientes	0	Total de la entidad	0000	0	1425607	72
1	1	Aguascalientes	1	Aguascalientes	0	Total del municipio	0000	0	948990	48
2	1	Aguascalientes	1	Aguascalientes	1	Total de la localidad urbana	0000	0	863893	44
3	1	Aguascalientes	1	Aguascalientes	1	Total AGEB urbana	0017	0	2237	
4	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	1	170	
...	...	...	...	...	...	...	...	...	...	...
33839	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	18	2	
33840	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	19	0	
33841	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	20	0	
33842	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	21	2	
33843	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	800	26	

1683504 rows × 230 columns

Dado que únicamente nos interesan las variables de vivienda, se procede a eliminar las demás:

In [ ]:
```
df= df.iloc[:, list(range(8)) + list(range(177, len(df.columns)))]
print("El conjunto de datos cuenta con "+str(df.shape[1])+" columnas y "+str("{:,}"
df
```

```
El conjunto de datos cuenta con 61 columnas y 1,683,504 registros
```

Out[ ]:

	entidad	nom_ent	mun	nom_mun	loc	nom_loc	ageb	mza	vivtot	tvivh
0	1	Aguascalientes	0	Total de la entidad Aguascalientes	0	Total de la entidad	0000	0	463972	3866
1	1	Aguascalientes	1	Aguascalientes	0	Total del municipio	0000	0	313256	2669
2	1	Aguascalientes	1	Aguascalientes	1	Total de la localidad urbana	0000	0	286646	2462
3	1	Aguascalientes	1	Aguascalientes	1	Total AGEB urbana	0017	0	1288	6
4	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	1	82	
...	...	...	...	...	...	...	...	...	...	...
33839	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	18	3	
33840	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	19	1	
33841	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	20	0	
33842	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	21	1	
33843	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	800	11	

1683504 rows × 61 columns

In [ ]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1683504 entries, 0 to 33843
Data columns (total 61 columns):
 #   Column       Non-Null Count    Dtype 
---  ------       --------------    ----- 
 0   entidad      1683504 non-null  int64 
 1   nom_ent      1683504 non-null  object
 2   mun          1683504 non-null  int64 
 3   nom_mun      1683504 non-null  object
 4   loc          1683504 non-null  int64 
 5   nom_loc      1683504 non-null  object
 6   ageb         1683504 non-null  object
 7   mza          1683504 non-null  int64 
 8   vivtot       1683504 non-null  int64 
 9   tvivhab      1683504 non-null  object
 10  tvivpar      1683504 non-null  object
 11  vivpar_hab   1683504 non-null  object
 12  vivparh_cv   1683504 non-null  object
 13  tvivparhab   1683504 non-null  object
 14  vivpar_des   1683504 non-null  object
 15  vivpar_ut    1683504 non-null  object
 16  ocupvivpar   1683504 non-null  object
 17  prom_ocup    1683504 non-null  object
 18  pro_ocup_c   1683504 non-null  object
 19  vph_pisodt   1683504 non-null  object
 20  vph_pisoti   1683504 non-null  object
 21  vph_1dor     1683504 non-null  object
 22  vph_2ymasd   1683504 non-null  object
 23  vph_1cuart   1683504 non-null  object
 24  vph_2cuart   1683504 non-null  object
 25  vph_3ymasc   1683504 non-null  object
 26  vph_c_elec   1683504 non-null  object
 27  vph_s_elec   1683504 non-null  object
 28  vph_aguadv   1683504 non-null  object
 29  vph_aeasp    1683504 non-null  object
 30  vph_aguafv   1683504 non-null  object
 31  vph_tinaco   1683504 non-null  object
 32  vph_cister   1683504 non-null  object
 33  vph_excsa    1683504 non-null  object
 34  vph_letr     1683504 non-null  object
 35  vph_drenaj   1683504 non-null  object
 36  vph_nodren   1683504 non-null  object
 37  vph_c_serv   1683504 non-null  object
 38  vph_ndeaed   1683504 non-null  object
 39  vph_dsadma   1683504 non-null  object
 40  vph_ndacmm   1683504 non-null  object
 41  vph_snbien   1683504 non-null  object
 42  vph_refri    1683504 non-null  object
 43  vph_lavad    1683504 non-null  object
 44  vph_hmicro   1683504 non-null  object
 45  vph_autom    1683504 non-null  object
 46  vph_moto     1683504 non-null  object
 47  vph_bici     1683504 non-null  object
 48  vph_radio    1683504 non-null  object
 49  vph_tv       1683504 non-null  object
 50  vph_pc       1683504 non-null  object
```

```
 51  vph_telef    1683504 non-null  object
 52  vph_cel      1683504 non-null  object
 53  vph_inter    1683504 non-null  object
 54  vph_stvp     1683504 non-null  object
 55  vph_spmvpi   1683504 non-null  object
 56  vph_cvj      1683504 non-null  object
 57  vph_sinrtv   1683504 non-null  object
 58  vph_sinltc   1683504 non-null  object
 59  vph_sincint  1683504 non-null  object
 60  vph_sintic   1683504 non-null  object
dtypes: int64(5), object(56)
memory usage: 796.3+ MB
```

Eliminar agregados

Estos conjuntos de información cuentan con agregados por entidad, municipio, localidad y manzana. Para este caso, únicamente nos interesa la información a nivel manzana, por lo que se procede a eliminar los agregados:

In [ ]:
```
#Eliminar agregados de entidad.
df=df[df['mun']!=0]
#Eliminar agregados de municipio.
df=df[df['loc']!=0]
#Eliminar agregados de localidad.
df=df[df['ageb']!='0000']
#Eliminar agregados de manzana.
df=df[df['mza']!=0]
```

In [ ]:
```
df
```

Out[ ]:

	entidad	nom_ent	mun	nom_mun	loc	nom_loc	ageb	mza	vivtot	tvivha
4	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	1	82	5
5	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	2	83	5
6	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	3	84	5
7	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	4	84	5
8	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	5	68	4
...	...	...	...	...	...	...	...	...	...	
33839	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	18	3	
33840	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	19	1	
33841	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	20	0	
33842	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	21	1	
33843	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	800	11	

1611448 rows × 61 columns

In [ ]:
```
print("El conjunto de datos cuenta con "+str(df.shape[1])+" columnas y "+str("{:,}"
```

```
El conjunto de datos cuenta con 61 columnas y 1,611,448 registros
```

Transformación de variables a numérico

Para poder trabajar con los datos, es necesario que las variables sean numéricas. En algunos casos, por el nivel de desagregación, las variables cuentan con un "*". Estos se sustituirán por NA. Para mayor facilidad, se creará una copia de la df para no tener que correr el código en su totalidad.

In [ ]:
```
df_limpia=df.copy()
```

In [ ]:
```
#transformar variables de la 8 en adelante a numéricas
df_limpia.iloc[:,8:]=df_limpia.iloc[:,8:].apply(pd.to_numeric, errors='coerce')
```

```
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_19888\541626342.py:2: Deprec
ationWarning: In a future version, `df.iloc[:, i] = newvals` will attempt to set t
he values inplace instead of always setting a new array. To retain the old behavio
r, use either `df[df.columns[i]] = newvals` or, if columns are non-unique, `df.ise
titem(i, newvals)`
  df_limpia.iloc[:,8:]=df_limpia.iloc[:,8:].apply(pd.to_numeric, errors='coerce')
```

In [ ]:
```
df_limpia
```

Out[ ]:

	entidad	nom_ent	mun	nom_mun	loc	nom_loc	ageb	mza	vivtot	tvivha
4	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	1	82	54
5	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	2	83	52
6	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	3	84	55
7	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	4	84	57
8	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	5	68	48
...	...	...	...	...	...	...	...	...	...	
33839	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	18	3	1
33840	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	19	1	Na
33841	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	20	0	0
33842	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	21	1	1
33843	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	800	11	7

1611448 rows × 61 columns

Eliminar información de manzanas que no tienen viviendas o que son NAs

In [ ]:
```
#Eliminar manzanas sin viviendas o con NaN
df_limpia=df_limpia[df_limpia['tvivparhab']!=0]
df_limpia=df_limpia[df_limpia['tvivparhab'].notna()]
df_limpia
```

Out[ ]:

	entidad	nom_ent	mun	nom_mun	loc	nom_loc	ageb	mza	vivtot	tvivha
4	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	1	82	54
5	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	2	83	52
6	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	3	84	55
7	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	4	84	57
8	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	5	68	48
...	...	...	...	...	...	...	...	...	...	
33828	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	7	8	6
33831	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	10	6	4
33834	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	13	13	11
33838	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	17	5	3
33843	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	800	11	7

1305275 rows × 61 columns

Construcción de la clave geoestadística

Para poder realizar el vínculo información geográfica, se debe construir la clave geoestadística cocatenando la clave de entidad, la de municipio, la de localidad, ageb y manzana. Al final, se debe obtener una variable de tipo string con una longitud de 16 caracteres.

In [ ]:
```
df_limpia['cvegeo'] = df_limpia['entidad'].astype(str).str.zfill(2) + df_limpia['mu
```

In [ ]:
```
df_limpia
```

Out[ ]:

	entidad	nom_ent	mun	nom_mun	loc	nom_loc	ageb	mza	vivtot	tvivha
4	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	1	82	54
5	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	2	83	52
6	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	3	84	55
7	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	4	84	57
8	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017	5	68	48
...	...	...	...	...	...	...	...	...	...	
33828	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	7	8	6
33831	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	10	6	4
33834	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	13	13	11
33838	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	17	5	3
33843	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123	800	11	7

1305275 rows × 62 columns

In [ ]:
```
#Colocar la variable cvegeo al principio
df_limpia=df_limpia[['cvegeo']+df_limpia.columns.tolist()[:-1]]
```

In [ ]:
```
#Mostrar todas las columnas
pd.set_option('display.max_columns', None)
df_limpia
```

Out[ ]:

	cvegeo	entidad	nom_ent	mun	nom_mun	loc	nom_loc	ageb
4	0100100010017001	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017
5	0100100010017002	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017
6	0100100010017003	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017
7	0100100010017004	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017
8	0100100010017005	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017
...	...	...	...	...	...	...	...	...
33828	3205800010123007	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123
33831	3205800010123010	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123
33834	3205800010123013	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123
33838	3205800010123017	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123
33843	3205800010123800	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123

1305275 rows × 62 columns

Se realiza la comprobación de la longitud de la variable cvegeo:

In [ ]:
```
#Obtener resumen de largo de cvegeo
df_limpia['cvegeo'].str.len().value_counts()
```

Out[ ]:
```
16    1305275
Name: cvegeo, dtype: int64
```

Cálculo de porcentajes respecto a tvivparhab

Para poder calcular un indicador en conjunto, en este caso, una aproximación del rezago habitacional, se calculan todas las variables como porcentaje respecto a la variable *tvivparhab* la cual representa el número total de viviendas particulares habitadas en la manzana.

In [ ]:
```
#Dividir todas las variables entre tvivparhab y multiplicar por 100.
df_limpia.iloc[:,20:]=df_limpia.iloc[:,20:].div(df_limpia['tvivparhab'],axis=0).mul
```

In [ ]:
```
df_limpia
```

Out[ ]:

	cvegeo	entidad	nom_ent	mun	nom_mun	loc	nom_loc	ageb
4	0100100010017001	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017
5	0100100010017002	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017
6	0100100010017003	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017
7	0100100010017004	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017
8	0100100010017005	1	Aguascalientes	1	Aguascalientes	1	Aguascalientes	0017
...	...	...	...	...	...	...	...	...
33828	3205800010123007	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123
33831	3205800010123010	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123
33834	3205800010123013	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123
33838	3205800010123017	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123
33843	3205800010123800	32	Zacatecas	58	Santa María de la Paz	1	Santa María de la Paz	0123

1305275 rows × 62 columns

Estadística descriptiva

In [ ]:
```
#Hacer describe con las variables de la 20 en adelante. No usar notación científica
pd.set_option('display.float_format', lambda x: '%.3f' % x)
df_limpia.iloc[:,20:].describe()
```

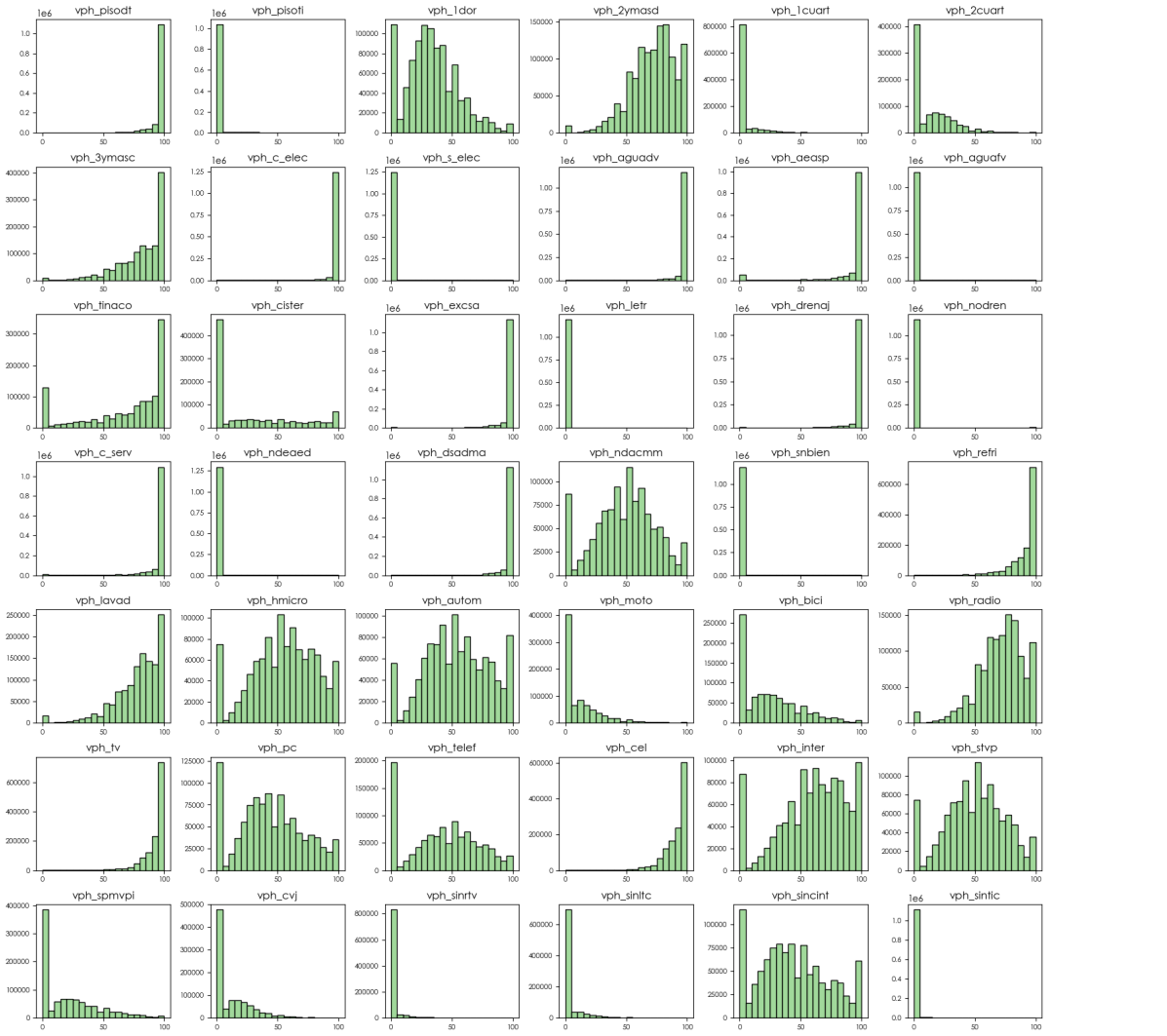
Out[ ]:

	vph_pisodt	vph_pisoti	vph_1dor	vph_2ymasd	vph_1cuart	vph_2cuart	vph_3ymasc	
count	1291833.000	1085389.000	971784.000	1211288.000	980280.000	863506.000	1239892.000	1
mean	97.241	1.279	33.237	71.357	3.888	14.170	81.501	
std	7.736	7.283	20.777	18.631	10.815	17.536	19.218	
min	0.000	0.000	0.000	0.000	0.000	0.000	0.000	
25%	100.000	0.000	20.000	60.000	0.000	0.000	71.429	
50%	100.000	0.000	31.579	73.333	0.000	9.091	85.714	
75%	100.000	0.000	45.455	84.615	0.000	23.913	100.000	
max	100.000	100.000	100.000	100.000	100.000	100.000	100.000	

Histogramas

In [ ]:
```
plt.rcParams['font.family'] = 'Century Gothic'
#Título de la figura
df_limpia.iloc[:,20:].hist(figsize=(20,20),bins=20, color='#a1d99b', edgecolor='bla
#Titulo de la figura
plt.suptitle("Histogramas de las variables de vivienda", fontsize=20, fontweight='b

#Fuente
plt.figtext(0.99, 0.01, 'Fuente: INEGI. Censo de Población y Vivienda 2020. Microda
```

Out[ ]:
```
Text(0.99, 0.01, 'Fuente: INEGI. Censo de Población y Vivienda 2020. Microdatos. h
ttps://www.inegi.org.mx/programas/ccpv/2020/#Microdatos')
```



Función para descarga de imágenes satelitales

Para el proyecto, se utilizarán imágenes satelitales. Éstas se obtendrán de la plataforma de Google Earth Engine. Para poder descargarlas, se define un directorio en donde se descará la

información.

In [ ]:
```
#Crear carpeta para guardar las imágenes
if not os.path.exists('datos/imagenes'):
    os.makedirs('datos/imagenes')
```

Una vez que se cuenta con el directorio, se procede a definir la función que descargará la información. Esta función recibe como parámetros una lista de coordenadas que representan el área de interés, el rango de fechas en el que se desea obtener los datoz y el nomnbre del estado al que pertenecen las coordenadas:

In [ ]:
```
def download_landsat_image(polygon_coords, start_date, end_date, state_name):
    #  Se incializa la API de Earth Engine
    ee.Initialize()

    # Polígono de interés

    polygon = ee.Geometry.Polygon(polygon_coords)

    # Colleción de imágenes de Landsat 8
    landsat = ee.ImageCollection('LANDSAT/LC08/C01/T1_TOA') \
                .filterDate(start_date, end_date) \
                .filterBounds(polygon) \
                .sort('CLOUD_COVER') \
                .first()

    # Parámetros de exportación.
    export_params = {
        'image': landsat,
        'description': state_name +'Landsat_image',
        'scale': 30,
        'region': polygon
    }

    # Se genera la URL para descargar la imagen
    download_url = landsat.getDownloadURL(export_params)

    # Descarga de imagen
    r = requests.get(download_url)
    z = zipfile.ZipFile(io.BytesIO(r.content))
    z.extractall(os.path.join(os.getcwd(),"datos/imagenes",state_name))
```

Ejemplo: Oaxaca

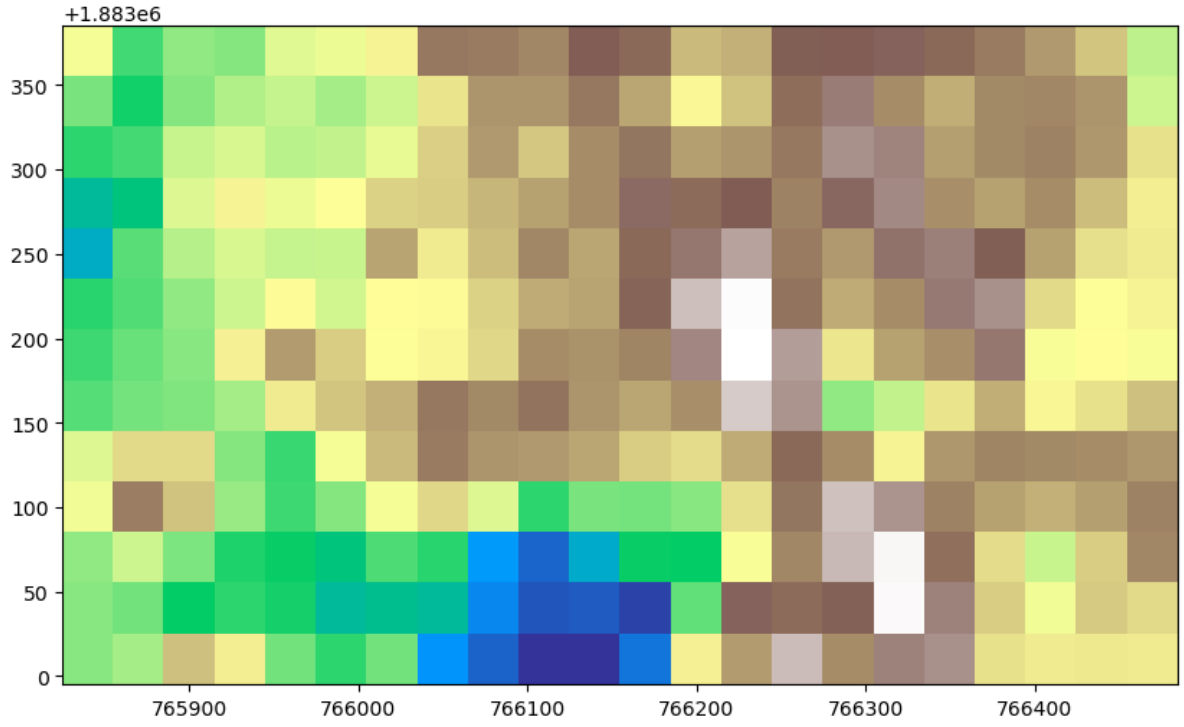
Si bien el preprocesamiento de las imágenes aún no se ha realizado, se puede observar que la información se descarga correctamente.

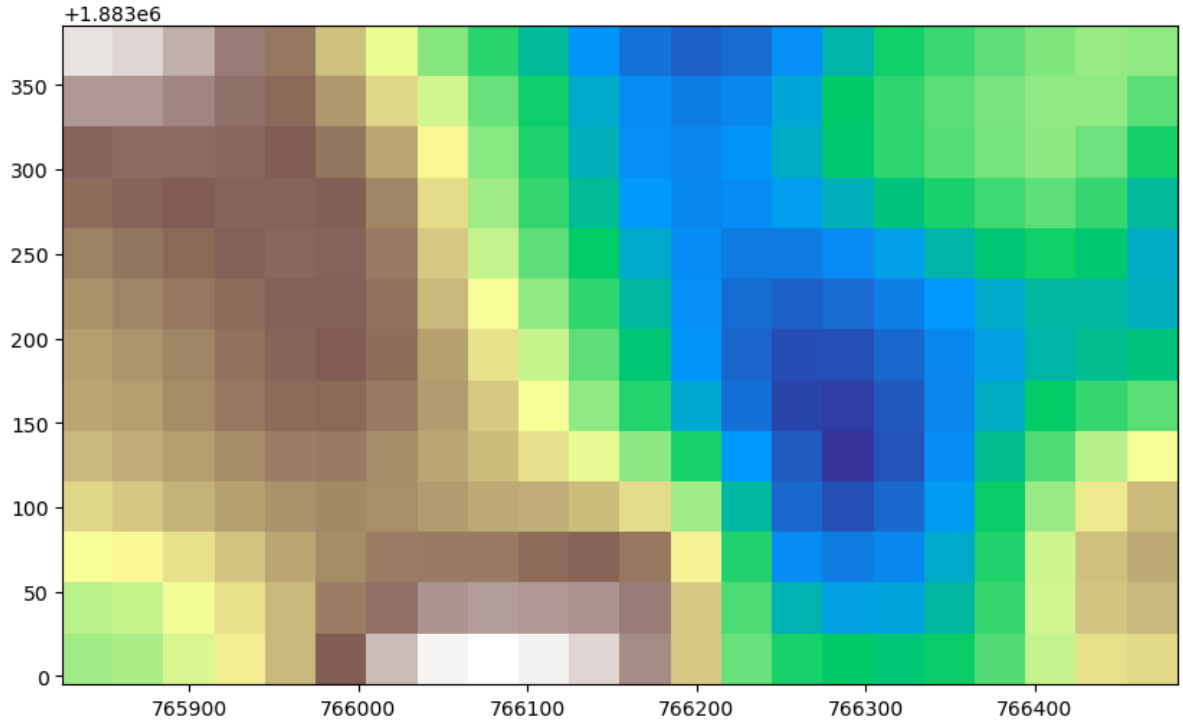
In [ ]:
```
polygon_coords = [[[-96.503, 17.016], [-96.503, 17.019], [-96.497, 17.019], [-96.49
start_date = '2020-01-01'
end_date = '2020-12-31'

download_landsat_image(polygon_coords, start_date, end_date, 'Oaxaca')
```

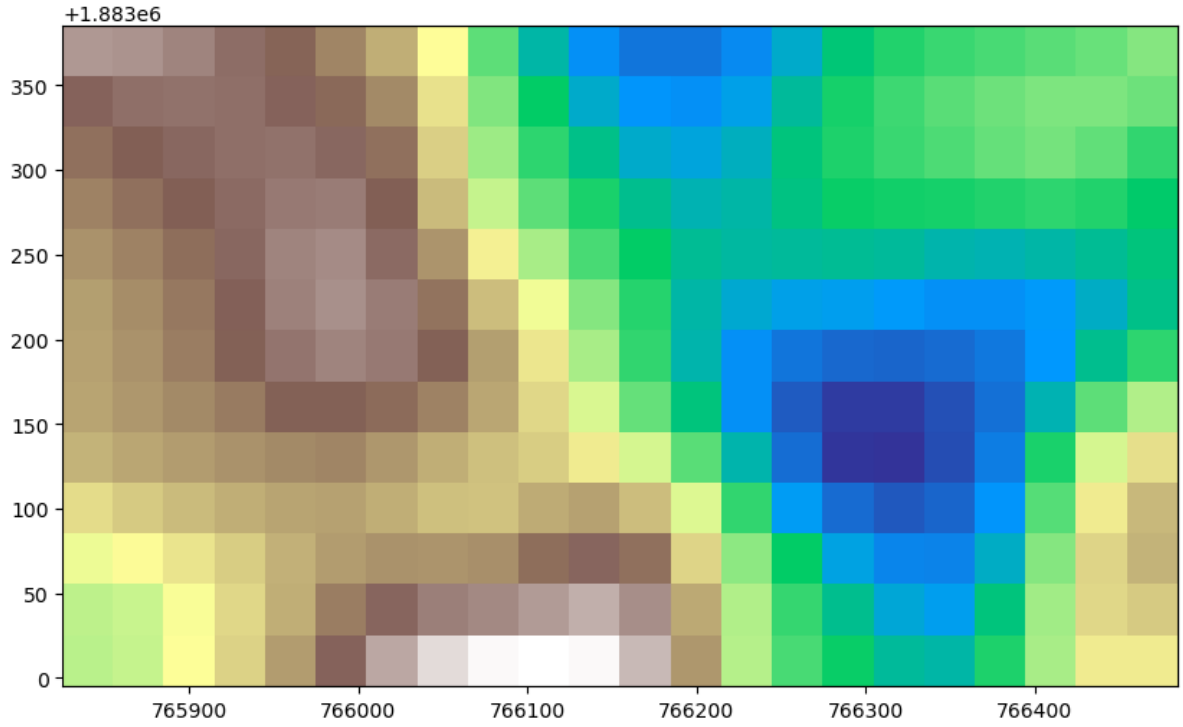
In [ ]:
```
#Enlistar todas las imágenes
imagenes=[x for x in os.listdir('datos/imagenes/Oaxaca') if x.endswith('.tif')]
imagenes
```

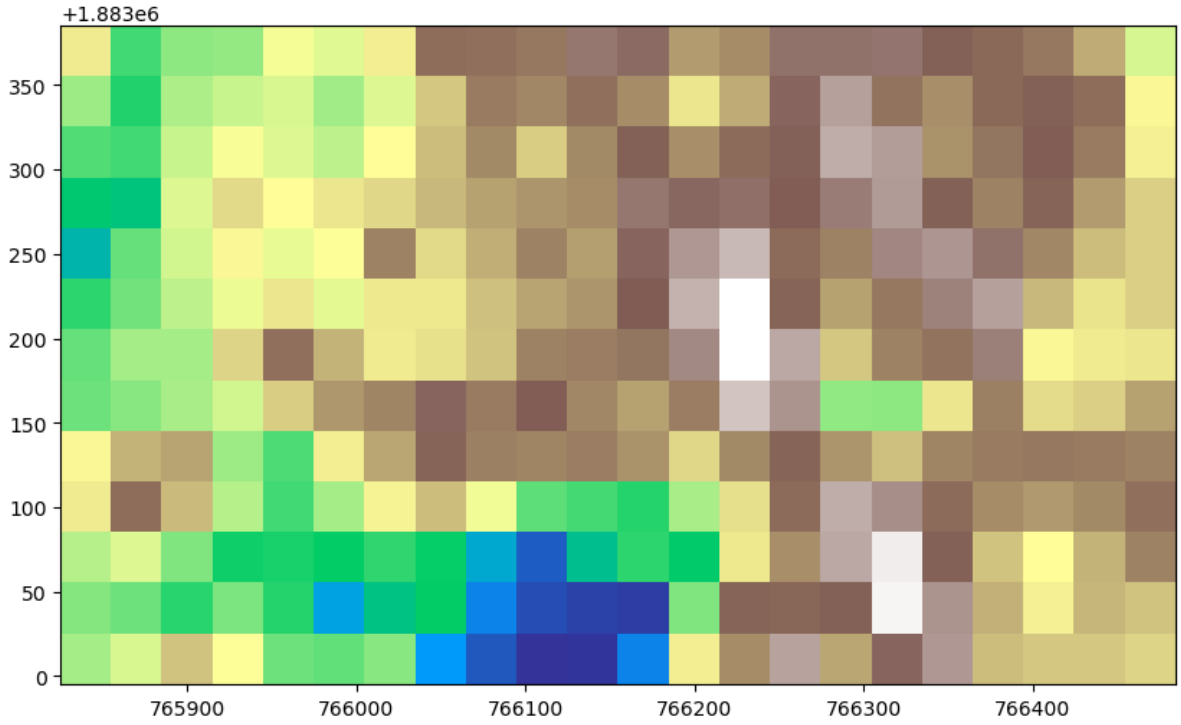
Out[ ]:
```
['LC08_024048_20200423.B1.tif',
 'LC08_024048_20200423.B10.tif',
 'LC08_024048_20200423.B11.tif',
 'LC08_024048_20200423.B2.tif',
 'LC08_024048_20200423.B3.tif',
 'LC08_024048_20200423.B4.tif',
 'LC08_024048_20200423.B5.tif',
 'LC08_024048_20200423.B6.tif',
 'LC08_024048_20200423.B7.tif',
 'LC08_024048_20200423.B8.tif',
 'LC08_024048_20200423.B9.tif',
 'LC08_024048_20200423.BQA.tif']
```

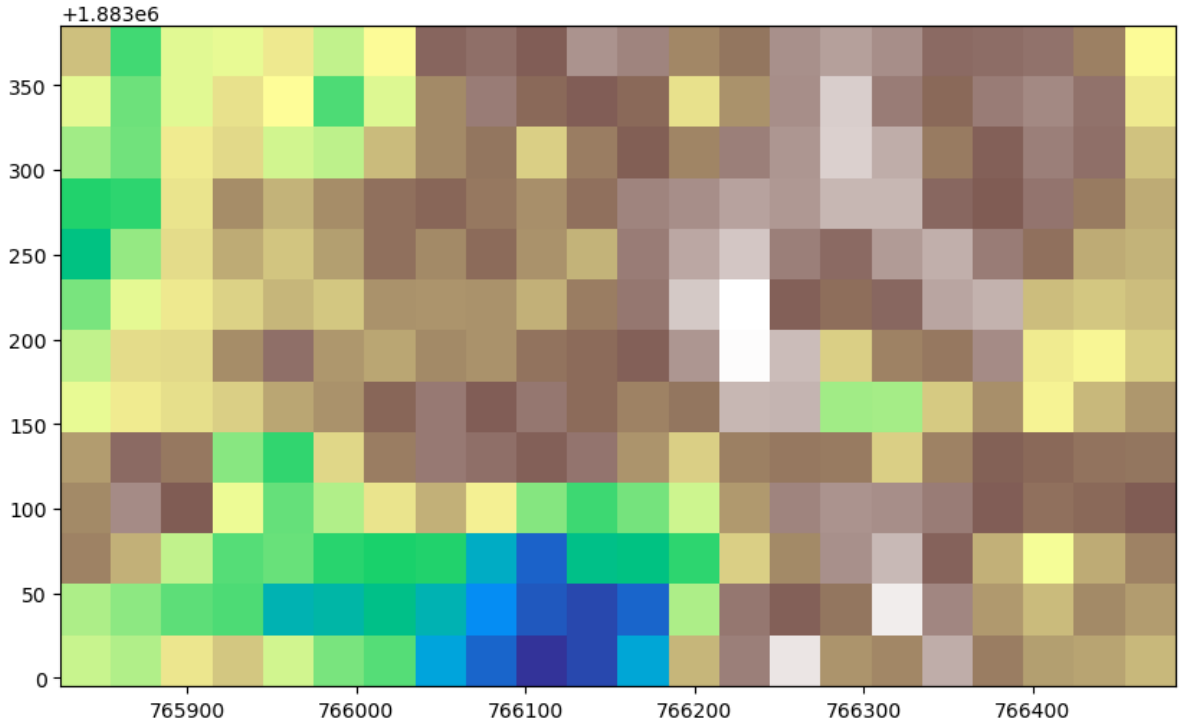
In [ ]:
```
#Graficar todas las imágenes
for imagen in imagenes:
    raster = rasterio.open("datos/imagenes/Oaxaca/"+imagen)
    fig, ax = plt.subplots(figsize=(10, 10))
    show(raster, ax=ax, cmap='terrain')
    plt.show()
```

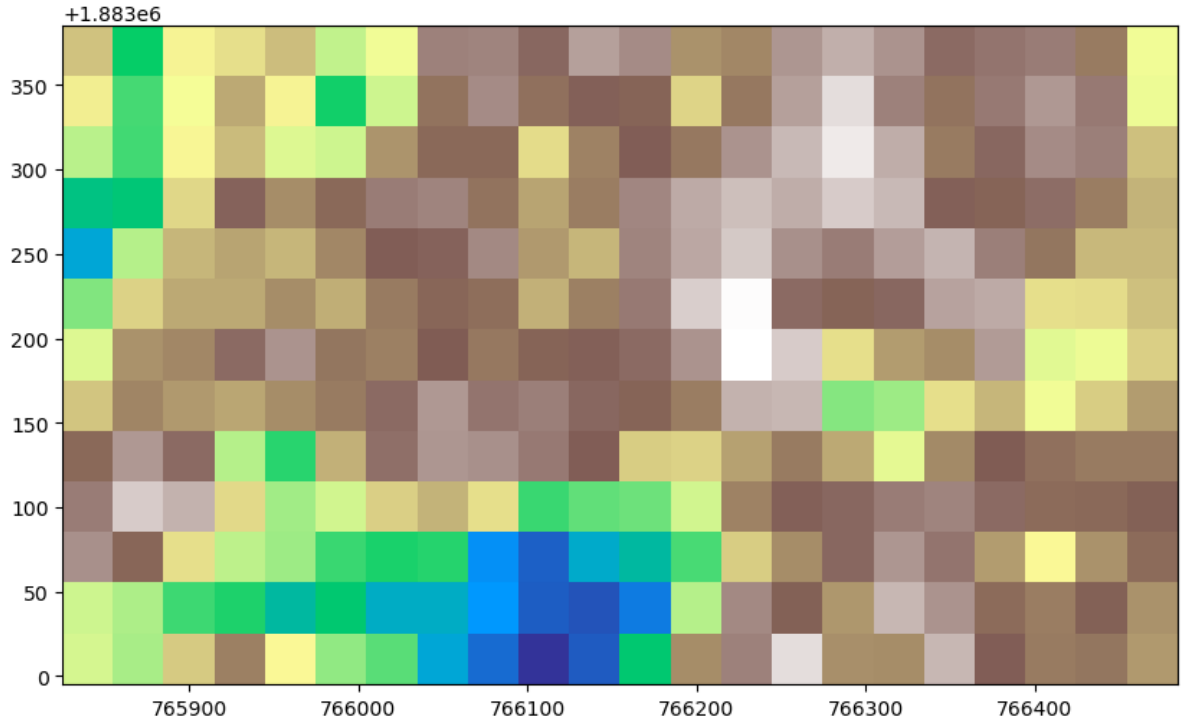


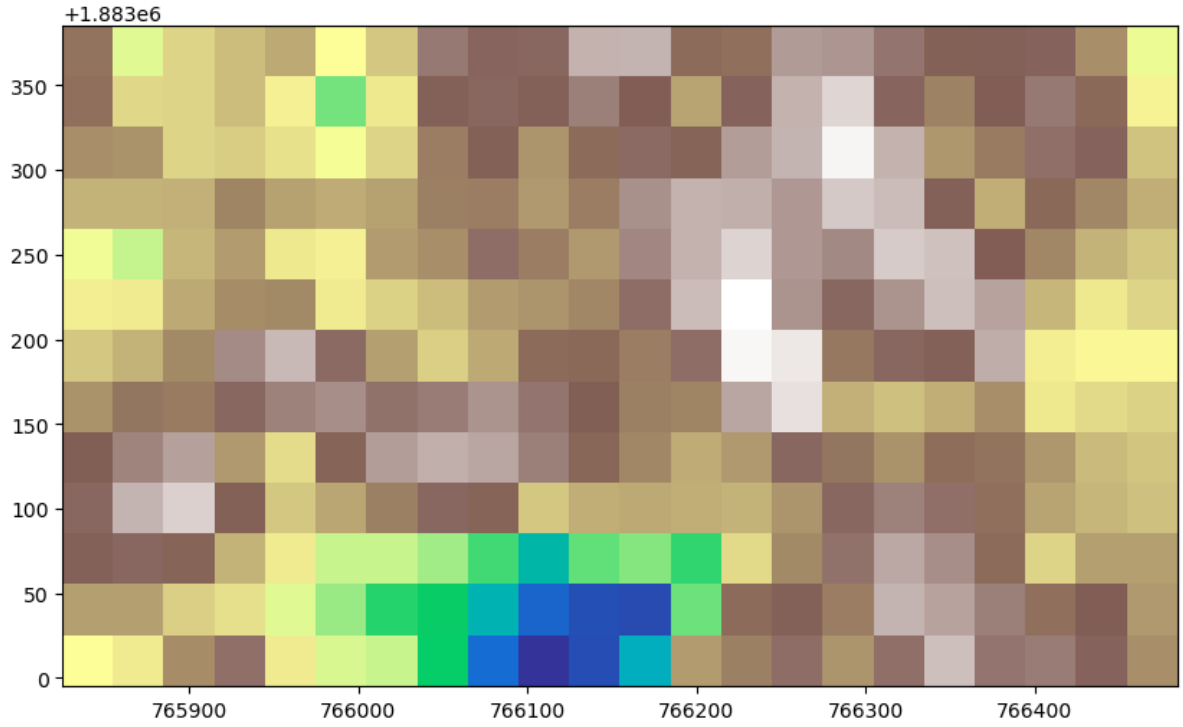

+1.883e6
350
300
250
200
150
100
50
0
765900
766000
766100
766200
766300
766400

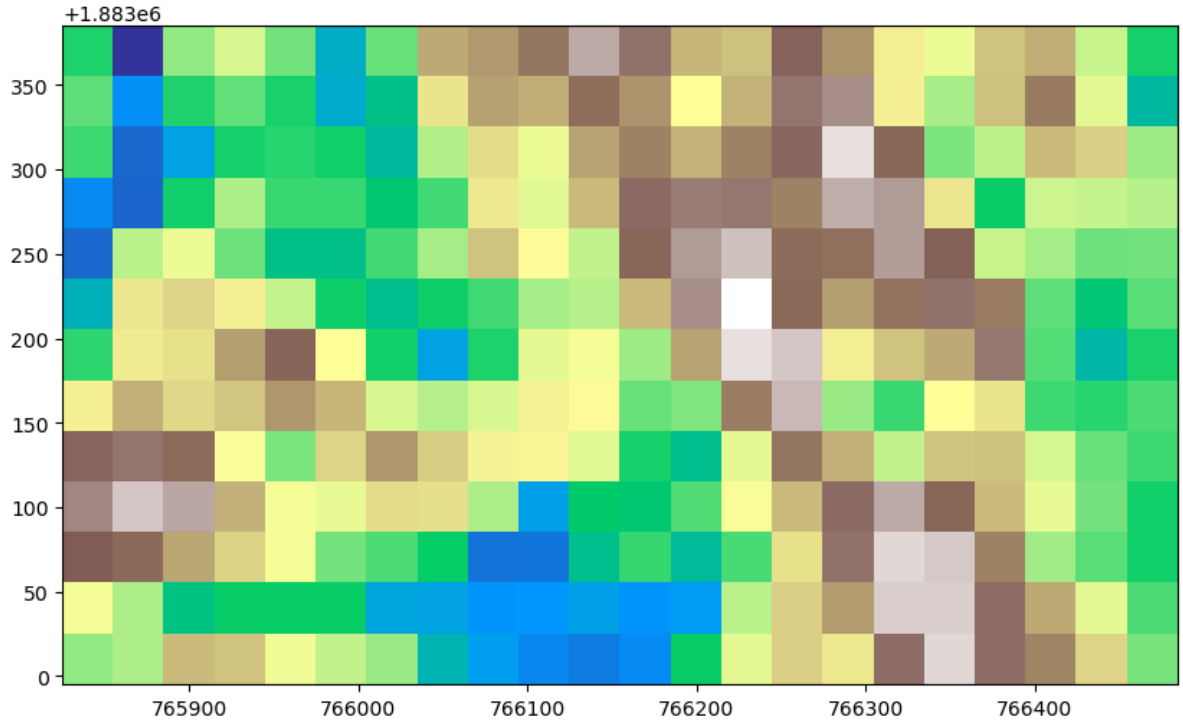

+1.883e6
350
300
250
200
150
100
50
0
765900
766000
766100
766200
766300
766400

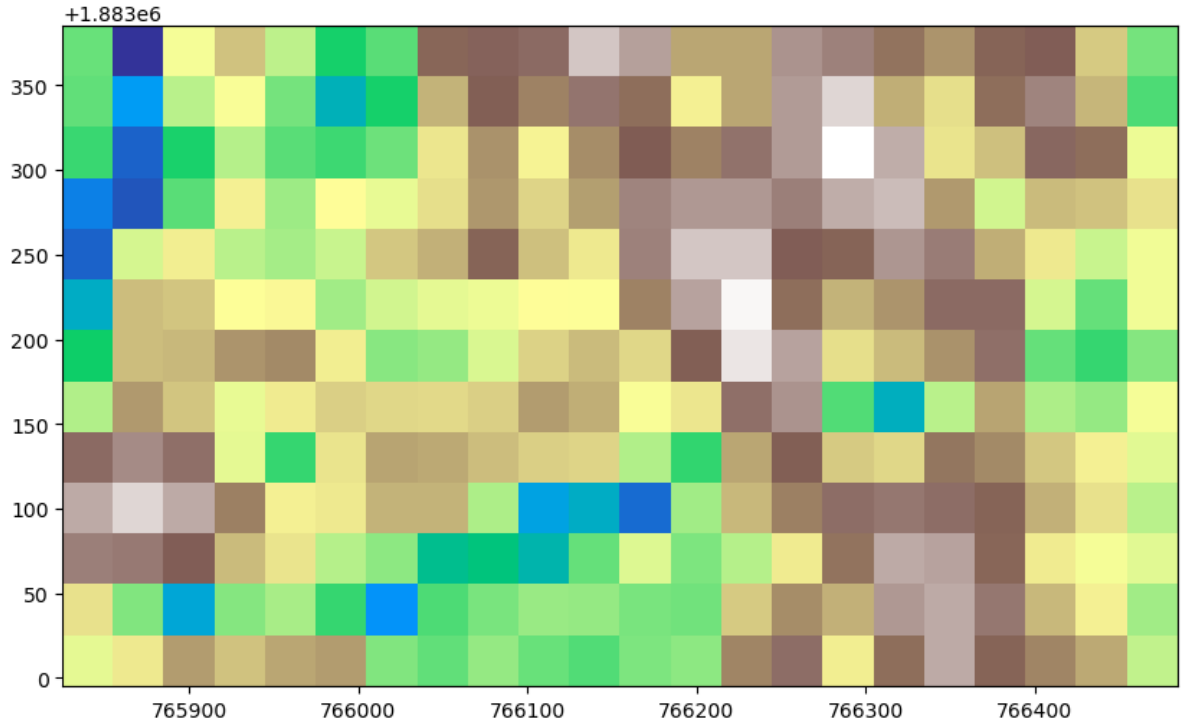

+1.883e6
350
300
250
200
150
100
50
0
765900
766000
766100
766200
766300
766400

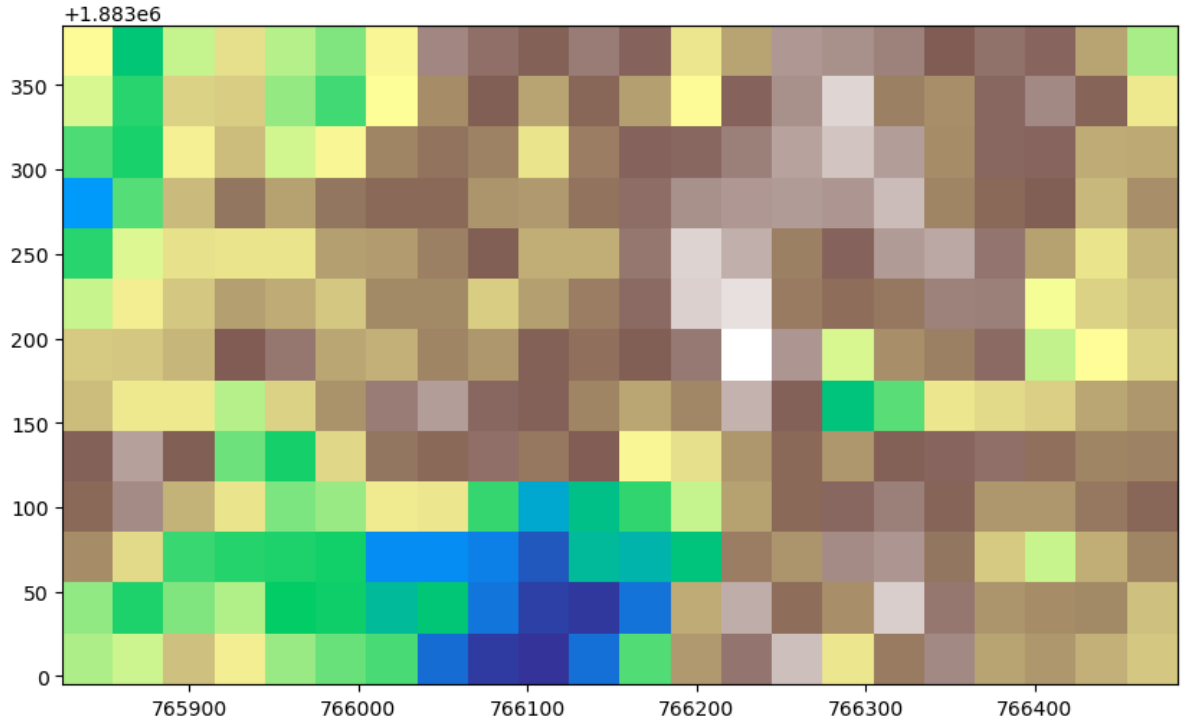

+1.883e6
350
300
250
200
150
100
50
0
765900
766000
766100
766200
766300
766400

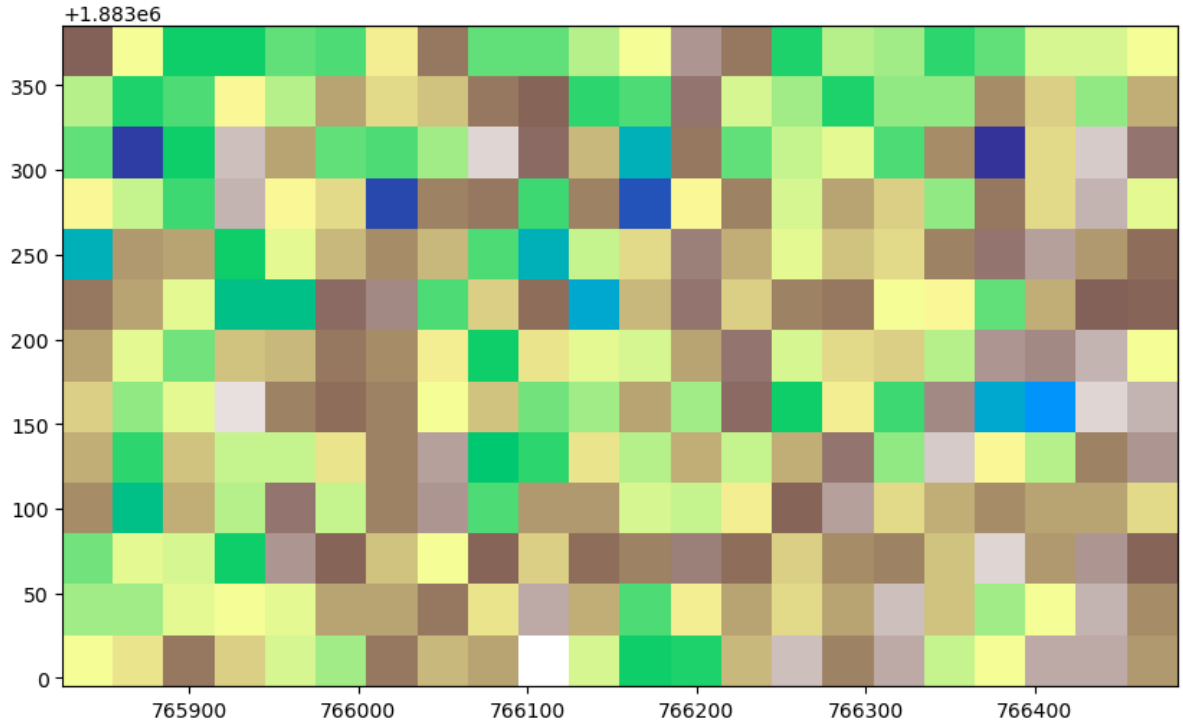

+1.883e6
350
300
250
200
150
100
50
0
765900
766000
766100
766200
766300
766400

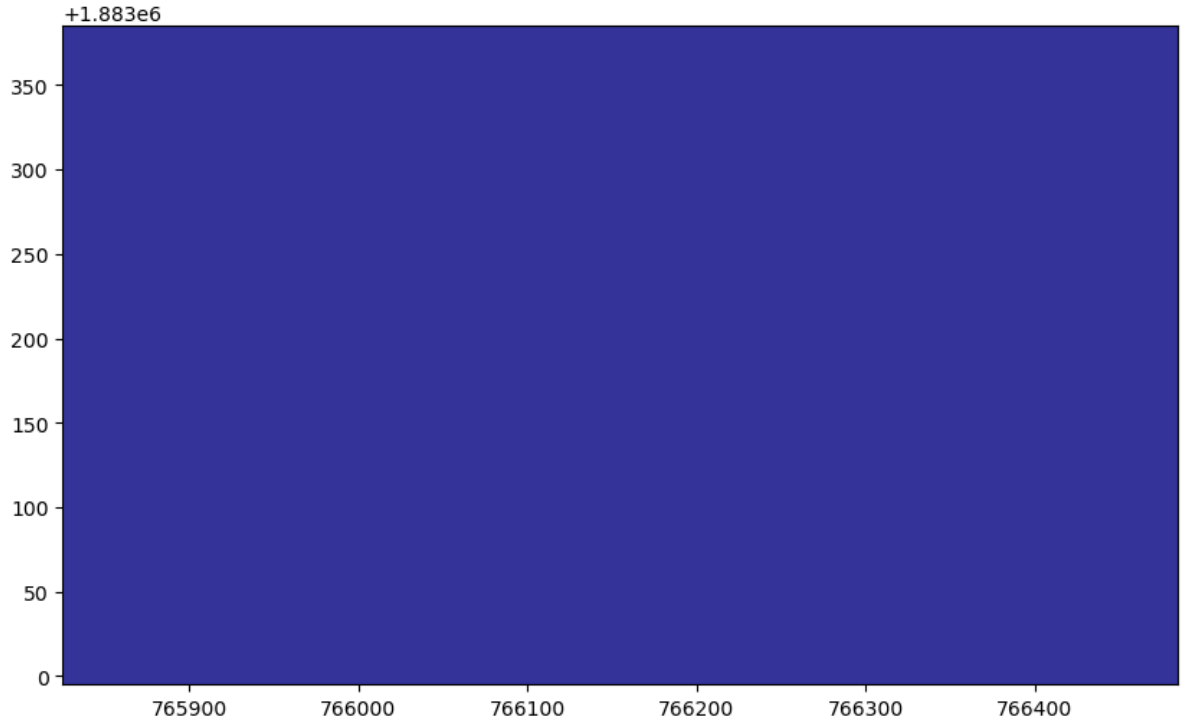

+1.883e6
350
300
250
200
150
100
50
0
765900
766000
766100
766200
766300
766400


+1.883e6
350
300
250
200
150
100
50
0
765900
766000
766100
766200
766300
766400


+1.883e6
350
300
250
200
150
100
50
0
765900
766000
766100
766200
766300
766400


+1.883e6
350
300
250
200
150
100
50
0
765900
766000
766100
766200
766300
766400


+1.883e6
350
300
250
200
150
100
50
0
765900
766000
766100
766200
766300
766400


+1.883e6
350
300
250
200
150
100
50
0
765900
766000
766100
766200
766300
766400