



# GEPANDAS

Visualizing Geospatial Data with Python



ناھید نعمتی کوتنائی (تیسا)  
دکتری جغرافیا و برنامه‌ریزی شهری  
مدرس دانشگاه

 **Dr.nemati.K**  
 **@Nemati\_k**  
 ۰۹۱۱۲۲۳۰۷۹۸



محمدطاهر طاهرپور  
دانشجوی ارشد مدیریت شهری  
دانشگاه تهران

 **mttahrpoor**  
 **@mtahrpoor**  
 ۰۹۳۳۶۱۴۴۹۴۷



## فهرست مطالب:

۳.....	Geopandas
۳.....	تحلیل داده‌های فضایی با پایتون
۵.....	Geoplotlib
۵.....	نصب Geopandas
۶.....	استفاده از نوت بوک ArcGIS pro
۷.....	انجام یه پروژه در geopandas
۷.....	گام اول: آوردن کتابخانه‌های مورد نیاز با import
۸.....	گام دوم: خواندن فایلها با geopandas.read_file()
۹.....	گام سوم: پلات گرفتن از فایلها GeoDataFrame.plot()
۱۱.....	استفاده از legend_kwds
۱۳.....	گام چهارم: اضافه کردن لایه‌های دیگه به پلات قبلی
۱۳.....	دیدن دو تا پلات کنار هم
۱۵.....	دیدن دو یا چند تا لایه در یک پلات
۱۶.....	گام پنجم: سیستم تصویر یا Projection
۱۷.....	گام ششم: کاربردهای Geopandas در پردازش داده‌های جغرافیایی
۱۸.....	Intersections
۱۹.....	Identity
۱۹.....	Union
۱۹.....	Symmetric Differences
۱۹.....	Difference
۱۹.....	Dissolve
۲۰.....	Buffer
۲۱.....	Centroid
۲۱.....	گام هفتم: محاسبه مساحت
۲۲.....	گام هشتم: محدود کردن فیلدهای اطلاعاتی
۲۲.....	گام نهم: خروجی گرفتن از لایه to_file()
۲۲.....	گام دهم: وبی کردن لایه با explore()

## Geopandas

تو این جزوه با Geopandas که یکی از کتابخانه‌های مهم پایتون هست و برای تحلیل و مصورسازی داده‌های جغرافیایی استفاده میشه آشنا میشید. واسه نوشتن جزوه از منابع زیر استفاده کردیم.

۱- [https://geopandas.org/en/stable/getting\\_started/introduction.html](https://geopandas.org/en/stable/getting_started/introduction.html)

۲- Introduction to Visualizing Geospatial Data with Python GeoPandas

ابهاماتی که تو این سایتها داشتیم رو هم از هوش مصنوعی peo پرسیدیم و برامون رفعش کرد:

<https://poe.com/>

## تحلیل داده‌های فضایی با پایتون


کتابخانه‌های زیادی برای تحلیل فضایی داده‌های جغرافیایی وجود دارن ولی تمرکز اصلی ما واسه تحلیل داده‌های جغرافیایی و مصورسازیشون روی این سه تا کتابخانه هست.


🚀 **Geopandas:** کتابخانه GeoPandas یک ابزار قدرتمند برای تحلیل و مدیریت داده‌های مکانی در محیط برنامه‌نویسی پایتون هست. این کتابخانه بر پایه کتابخانه Pandas ساخته شده و قابلیت‌های مربوط به تجزیه و تحلیل داده‌های جدولی رو با قابلیت‌های مکانی ترکیب کرده. با کمک GeoPandas، می‌تونی داده‌های مکانی رو به صورت ساده و کارآمد در پایتون مدیریت کنی. این کتابخانه قابلیت‌هایی مثل خوندن و نوشتن فرمت‌های داده‌های مکانی مختلف (مثل Shapefile و GeoJSON) رو داره. علاوه بر اون، میتونه عملیات مکانی مثل انجام عملیات بر روی ژئومتری‌ها، ترسیم نقشه‌ها و تجزیه و تحلیل مکانی داده‌ها رو هم انجام بده. GeoPandas از کتابخانه‌های معروف دیگه‌ای مثل Fiona، Shapely و Fiona و Matplotlib هم استفاده می‌کنه. این کتابخانه‌ها به GeoPandas کمک می‌کنن تا قابلیت‌های پیشرفته‌تری مثل ترسیم نقشه‌ها و انجام عملیات مکانی پیچیده انجام بده. استفاده از GeoPandas برای انواع پروژه‌های مربوط به داده‌های مکانی مثل تجزیه و تحلیل زمین‌شناسی، حوزه محیط‌زیست، برنامه‌ریزی شهری و سیستم‌های اطلاعات جغرافیایی (GIS) استفاده میشه.

🚀 **Geoplotlib:** یک کتابخانه کارآمد برای تصویرسازی داده‌هاست که قابلیت‌های گسترده‌ای برای ایجاد تصاویر استاتیک و با کیفیت برای نمایش داده‌های مکانی فراهم می‌کنه. این کتابخانه انواع مختلفی از نمودارها رو پشتیبانی می‌کنه، مثل نمودارهای پراکندگی، نمودارهای خطی، نمودارهای توزیع و نمودارهای میله‌ای که می‌تونن برای نمایش داده‌های مکانی سفارشی بشن. مت‌پلاتلیب می‌تونه به تنهایی یا در ترکیب با بقیه کتابخانه‌های مکانی برای ایجاد تصاویر مکانی جذاب و با کیفیت استفاده بشه.

🚀 **Rasterio:** یک کتابخانه پایتون هست که برای خوندن، نوشتن و پردازش داده‌های رستری استفاده می‌شه. Rasterio بر اساس کتابخانه GDAL ساخته شده و امکانات گسترده‌ای برای کار با داده‌های رستری فراهم می‌کنه. با استفاده از Rasterio، می‌تونی تصاویر رستری رو به عنوان آرایه‌های ناپیوسته مدیریت کنی، اطلاعات جغرافیایی مربوط به اونها رو بخونی و روشون عملیات محاسباتی انجام بدی و نتایج رو به فرمت‌های مختلفی خروجی بگیری. Rasterio معمولاً در کنار کتابخانه GeoPandas برای کار با داده‌های مکانی استفاده می‌شه، طوری که اطلاعات هندسی (مثل مرزها و موقعیت مکانی) رو با تصاویر رستری مرتبط می‌کنه.

معمولاً از کتابخانه‌های تحلیل و مصورسازی داده‌های جغرافیایی در صورتی استفاده می‌شود که نیاز به کار با داده‌های مکانی داریم ولی به ArcGIS/ ArcGIS pro یا QGIS دسترسی نداریم. والا بیشتر قابلیت‌های این کتابخانه‌ها رو میشه با Arcpy و یا pyQGIS هم انجام داد.

**Arcpy**  به کتابخانه پایتون هست و توسط شرکت Esri توسعه داده شده که بهمون برای انجام تحلیل‌های فضایی روی داده‌های جغرافیایی کمک میکنه. میشه بهش توی ArcGIS و ArcGIS Pro دسترسی داشت.

**PyQGIS**  به رابط برنامه‌نویسی در پایتون برای دسترسی و کنترل QGIS (Quantum GIS) هست. با استفاده از PyQGIS میتونی عملیات مکانی و تحلیل داده‌ها رو از طریق اسکریپت‌های پایتون انجام بدی.

با اینحال دونه‌ستنه سه تا کتابخانه geopandas, geopy و rasterio میتونه در ترکیب با Arcpy و یا pyQGIS برای تحلیل و مصورسازی داده‌های جغرافیایی مفید باشه. واسه همین تو این جزوه Geopandas رو با هم یاد می‌گیریم و تو جزوه‌های بعدی در مورد آموزش Geoplotlib و Rasterio براتون مطلب می‌ذاریم.

ساختر اصلی داده در GeoPandas، **geopandas.GeoDataFrame** هست که یک زیرکلاس از `pandas.DataFrame` محسوب میشه و می‌تونه ستون‌های ژئومتری یا هندسی (نقطه، خط، سطح و ...) رو ذخیره کنه و عملیات مکانی انجام بده.

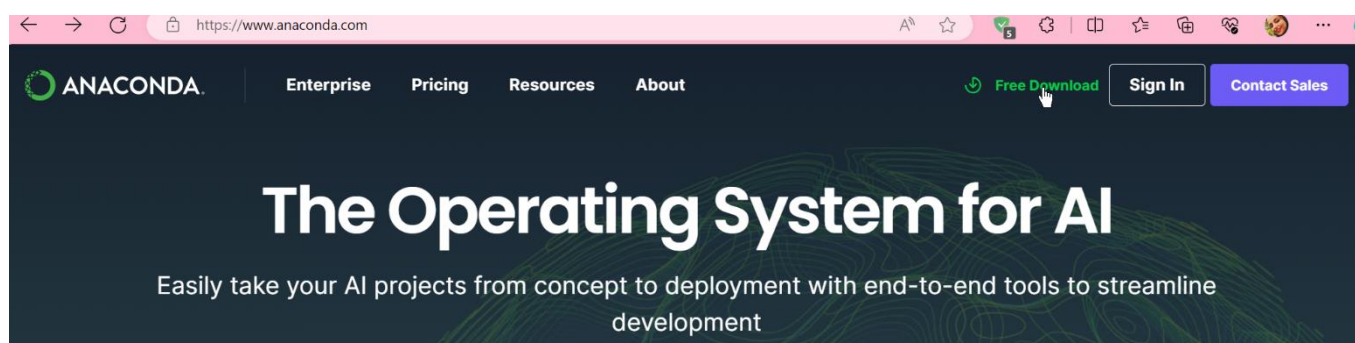
**geopandas.GeoSeries** هم یک زیرکلاس از `pandas.Series` هست که با ژئومتریها سروکار داره. بنابراین، `GeoDataFrame` ترکیبی از `pandas.Series` با داده‌های سنتی (عددی، بولینی، متن و غیره) و `geopandas.GeoSeries` با ژئومتریها (نقاط، چندضلعی‌ها و غیره) میشه.

هر `GeoSeries` دارای ویژگی **GeoSeries.crs** هست که مخفف `Coordinate Reference System` یا سیستم مختصات میشه.

## نصب Geopandas

اگه دسترسی به ArcGIS نداري میتونی از `anaconda prompt` برای نوشتن کدهات استفاده کنی. برو تو سایت `anaconda.com` و از بخش `Free Download` و روی گزینه دانلود کلیک کن.

[Free Download | Anaconda](https://www.anaconda.com/free-download)

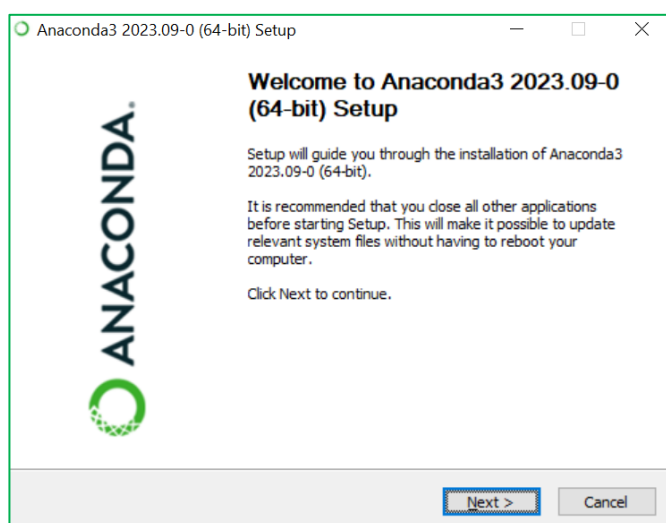


گزینه دانلود مربوط به سیستم رو بزنی که فایل برات دانلود شه.

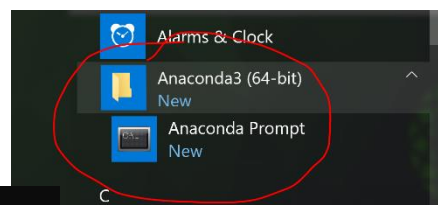


روی فایل دانلود شده کلیک کن.

مراحل نصبش ساده هست. مسیر نصب رو براش تعیین کن و تا انتها `next` بزنی و روی گزینه `install` کلیک کن. صبر کن تا نصب شه.



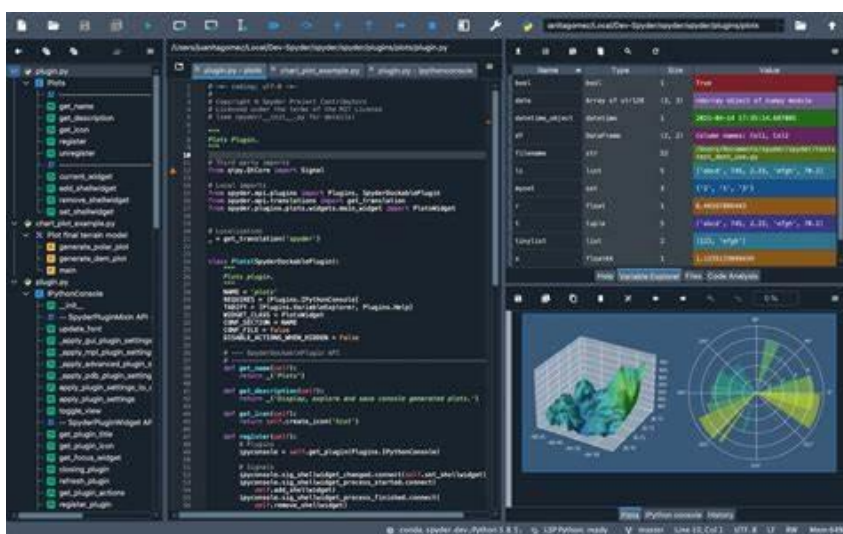
بعد از نصب باید Anaconda prompt رو باز کنی و یه سری کتابخونه روش نصب کنی.



```
(base) PS C:\Users\Geeks> conda list geopandas
# packages in environment at C:\Users\Geeks\anaconda3:
#
# Name                      Version      Build Channel
geopandas                   0.9.0        py_1
geopandas-base              0.9.0        py_1
(base) PS C:\Users\Geeks>
```

تو پنجره مقابل دستور  
conda install geopandas  
بنویس و اینتر بزنی که نصبش  
کنه.

با دستور pip install descartes<sup>۱</sup> افزونه descartes رو هم نصب کن.



محیط برنامه نویسی آناکوندا به این  
شکل هست که میتونی توش کد بنویسی،  
جدول اطلاعاتی لایه ها رو ببینی و یا از لایه ها  
پلات تهیه کنی و ...

ما تو ایران به راحتی به ArcGIS Pro  
دسترسی داریم میتونی به جای آناکوندا از  
نوت بوک ArcGIS Pro

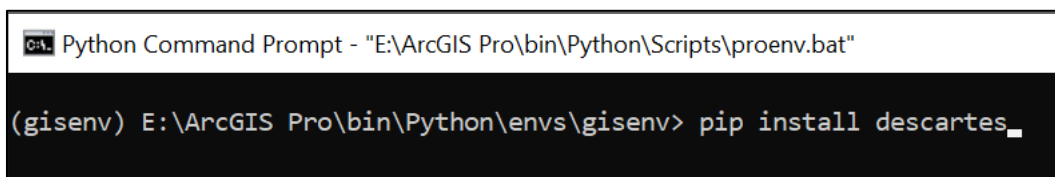
## استفاده از نوت بوک ArcGIS pro

واسه استفاده از geopandas تو نوت بوک ArcGIS Pro بسته به ورژنی که داری به مشکل بخوری مثلاً  
ممکنه نمودارها رو بهت نشون نده. واسه همین توصیه میکنم که کدها رو تو Python Command Prompt بنویسی.  
برای اینکار حواستون باشه که فولدر Shp که واسه لایه ها تمرینی برات گذاشتم رو ببری تو مسیر نصب نرم افزار کپی  
کنی. برای من مسیر نصبش اینجاست: (جزوه شماره ۱۰ رو ببین).

E:\ArcGIS Pro\bin\Python\envs\gisenv

برای نصب descartes هم از مسیر نصب نرم افزار، Python Command Prompt رو بیار و جلوی خطی که میده  
این عبارت رو بنویس و اینتر بزنی که کار نصب رو کامل کنه (جزوه شماره ۹ رو ببین).

pip install descartes



<sup>۱</sup> کتابخونه descartes در واقع یه افزونه برای کتابخانه Matplotlib برای ترسیم نمودارهای دوبعدی با استفاده از مختصات کارتزی هست.  
descartes مجموعه ای از توابع رو ارائه می ده که بهت کمک میکنه که ژئومتریها یا اشکال هندسی مثل خطوط، چندضلعی ها و دایره ها و ... رو  
به راحتی روی نمودارهای Matplotlib بکشی.

## انجام یه پروژه در geopandas

برای انجام پروژه‌ها معمولاً مراحل زیر باید طی شه:

- ۱- اول کتابخانه‌های موردنیاز رو باید وارد کنی **(با دستور import)**.
- ۲- فایل‌های مورد نظرت که میتونه فرمت‌های مختلفی مثل GeoJSON، Shapefile و ... داشته باشن رو باید بخونی **(با دستور geopandas.read\_file())**.
- ۳- از فایل‌ها پلات بگیر **(با دستور geopandas.plot())**.
- ۴- بتونی لایه‌های دیگه رو هم به پلات اضافه کنی (اینجا نیاز به کتابخانه **matplotlib.pyplot** داری که دستور **fig, ax = plt.subplots()** رو باهاش بنویسی).
- ۵- بتونی به لایه‌ها سیستم مختصات یا سیستم تصویر بدی یا سیستم تصویری که از قبل دارن رو ببینی یا سیستم مختصاتشون رو به چیز دیگه‌ای تبدیل کنی **(با دستور crs و دستور to\_crs())**.
- ۶- بتونی یه سری تحلیل روی لایه‌ها انجام بدی **(جلوتر چند تا تحلیل مثل intersection, union, buffer و ... روی لایه‌ها رو برات گذاشتم)**.
- ۷- بتونی مساحت عوارض رو در لایه پلیگونی بدست بیاری و به هکتار تبدیلشون کنی **(با دستور area)**.
- ۸- بتونی یه سری از فیلدها رو که بهش نیاز نداری از جدول اطلاعاتی پاک کنی یا یه سری فیلدها رو نگه داری و بقیه دیده نشن. **(با دستور fields\_to\_keep = [])**.
- ۹- بتونی از لایه‌ها خروجی که می‌خوای رو تهیه کنی که تو فودر مورد نظرت به صورت یه فایل ذخیره شه **(با دستور to\_file())**.
- ۱۰- بتونی لایه‌ها رو و بی کنی و در اختیار بقیه قرار بدی **(با دستور explore)**. برای اینکه چند تا لایه رو با هم و بی کنی به کتابخانه **folium**<sup>۲</sup> نیاز پیدا میکنی که دستور **folium.LayerControl().add\_to(m)** رو براش بنویسی).

در ادامه این ۱۰ تا گام رو با مرحله به مرحله با مثال واقعی انجام میدیم.

### گام اول: آوردن کتابخانه‌های مورد نیاز با import

برای آوردن **geopandas** تو نوت بوک ژوپیتر عبارت زیر رو تایپ کن و شیفت اینتر بزنی.

```
import geopandas as gpd
```

```
help(gpd)
```

تو خط بعدی تایپ کن `help(gpd)` که دستورش رو برات بیاره.

Help on package geopandas:

NAME

geopandas

PACKAGE CONTENTS

\_compat

\_config

\_decorator

\_vectorized

\_version

array

base

<sup>۲</sup> یک کتابخانه قدرتمند در پایتون هست که برای نمایش داده‌های جغرافیایی به صورت تعاملی و زیبا استفاده میشه. این کتابخانه از Leaflet.js (یه کتابخانه جاوااسکریپت هست) واسه نمایش داده‌های مکانی استفاده میکنه.



```

conftest
datasets (package)
explore
geodataframe
geoseries
io (package)
plotting
sindex
testing
tests (package)
tools (package)
DATA
options = Options(
    display_precision: None [default: Non...e``. Opti...
VERSION
0.14.0
FILE
e:\arcgis pro\bin\python\envs\gisenv\lib\site-packages\geopandas\__init__.py

```

جلوتر به کتابخانه‌های **matplotlib** (برای اینکه چندتا پلات نقشه رو کنار هم ببینی یا لایه‌ها رو بندازی تو یه پلات یا اینکه سائیز راهنمای نقشه رو تغییر بدی و ...) هم نیاز پیدا میکنی. اونا رو هم با دستور import وارد کن.

```
import matplotlib.pyplot as plt
```

علاوه بر اون به یه ماژول از **mpl\_toolkits.axes\_grid1** هم نیاز پیدا میکنی به اسم **make\_axes\_locatable** که اینجا برای کار با راهنمای نقشه ازش استفاده میکنیم.

```
from mpl_toolkits.axes_grid1 import make_axes_locatable
```

اگه بخوای لایه‌ها رو تعاملی کنی یعنی لایه‌های تحت وب ازشون بسازی باید از کتابخانه **Folium** استفاده کنی. با import واردش کن.

```
import folium
```

```

import geopandas as gpd
import matplotlib.pyplot as plt
from mpl_toolkits.axes_grid1 import make_axes_locatable
import folium

```

### گام دوم: خواندن فایلها با **geopandas.read\_file()**

فرض کن یه فایل داری که هم یه سری داده داره و هم ژئومتری مثل GeoJSON, geopackage, یا Shapefile و ... میتونی این فایل رو با دستور زیر بخونی:

**geopandas.read\_file()**

که بلافاصله نوع فایل رو تشخیص میده و یه GeoDataFrame ایجاد میکنه.

تو فایلهای تمرینی یه فولدر به اسم shp براتون گذاشتم که شامل لایه نقطه‌ای ایستگاه‌های مترو تهران (به همراه نام ایستگاه)، لایه خطی خطوط مترو تهران (به همراه نام و شماره خطوط)، لایه پلیگونی محلات تهران (به همراه اسامی محلات و ...) و لایه پلیگونی محدوده طرح ترافیک (خودم حدودی ترسیمش کردم، لایه قابل اعتمادی برای استفاده رسمی نیست) هست. این فولدر رو ببر تو مسیر نصب نرم افزار و تو محیط شخصی که قبلا ساخته بودی کپی کن:

E:\ArcGIS Pro\bin\Python\envs\gisenv

با دستور زیر لایه محلات تهران رو وارد کن و اسمش رو بذار Mahale.



```
Mahale = gpd.read_file(r"E:\ArcGIS Pro\bin\Python\envs\gisenv\Shp\Mahalat.shp")
```

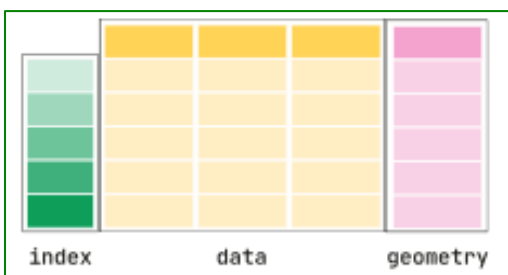
باید آدرس فایل رو به همراه اسم و فرمتش بذاری تو پرانتز `read_file()` و دورش حتما کوتیشن بیاری. آدرس وقتی به صورت \ جدا شده باشه قابل تشخیص نیست و باید دستی همه \ ها رو تبدیل به / کنی. ولی شیوه راحت‌ترش اینه که از ('r') به معنی reverse یا برعکس اول ادرس قبل از کوتیشن استفاده کنی. به این شیوه لایه shp ما که محلات تهران هست خوانده میشه.

```
Mahale = gpd.read_file(r"E:\ArcGIS Pro\bin\Python\envs\gisenv\Shp\Mahalat.shp")
```

Mahale													
ENAME	ID	AREA	...	AREA_CATEG	FLAG	ID_UNIQUE	NAHIE	GEOM_AREA	GEOM_LEN	Shape_Leng	Shape_Le_1	Shape_Area	geometry
Zeynabiyeh	159	1.240729e+06	...	از 100 تا 150 هکتار	0.0	172.0	1304	0.0	0.0	9522.099426	9522.099426	1.240729e+06	POLYGON ((545290.471, 3951036.932, 545314.222, 3...
Shora	136	7.261168e+05	...	از 50 تا 100 هکتار	0.0	173.0	1303	0.0	0.0	3926.180725	3926.180725	7.261168e+05	POLYGON ((545408.551, 3951622.957, 545408.392, 3...
Tehranpars	138	2.410166e+06	...	از 200 تا 300 هکتار	0.0	174.0	0801	0.0	0.0	8041.317136	8041.317136	2.410166e+06	POLYGON ((545426.000, 3952806.364, 545425.162, 3...
Javadiyeh	139	1.770828e+06	...	از 150 تا 200 هکتار	0.0	175.0	0406	0.0	0.0	6857.026087	6857.026087	1.770828e+06	POLYGON ((550285.834, 3955333.336, 550345.492, 3...

حالا اگه اسم متغیری که بهش دادیم و Mahale هست رو جدا بنویسیم و شیفت اینتر بزیم اطلاعات جدولی

لایه رو بهمون میده.



در واقع علاوه بر فیلدهای اطلاعاتی که خود لایه داره، دو تا فیلد جدید هم به لایه اضافه میشه. یکیش Index هست که از ۰ شروع میشه و ابتدای جدول می‌آد و یکی هم geometry که به عنوان آخرین ستون جدول اضافه میشه.

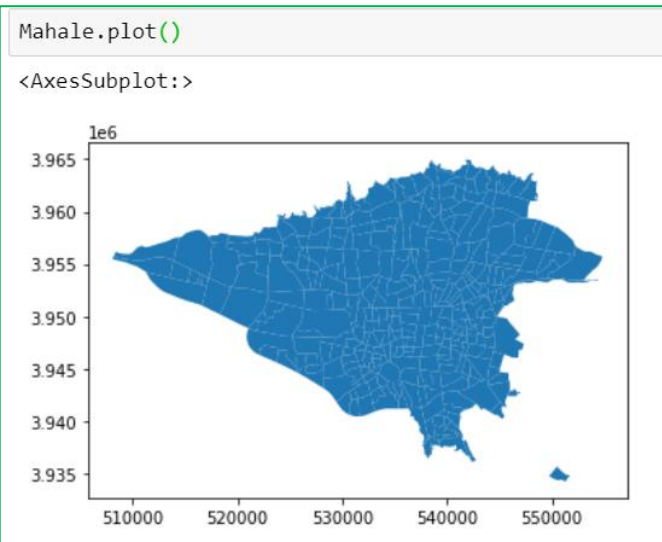
این لایه حالا تبدیل به **GeoDataFrame** شده.

**گام سوم: پلات گرفتن از فایلها** `GeoDataFrame.plot()`

برای پلات گرفتن از لایه‌ها از دستور `GeoDataFrame.plot()` استفاده می‌کنیم. به جای `GeoDataFrame` اسم لایه رو می‌نویسیم.

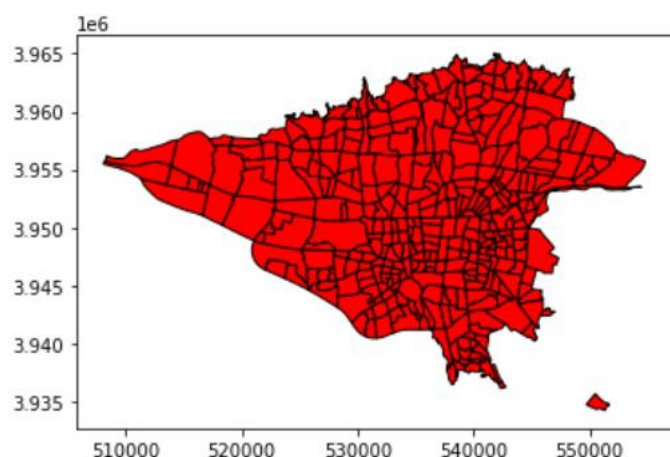
`Mahale.plot()`

اگه پرانتزش رو خالی بذاری به صورت تک رنگ لایه رو بهت نشون میده که خودش به رنگ پیش فرض بهش میده.



```
Mahale.plot(color='red', edgecolor = 'black')
```

<AxesSubplot:>

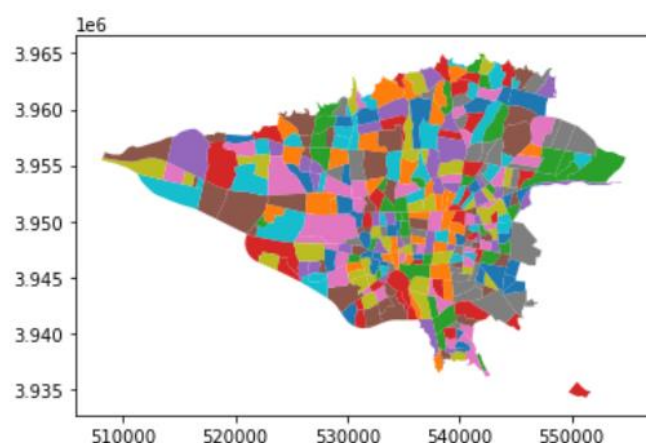


میتونی نقشه رو با " color به رنگی که میخوای  
درباری و همینطور واسه اینکه خط دور بیفته هم یه  
edgecolor=" بهش بدی.

```
Mahale.plot(color='red', edgecolor = 'black')
```

```
Mahale.plot('NAME_MAHAL')
```

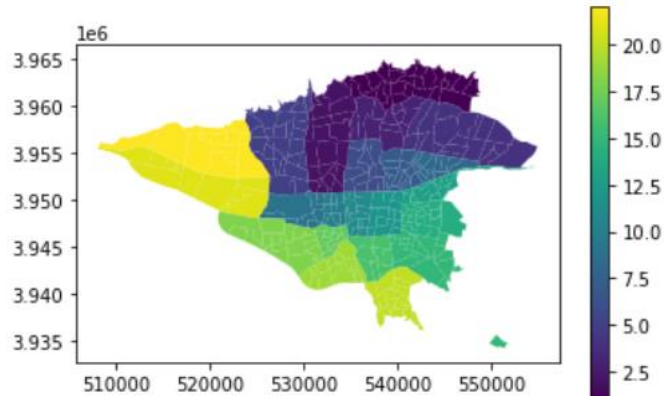
<AxesSubplot:>



اگه تو پرانتزش اسم ستونی که میخوام لایه بر  
اساسش رنگ بگیره رو تو کوتیشن بهش بدی، هر عارضه رو  
جداگونه بر اساس اطلاعات اون فیلد رنگی میکنه.

```
Mahale.plot(column = "REGION", legend = True)
```

<AxesSubplot:>



اگه بخوای پلات راهنما داشته باشه تو پرانتز بعد  
از اسم ستون مورد نظرت یه ویرگول بذار و بنویس  
legend = True

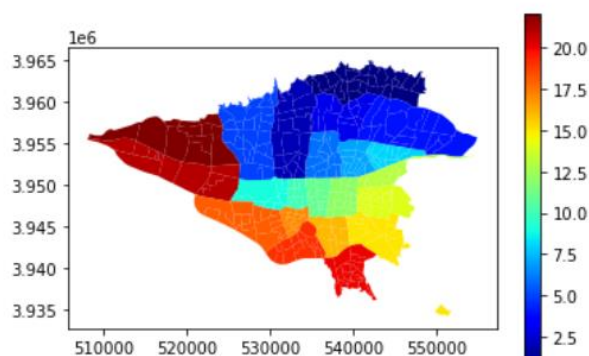
```
Mahale.plot(column = "REGION", legend = True)
```

اگر هم دوست داری رنگ مشخصی رو بهش  
بدی، از " cmap=" استفاده کن و تو کوتیشن اسم  
پلات رنگی رو بهش بده.

```
Mahale.plot(column = "REGION", cmap =  
'hsv', legend = True)
```

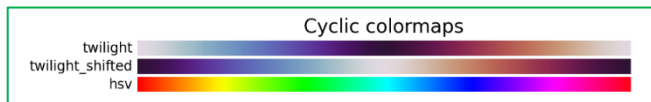
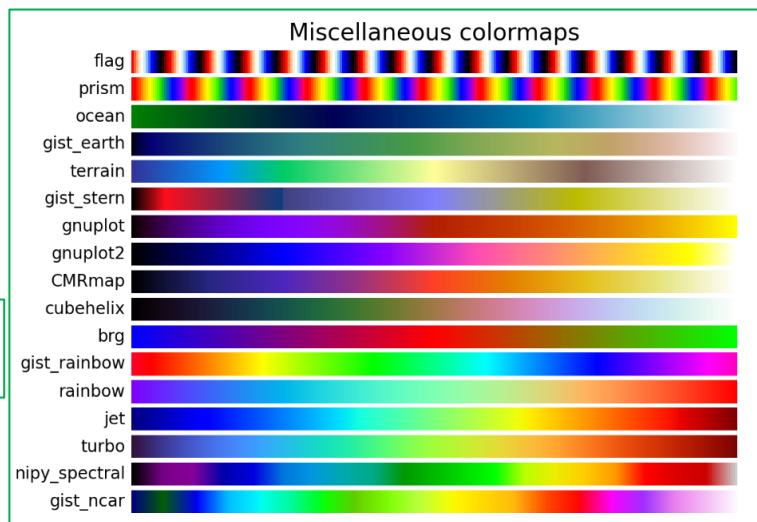
```
Mahale.plot(column = "REGION", cmap = 'jet', legend = True)
```

<AxesSubplot:>



تو سایت زیر میتونی اسم انواع پلتهای رنگی  
رو ببینی و به cmap بدی:

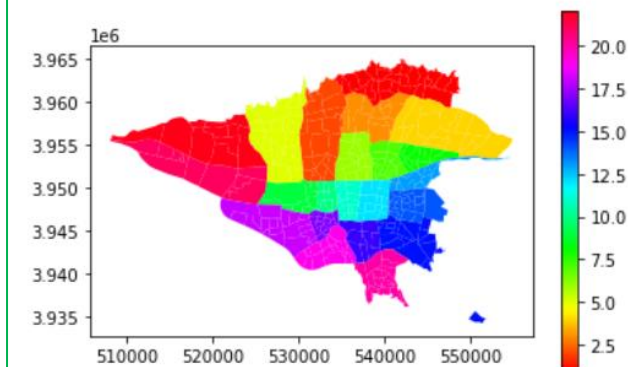
[matplotlib.org](http://matplotlib.org)



برای مثال رنگ hsv رو بهش میدیم که  
تغییر رنگ پیدا کنه.

```
Mahale.plot(column = "REGION", cmap = 'hsv', legend = True)
```

<AxesSubplot:>



استفاده از legend\_kwds

همونطور که تو نقشه بالا میبینی راهنمای نقشه زیاد گویا نیست. واسه تغییر دادنش از legend\_kwds که مخفف keywords هست استفاده میکنیم و اطلاعاتش رو به صورت دیکشنری یا {key:value} بهش میدیم.

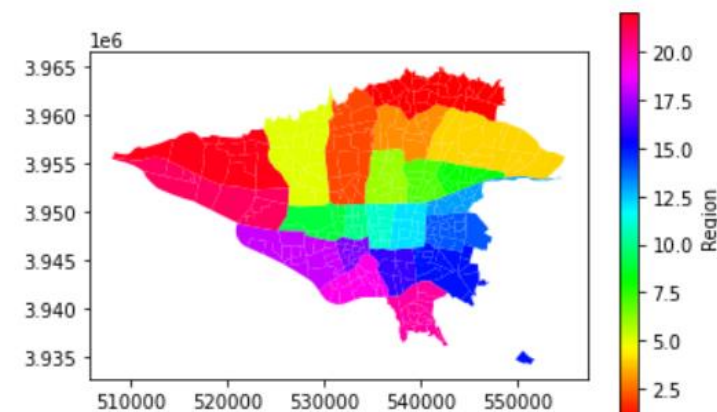
```
legend_kwds = {'label': "Region"}
```

برای برچسب دادن یا labeling از دستور زیر استفاده کن:

```
Mahale.plot(column = "REGION", cmap = 'hsv', legend = True, legend_kwds = {'label': "Region"})
```

```
Mahale.plot(column = "REGION", cmap = 'hsv', legend = True, legend_kwds = {'label': "Region"})
```

<AxesSubplot:>



برای اینکه راهنما رو بیاریم تو کادر نقشه و سائزش رو تغییر بدیم باید از کتابخونه **matplotlib** استفاده کنیم که قبلاً با دستور import واردش کردی.

```
import matplotlib.pyplot as plt
```

حالا باید براش فیگور به اسم fig و محور به اسم ax درست کنی. اینکار با **plt.subplot()** انجام میشه. تو پرانتزش **figsize()** هم میدیم که اندازه شکل رو برامون مشخص میکنه.

```
fig, ax = plt.subplot(figsize = (۷, ۷))
```

واسه اینکه بتونیم راهنما رو تغییر بدیم باید از ماژول **make\_axes\_locatable** استفاده کنیم که قبلاً با دستور زیر آوردیمش.

```
from mpl_toolkits.axes_grid1 import make_axes_locatable
```

حالا به متغیر به نام driver تعریف کن و معادل عبارت زیر قرارش بده و تو پرانتزش بگو که روی همون محور ax که قبلاً مشخص کردم بیفته.

```
driver = make_axes_locatable(ax)
```

یه متغیر دیگه به اسم cax تعریف کن (c مخفف color هست که رنگ نمودار رو مشخص میکنه) و با دستور **append\_axes()** متغیر قبلی رو بهش وصل کن و تو پرانتزش مشخص کن که راهنما کجای محور قرار بگیره. میتونی سمت راست یا 'right' بذاری و size رو هم به درصد بهش بدی. از آرگومان pad هم استفاده کن که بین نوار راهنما و نقشه فاصله ایجاد کنی.

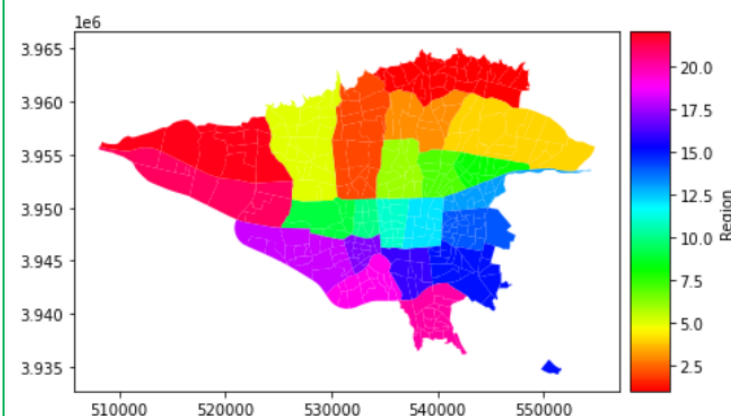
```
cax = driver.append_axes('right', size = '۷%', pad = ۰.۱)
```

مجدد ازش پلات بگیر. تو پرانتزش باید ax = ax رو هم براش تعریف کنی و cax که معادل رنگ محور هست رو هم بهش بدی.

```
Mahale.plot(column = "REGION", cmap = 'hsv', legend = True, legend_kwds = {'label':"Region"}, ax = ax, cax = cax)
```

```
fig, ax = plt.subplots(figsize = (7,7))
driver = make_axes_locatable(ax)
cax = driver.append_axes('right', size = '7%', pad= 0.1 )
Mahale.plot(column = "REGION", cmap = 'hsv' , legend = True, legend_kwds = {'label':"Region"}, ax = ax, cax = cax)
```

<AxesSubplot:>



## گام چهارم: اضافه کردن لایه‌های دیگه به پلات قبلی

میخواهم لایه خطی مترو رو بیارم و روی این نقشه بندازم. به همون شیوه قبلی لایه رو وارد کن.

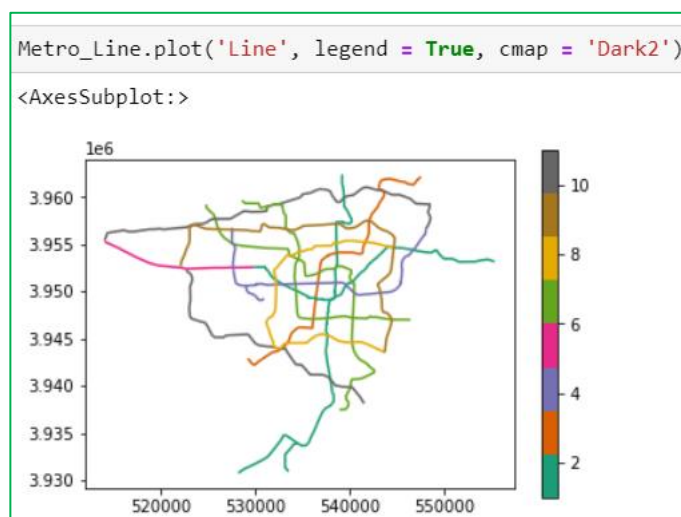
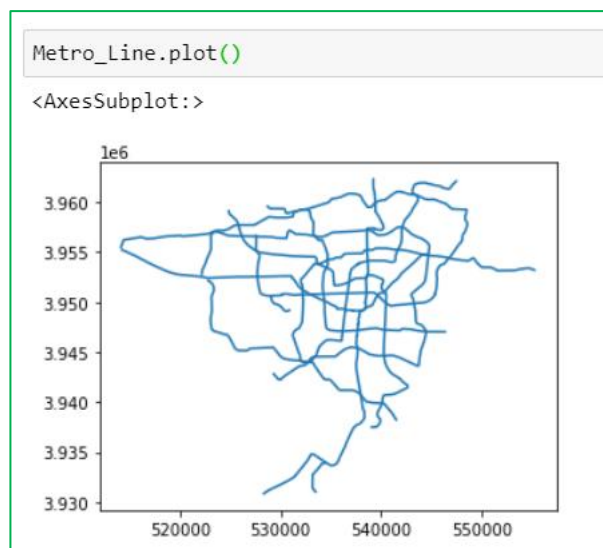
`Metro_Line = gpd.read_file(r"E:\ArcGIS Pro\bin\Python\envs\gisenv\Shp\Metro_Line.shp")`

```
Metro_Line = gpd.read_file(r"E:\ArcGIS Pro\bin\Python\envs\gisenv\Shp\Metro_Line.shp")
Metro_Line
```

	OBJECTID	ENTITY	LAYER	ELEVATION	THICKNESS	COLOR	Name	Line	Shape_Leng	geometry
0	1	Polyline	M_ROUTE5	0.0	0.0	53	میدان صادقیه (تهران)-گلشهر-خط 5	5	16377.091186	LINESTRING Z (514067.554 3955287.359 0.000, 51...
1	2	Polyline	M_ROUTE1	0.0	0.0	230	تجریش-کهریزک- خط 1	1	45446.708873	MULTILINESTRING Z ((533549.879 3930947.051 0.0...
2	3	Polyline	M_ROUTE3	0.0	0.0	183	بزرگراه شهید آیت الله سعیدی-شهرک قائم-خط 3	3	34225.061933	LINESTRING Z (547540.669 3962146.722 0.000, 54...
3	4	Polyline	M_ROUTE2	0.0	0.0	32	میدان صادقیه(تهران)-فرهنگسرای اشراق-خط 2	2	30099.902652	LINESTRING Z (529865.207 3952595.970 0.000, 53...
4	5	Polyline	M_ROUTE6	0.0	0.0	124	شهرک دولت آزادسولقان (شهرک شرکت نفت-خط 6)	6	38987.964880	LINESTRING Z (524837.609 3959243.814 0.000, 52...
5	6	NaN	NaN	0.0	0.0	0	خط 11	11	28006.792460	LINESTRING Z (522690.320 3952229.936 0.000, 52...
6	7	NaN	NaN	0.0	0.0	0	خط 10	10	40776.678908	LINESTRING Z (548116.505 3956632.985 0.000, 54...
7	8	NaN	NaN	0.0	0.0	0	شهرک امیرالمومنین -فرودگاه مهرآباد- خط 7	7	29509.030651	LINESTRING Z (528611.468 3959658.395 0.000, 52...
8	9	NaN	NaN	0.0	0.0	0	شهید کلاهدوز-ایکباتان-خط 4	4	26590.803014	LINESTRING Z (548116.505 3956632.985 0.000, 54...
9	10	NaN	NaN	0.0	0.0	0	شهید کلاهدوز-ایکباتان-خط 4	4	7500.878624	MULTILINESTRING Z ((527709.722 3952185.565 0.0...
10	11	Polyline	M_ROUTE8	0.0	0.0	96	خط 8	8	35476.081738	LINESTRING Z (544861.390 3954698.285 0.000, 54...
11	12	NaN	NaN	0.0	0.0	0	خط 9	9	41383.235250	LINESTRING Z (543747.005 3943566.810 0.000, 54...

حالا ازش پلات بگیر. یکبار بدون رنگ یعنی تو پرانتزش هیچی ننویس. یکبار دیگه هم تو پرانتز بهش یه ستون برای رنگ دادن معرفی کن، راهنما بهش اختصاص بده و کد رنگ بده که بر اساس اطلاعات اون ستون رنگ بگیره.

`Metro_Line.plot('Line', legend = True, cmap = 'Dark2')`



## دیدن دو تا پلات کنار هم

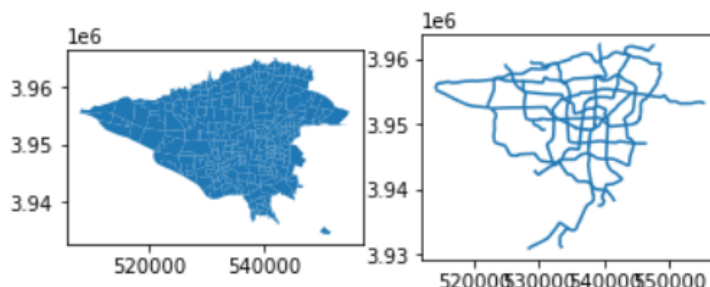
الان یه پلات برای لایه محلات داری و یه پلات هم برای لایه خطوط مترو. میخوایم این دو تا پلات رو مجاور هم یا به صورت پایین و بالای هم ببینیم. برای اینکار باز هم باید از کتابخانه matplotlib استفاده کنی که بتونی فیگور و محور بسازی. حالا باید fig و ax رو براش مشخص کنی و ازشون subplot بسازی. چون میخوایم کنار هم باشن باید **ncols** رو بدیم ۲ که تو دو تا ستون کنار هم بیاره.



```
fig, (ax۱, ax۲) = plt.subplots(ncols=۲)
Mahale.plot(ax = ax۱)
Metro_Line.plot(ax = ax۲)
```

```
fig, (ax1, ax2) = plt.subplots(ncols=2)
Mahale.plot(ax = ax1)
Metro_Line.plot(ax = ax2)
```

<AxesSubplot:>

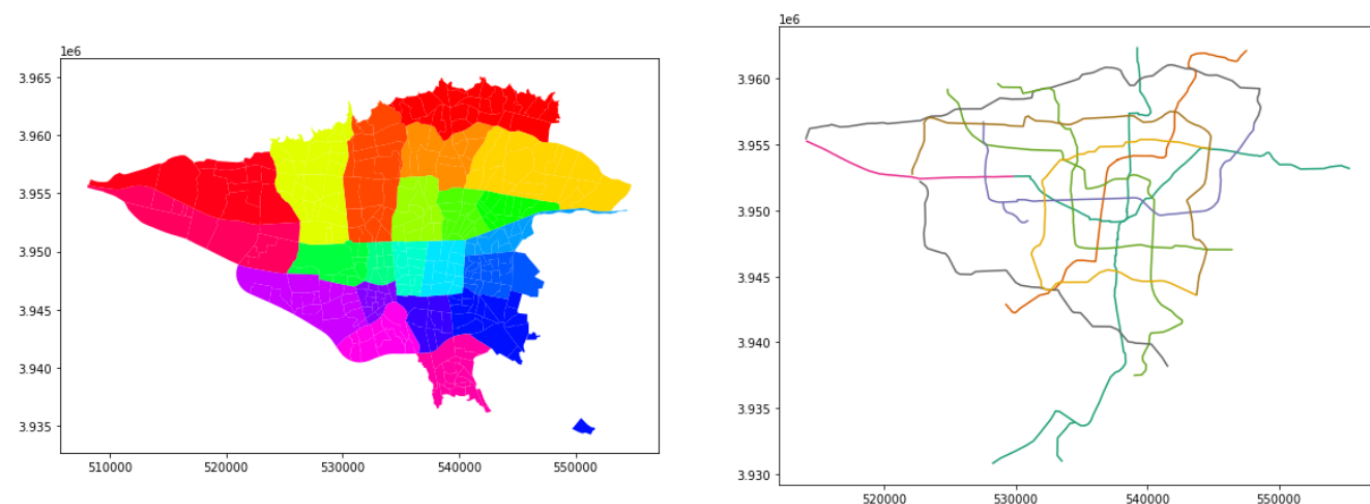


اگه بخوای رنگهای قبلی رو هم داشته باشن اطلاعات قبلی که به هر لایه داده بودی رو برای هر کدوم کپی کن. فقط اینجا دیگه باید کنار اسم ستونها عبارت " " column رو هم بیاری. اندازه نمودارها کوچک میشه، با figsize(,) اندازه‌اش رو تغییر بده.

```
fig, (ax۱, ax۲) = plt.subplots(ncols=۲, figsize = (۲۰, ۱۰))
Mahale.plot(ax = ax۱, column = "REGION", cmap = 'hsv')
Metro_Line.plot(ax = ax۲, column = 'Line', cmap = 'Dark۲')
```

```
fig, (ax1, ax2) = plt.subplots(ncols=2, figsize = (20, 10))
Mahale.plot(ax = ax1, column = "REGION", cmap = 'hsv')
Metro_Line.plot(ax = ax2, column = 'Line', cmap = 'Dark2')
```

<AxesSubplot:>

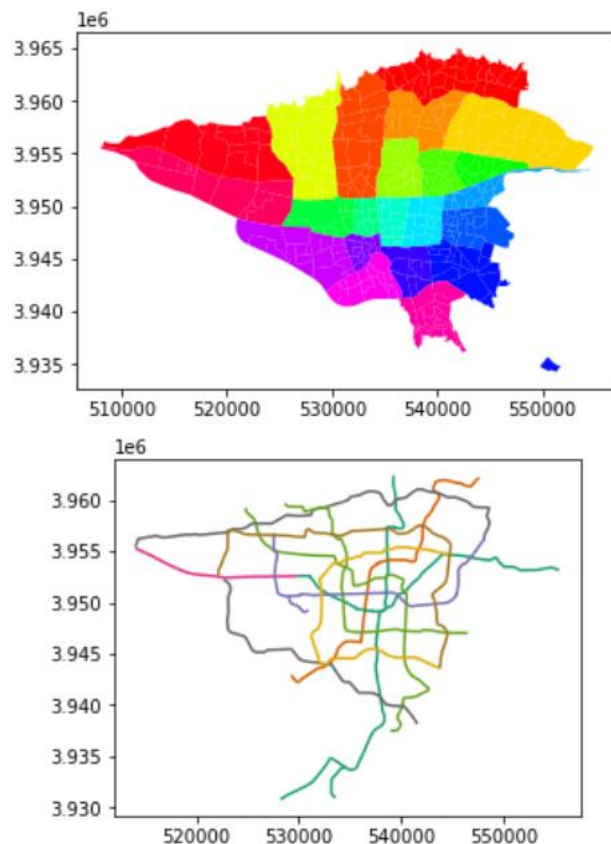


اگه تو کد بالا به جای ncols عبارت nrows=۲ بذاریم، پلاتها زیر هم نشون داده میشن.

```
fig, (ax۱, ax۲) = plt.subplots(nrows=۲, figsize = (۱۰, ۸))
Mahale.plot(ax = ax۱, column = "REGION", cmap = 'hsv')
Metro_Line.plot(ax = ax۲, column = 'Line', cmap = 'Dark۲')
```

```
fig, (ax1, ax2) = plt.subplots(nrows=2, figsize = (10, 8))
Mahale.plot(ax = ax1, column = "REGION", cmap = 'hsv')
Metro_Line.plot(ax = ax2, column = 'Line', cmap = 'Dark2')
```

<AxesSubplot:>



### دیدن دو یا چند تا لایه در یک پلات

همون کد بالا رو کپی کن. اینجا نیازی نیست که دو تا ax تعریف کنی. تعداد ستون و سطر رو هم نیازی نیست تو subplot() بهش بدی. فقط اندازه شکل رو مشخص کن. هر دو رو با ax برابر قرار بده. به این شکل لایه‌ها روی هم می‌افتن.

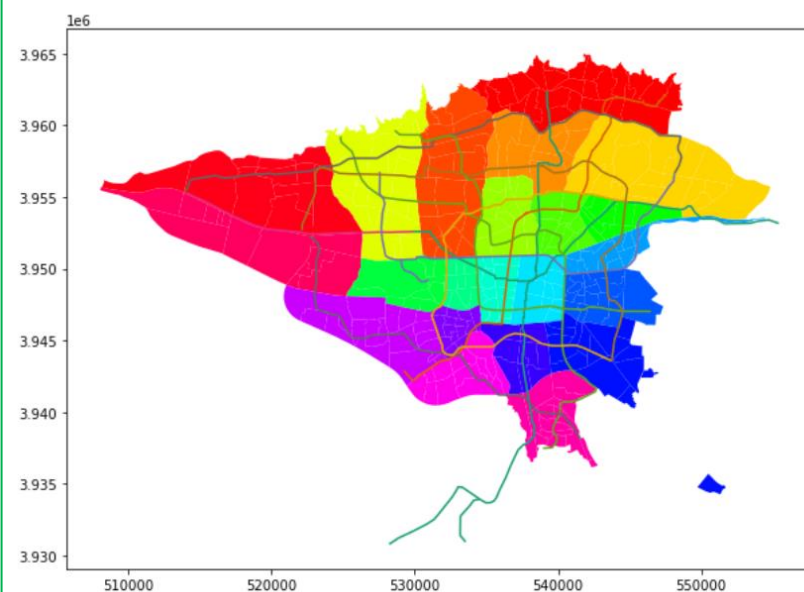
```
fig, ax = plt.subplots(figsize = (10, 8))
```

```
Mahale.plot(ax = ax, column = "REGION", cmap = 'hsv')
```

```
Metro_Line.plot(ax = ax, column = 'Line', cmap = 'Dark2')
```

```
fig, ax = plt.subplots(figsize = (10, 8))
Mahale.plot(ax = ax, column = "REGION", cmap = 'hsv')
Metro_Line.plot(ax = ax, column = 'Line', cmap = 'Dark2')
```

<AxesSubplot:>





اگه لایه‌ها پليگونی بودن با اینکار لایه دوم روی لایه اول می‌افتاد و لایه زیر دیده نمی‌شد. واسه رفعش میتونستی رنگ یکی رو None بدی و فقط رنگ دور داشته باشه که وقتی روی هم می‌افتن اطلاعات لایه زیری محو نشه.

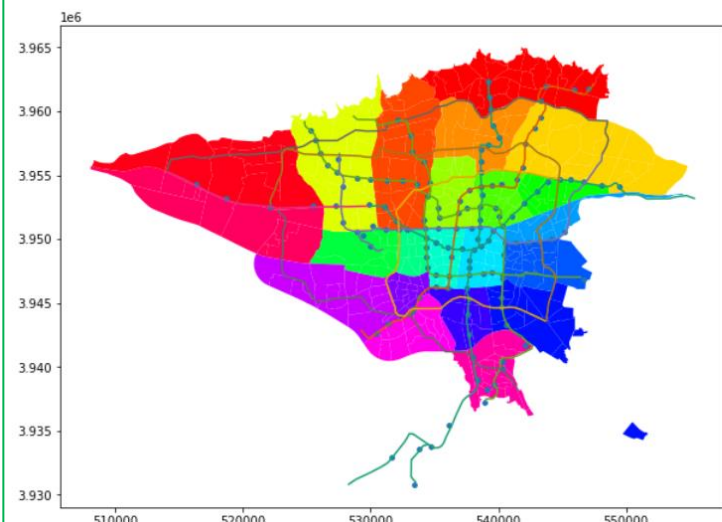
میتونی به همین شیوه لایه نقطه‌ای ایستگاه‌ها رو هم بیاری و روی لایه‌ها بندازی.

`Metro_St = gpd.read_file(r"E:\ArcGIS Pro\bin\Python\envs\gisenv\Shp\Metro_Station.shp")`

```
Metro_St = gpd.read_file(r"E:\ArcGIS Pro\bin\Python\envs\gisenv\Shp\Metro_Station.shp")
```

```
fig, ax = plt.subplots(figsize = (10, 8))
Mahale.plot(ax = ax, column = "REGION", cmap = 'hsv')
Metro_Line.plot(ax = ax, column = 'Line', cmap = 'Dark2')
Metro_St.plot(ax = ax, markersize = 14)
```

<AxesSubplot:>



میتونی رنگ و اندازه نقطه‌ها رو هم

عوض کنی. رنگ رو با همون `color="` و اندازه

با `markersize=` بهش بده.

```
fig, ax = plt.subplots(figsize = (10, 8))
Mahale.plot(ax = ax, column = "REGION", cmap = 'hsv')
Metro_Line.plot(ax = ax, column = 'Line', cmap = 'Dark2')
Metro_St.plot(ax = ax, markersize = 14)
```

### گام پنجم: سیستم تصویر یا Projection

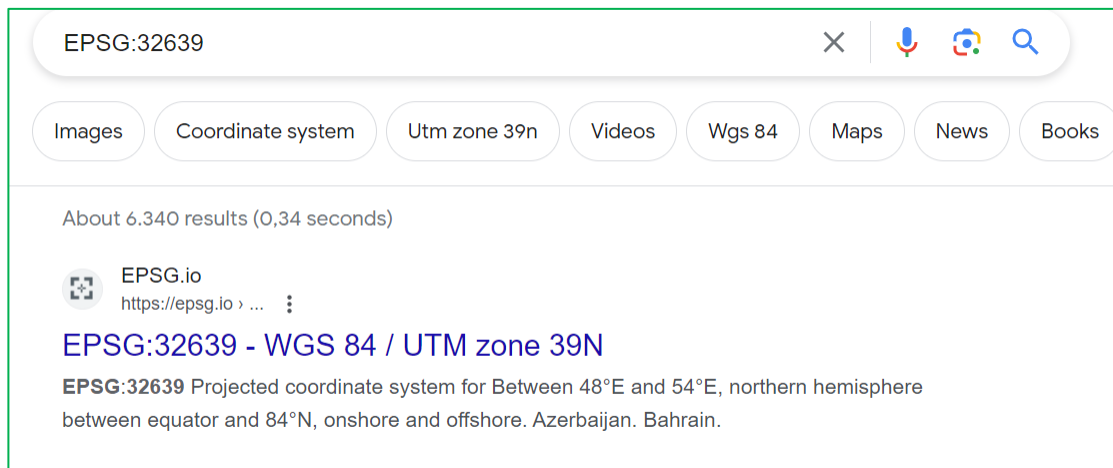
هر ژتومتري سیستم مختصات یا CRS داره که از طریق `GeoSeries.crs` میشه بهش دسترسی داشت. CRS به Geopandas میگه که سیستم مختصات هر ژتومتري کجای سطح کره زمین هست. گاهی CRS جغرافیایی هست یعنی طول و عرض داره. در این حالت CRS میشه WGS84 که با کد ۴۳۲۶ WSGC نمایش داده میشه. بیا سیستم مختصات داده‌مون رو چک کنیم.

`Metro_St.crs`

Metro\_St.crs

```
<Projected CRS: EPSG:32639>
Name: WGS 84 / UTM zone 39N
Axis Info [cartesian]:
- E[east]: Easting (metre)
- N[north]: Northing (metre)
Area of Use:
- name: Between 48°E and 54°E, northern hemisphere between equator and 84°N, onshore and offshore. Azerbaijan. Bahrain. Islamic Republic of Iran. Iraq. Kazakhstan. Kuwait. Oman. Qatar. Russian Federation. Saudi Arabia. Somalia. Turkmenistan. United Arab Emirates. Yemen.
- bounds: (48.0, 0.0, 54.0, 84.0)
Coordinate Operation:
- name: UTM zone 39N
- method: Transverse Mercator
Datum: World Geodetic System 1984 ensemble
- Ellipsoid: WGS 84
- Prime Meridian: Greenwich
```

اینجا بهمون EPSG رو معادل ۳۲۶۳۹ داده تو گوگل سرچش کن ببین چی بهت میده.



برای تبدیل سیستمهای مختصات به هم همیشه از دستور **to\_crs()** استفاده کرد. تو پرانتزش باید = espg عددی که مربوط به اون محدوده همیشه رو بهش بدی. سایت زیر بهت میگوید که سیستم مختصات محدودهات چی هست:

[EPSG.io: Coordinate Systems Worldwide](https://epsg.io)

علاوه بر اون تو پرانتزش باید inplace = True قرار بدی که تغییرش رو دائمی کنه.

لایه‌های ما سیستم مختصات دارن و نیازی به تغییرشون نداریم.



### گام ششم: کاربردهای Geopandas در پردازش داده‌های جغرافیایی

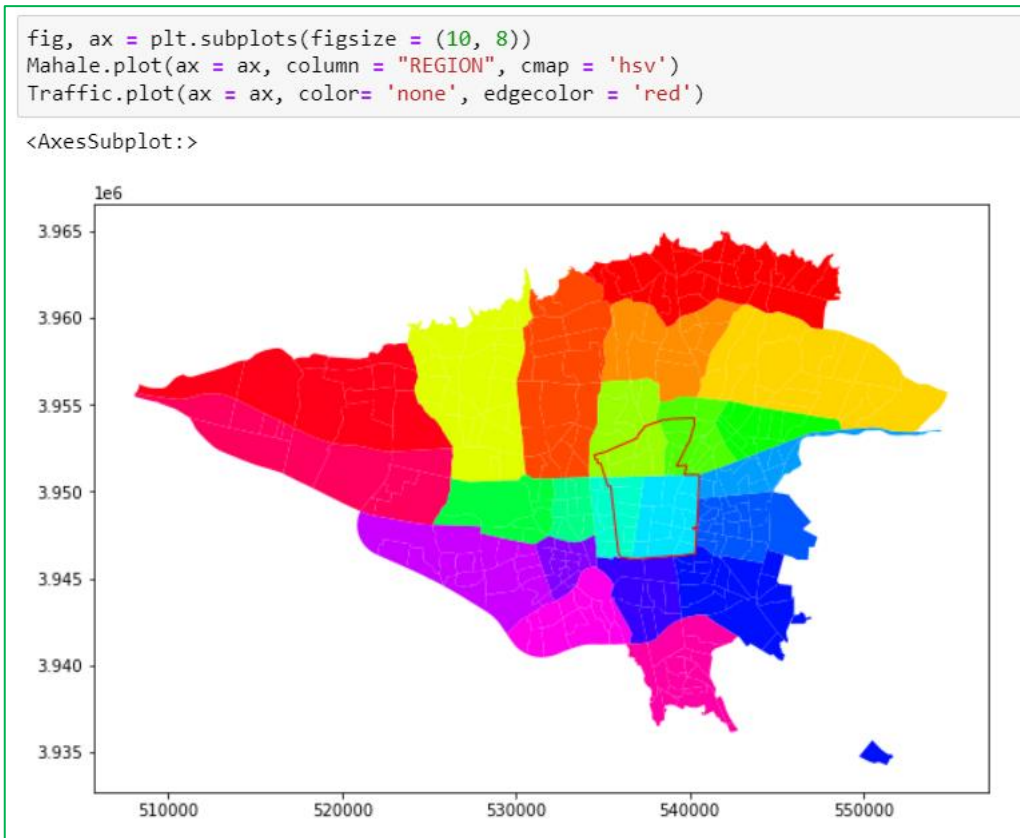
تو این بخش می‌خوایم به سری کارهای تحلیلی روی لایه‌ها انجام بدیم. فرض کن که می‌خوایم ببینیم که کدوم محلات تو طرح ترافیک زوج و فرد قرار می‌گیرن. پس به دو تا لایه نیاز داریم یکی لایه محلات و یکی هم لایه طرح ترافیک هست.

لایه Traffic رو وارد کن و ازش خروجی بگیر که جدولش رو ببینی. (لایه طرح ترافیک رو خودم حدودی ترسیم کردم، قابل اعتماد نیست).

```
Traffic = gpd.read_file(r"E:\ArcGIS Pro\bin\Python\envs\gisenv\Shp\Traffic.shp")
```

لایه ترافیک و محلات رو تو یه فیگور بنداز و پلات بگیر. رنگ داخل لایه ترافیکی رو بردار که اطلاعات لایه زیری محو نشه.

```
fig, ax = plt.subplots(figsize = (10, 8))
Mahale.plot(ax = ax, column = "REGION", cmap = 'hsv')
Traffic.plot(ax = ax, color = 'none', edgecolor = 'red')
```



یه سری از تحلیلها روی داده‌های جغرافیایی با دستور زیر انجام میشن:

**`geopandas.overlay(df۱, df۲, how='intersection', keep_geom_type=None, make_valid=True)`**

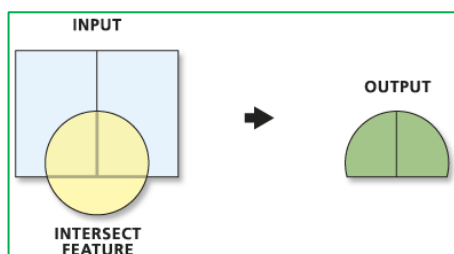
df<sup>۱</sup> یعنی دیتافریم یا لایه اول. df<sup>۲</sup> یعنی لایه دوم.

بخش how معرف روشی هست که میخوای لایه‌ها رو روی هم بندازی. پیش فرضش روی **intersection** هست ولی میتونی به جاش این موارد رو بذاری.

**‘union’, ‘identity’, ‘symmetric\_difference’ or ‘difference’**

بخش keep\_geom\_type مربوط به بخش ژئومتری میشه که می‌گه کدوم لایه رو به عنوان شکل هندسی نگه داریم. معمولاً لایه دوم رو باید بهش بدیم.

بخش make\_valid هم اگه روی True باشه و ژئومتری لایه ایراد داشته باشه تصحیحش میکنه ولی اگه روی False بذاری و ایرادی تو ژئومتری لایه باشه بهت خطا میده.



### Intersections

برای intersection به دو تا لایه نیاز داریم که بخشی از این دو تا لایه روی هم باشن. میخوایم این بخش مشترک رو جدا کنیم.

برای مثال دو تا لایه بالا با دستور intersection به شکل زیر خروجی

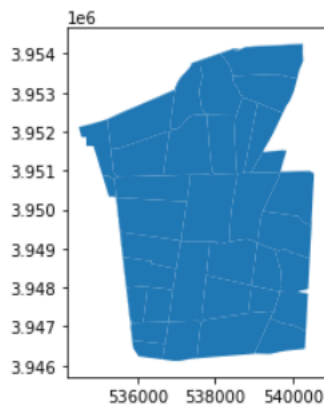
میدن:

`T_M_intersection = gpd.overlay(Traffic, Mahale, how = 'intersection')`

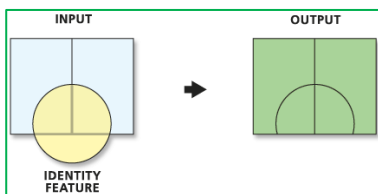
`T_M_intersection.plot ()`

```
T_M_intersection = gpd.overlay(Traffic, Mahale, how = 'intersection')
T_M_intersection.plot()
```

<AxesSubplot:>

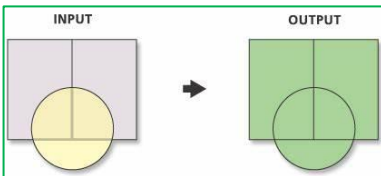


### Identity



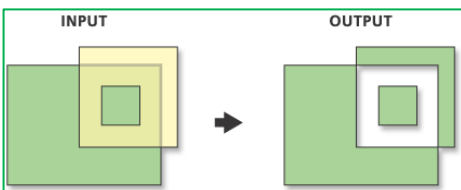
همونطور که تو شکل میبینی با این دستور لایه اصلی به همراه بخشی از لایه دوم که با هم تقاطع دارن خروجی داده میشه.

### Union



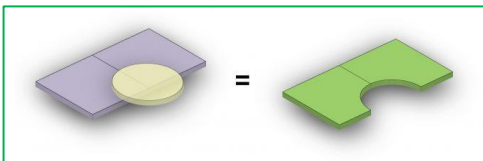
با Union اطلاعات هر دو تا لایه با هم جمع میشه. در واقع برعکس intersection هست.

### Symmetric Differences



با این دستور شبیه شکل مقابل، بخشی از دو تا لایه که با هم تداخل داره جدا میشه.

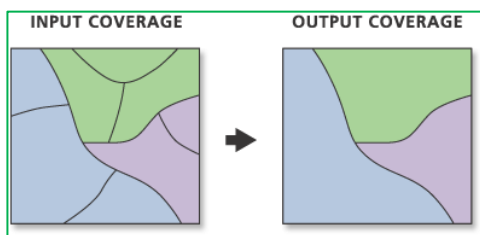
### Difference



این دستور محدوده اول رو از محدوده دوم کم میکنه و نتیجه رو میده بیرون.

با **Geopandas** میشه یه سری تحلیل دیگه هم روی لایه انجام داد. برای مثال:

### Dissolve

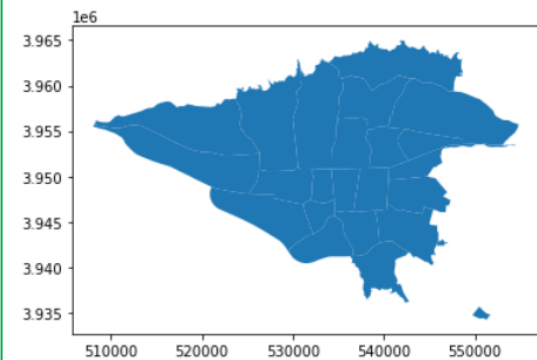


معمولا این دستور به همراه دستور Union همراه هست. علاوه بر اون میشه مثلا برای بدست آوردن مثلا لایه بلوکهای شهری با حل کردن خطوط قطعات داخل بلوک ازش استفاده کرد.

برای اینکار از دستور **GeoDataFrame.dissolve()** استفاده می‌کنیم. اگر فیلد مشخصی رو بهش معرفی نکنیم خودش روی ژئومتری فعال اعمال میشه. اگر فیلد مشخص تو ذهنت هست تو پرانتزش بنویس.

```
M_Dissolve = Mahale.dissolve('REGION')
M_Dissolve.plot()
```

<AxesSubplot:>



فرض کن میخوایم لایه محلات رو با کمک فیلد مربوط به شماره منطقه حل کنیم و ازش لایه مناطق شهری تهران رو استخراج کنیم. کد زیر رو براش بنویس.

```
M_Dissolve = Mahale.dissolve('REGION')
M_Dissolve.plot()
```

## Buffer

واسه ترسیم Buffer یا حریم از دستور زیر استفاده کن:

**GeoDataFrame.buffer()**

بافر روی ژئومتری فعال اعمال میشه ولی میشه با دستور GeoSeries هر ستونی که میخوایم رو بهش بدیم که بر اساس اون برامون حریم بزنه.

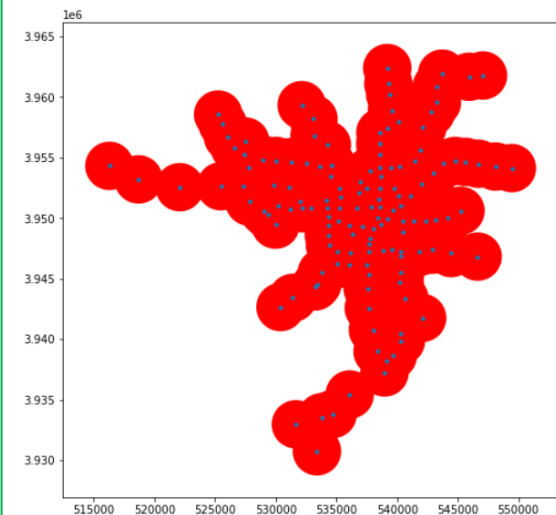
فرض کن میخوایم برای لایه ایستگاه‌های مترو یه حریم ۲۰۰۰

متری ترسیم کنیم.

```
Metro_Buffer = Metro_St.buffer(۲۰۰۰)
Metro_Buffer.plot()
```

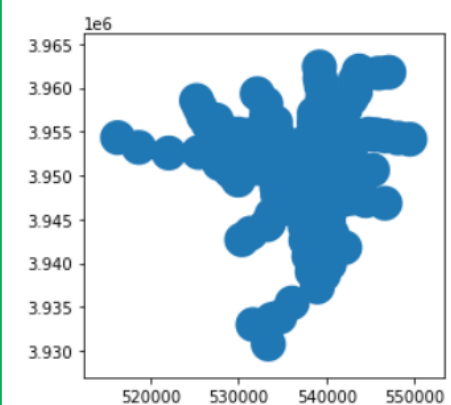
```
fig, ax = plt.subplots(figsize = (10, 8))
Metro_Buffer.plot(ax = ax, color= 'red')
```

<AxesSubplot:>



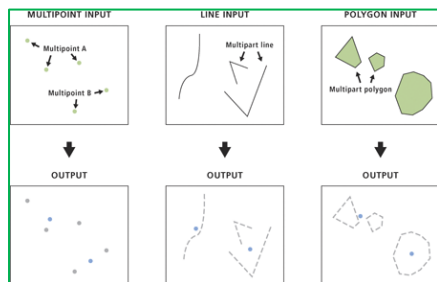
```
Metro_Buffer = Metro_St.buffer(2000)
Metro_Buffer.plot()
```

<AxesSubplot:>



اگر بخوایم لایه ایستگاه‌ها و حریمی که ترسیم کردیم با هم روی یه کادر بیفتن باید از کد زیر استفاده کنیم.

```
fig, ax = plt.subplots(figsize = (۱۰, ۸))
Metro_Buffer.plot(ax = ax, color= 'red')
Metro_St.plot(ax = ax, markersize = ۱۰)
```

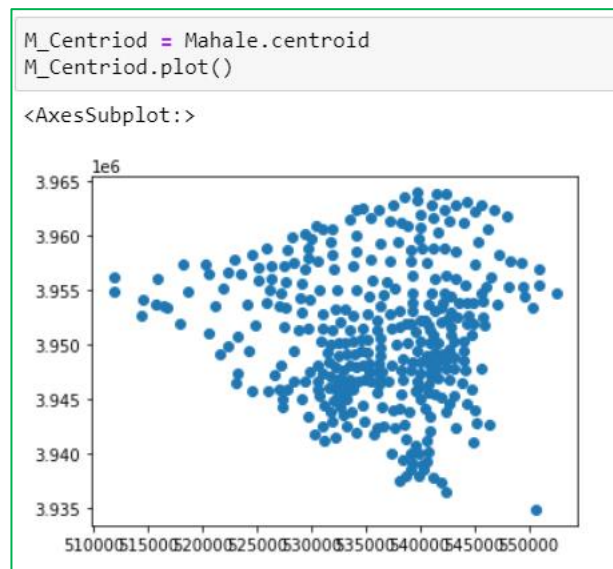


واسه اینکه نقطه مرکزی یه لایه پلیگونی یا خطی رو بدست بیاریم ازش

استفاده میشه. دستورش هم میشه **GeoDataFrame.centriod**

فرض کن میخوایم نقطه مرکزی هر محله رو دربیاریم. با دستور زیر باید کدش رو بنویسی:

```
M_Centriod = Mahale.centroid
M_Centriod.plot()
```



### گام هفتم: محاسبه مساحت

واسه محاسبه مساحت هر پلیگون یا چندین پلیگون باید از ویژگی **GeoDataFrame.area** استفاده کنی. این ویژگی یه سری پاندا یا **pandas.series** رو برمیگردونه. حواست باشه که **GeoDataFramme.area** فقط **Geoseries.area** هست که واسه ستون ژئومتری فعال اعمال میشه. اگه بخوای مساحت رو به هکتار بنویسی کافیه تو همون پرانتز به ۱۰۰۰۰ تقسیمش کنی.

میخوایم برای لایه‌ای از تحلیل بالا بدست آوردیم و نشون دهنده محلات موجود در محدوده طرح ترافیک هست یه فیلد مساحت به هکتار درست کنیم. کدش رو به صورت زیر می‌نویسیم.

```
T_M_intersection['area'] = T_M_intersection.area / 10000
T_M_intersection.head()
```

TID_2	REGION	NAME_MAHAL	...	FLAG	ID_UNIQUE	NAHIE	GEOM_AREA	GEOM_LEN	Shape_Leng_2	Shape_Le_1	Shape_Area_2	geometry	area
14	11.0	الغلاب	...	1.0	189.0	1101	0.0	0.0	4963.006669	4963.006669	1.151831e+06	POLYGON ((535459.014 3950308.206, 535447.042 3...	115.183124
16	11.0	البارفت	...	0.0	191.0	1104	0.0	0.0	3544.816855	3544.816855	6.643038e+05	POLYGON ((535886.188 3946453.750, 535863.875 3...	0.591389
17	12.0	سدگلج	...	1.0	192.0	1203	0.0	0.0	5854.288534	5854.288534	1.640351e+06	POLYGON ((537867.816 3949169.371, 537864.196 3...	164.035091
21	6.0	کشاورز	...	1.0	221.0	0602	0.0	0.0	5408.447037	5408.447037	1.493583e+06	POLYGON ((536880.867 3952029.595, 536851.531 3...	149.358280
22	6.0	وصال شیرازی	...	0.0	222.0	0602	0.0	0.0	4702.412926	4702.412926	1.243898e+06	POLYGON ((536817.958 3951959.409, 536817.561 3...	124.389751

### گام هشتم: محدود کردن فیلدهای اطلاعاتی

واسه اینکه فقط چند تا فیلد رو نگه داری از دستور `geoDataFrame[",","]` استفاده میکنیم. از فیلدهای بالا میخوایم نام محلات، شماره منطقه و مساحت رو نگه داریم و بقیه رو دور بریزیم. حواست باشه که فیلد Geometry رو پاک نکنی. از دستور زیر استفاده کن.

```
filds_to_keep = ['area', 'NAME_MAHAL', 'REGION', 'geometry']
T_M_I_F=T_M_intersection[filds_to_keep ]
T_M_I_F.head()
```

```
filds_to_keep = ['area', 'NAME_MAHAL', 'REGION', 'geometry']
T_M_I_F=T_M_intersection[filds_to_keep ]
T_M_I_F.head()
```

	area	NAME_MAHAL	REGION	geometry
0	115.183124	انقلاب	11.0	POLYGON ((535459.014 3950308.206, 535447.042 3...
1	0.591389	ایبارفت	11.0	POLYGON ((535886.188 3946453.750, 535863.875 3...
2	164.035091	سنگلج	12.0	POLYGON ((537867.816 3949169.371, 537864.196 3...
3	149.358280	کشاورز	6.0	POLYGON ((536880.867 3952029.595, 536851.531 3...
4	124.389751	وصال شیرازی	6.0	POLYGON ((536817.958 3951959.409, 536817.561 3...

### گام نهم: خروجی گرفتن از لایه `.to_file()`

میخوایم از لایه جدید یا T\_M\_I\_F به خروجی shp بگیریم. از دستور `to_file` استفاده میکنیم. یادت باشه تو پراپرتی ته اسم لایه پسوند shp. رو بذاری. `driver= "Esri shapefile"` رو هم توی پراپرتی می نویسیم.  
`T_M.to_file("Traffic_M.shp", driver="ESRI Shapefile")`

```
T_M_I_F.to_file("Traffic_M.shp", driver="ESRI Shapefile")
```

لایه اینجا ذخیره میشه.

E:\ArcGIS Pro\bin\Python\envs\gisenv

میتونی از اینجا برش داری و تو فولدر لایه های تمرینیت بذاری.

### گام دهم: وبی کردن لایه با `explore()`

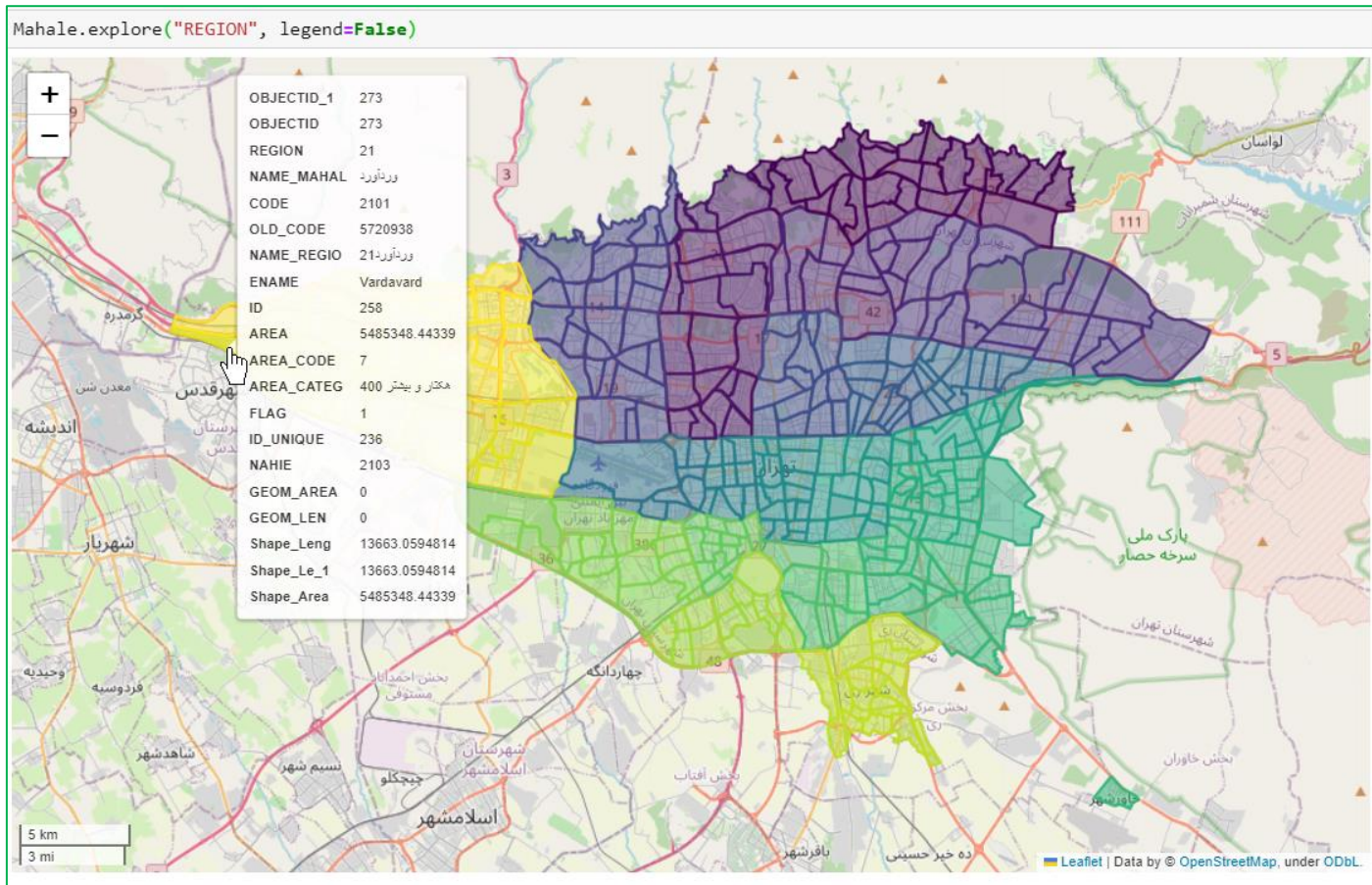
واسه تعاملی کردن نقشه ها یا به نوعی web ساختن ازش که دیگران هم بتونن ببینن نیاز به یه کتابخونه دیگه به اسم folium داری که قبلا واردش کردی. میتونی از دستور زیر واسه ساخت لایه تعاملی استفاده کنی:

**GeoDataFrame.explore()**

این دستور شبیه دستور `plot()` رفتار میکنه ولی نقشه رو به حالت وبی بهت میده. لایه محلات رو با استفاده از ستون REGION تعاملی کن.



Mahale.explore("REGION", legend=False)

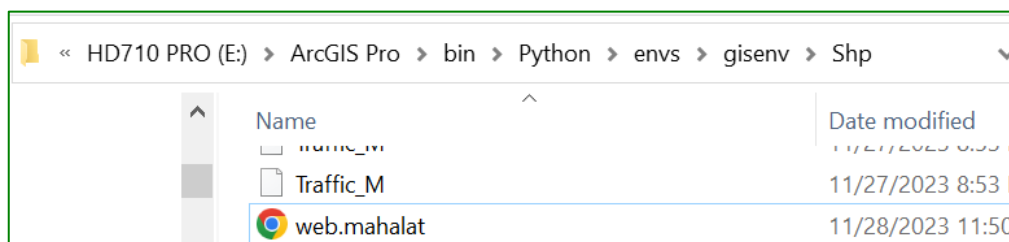


اگه بخوای میتونی این لایه رو تو مسیر فایلهای تمرینی به صورت فایل HTML ذخیره کنی که بتونی در اختیار دیگران هم قرارش بدی. برای اینکار از دستور **save()** استفاده کن و مسیر ذخیره و اسم فایل رو با پسوند HTML. بهش بده. قبلش نیاز هست که یه متغیر پشت کد بالا تعریف کنی.

Web\_Mahale = Mahale.explore("REGION", legend=False)

Web\_Mahale.save (r"E:\ArcGIS Pro\bin\Python\envs\gisenv\Shp\web.mahalat.html")

```
Web_Mahale = Mahale.explore("REGION", legend=False)
Web_Mahale.save (r"E:\ArcGIS Pro\bin\Python\envs\gisenv\Shp\web.mahalat.html")
```



اگه بخوای چندتا لایه رو روی هم بندازی به این شکل ازشون خروجی بگیری باید از دستور زیر استفاده کنی.

m = Mahale.explore(column= "REGION", legend=False)

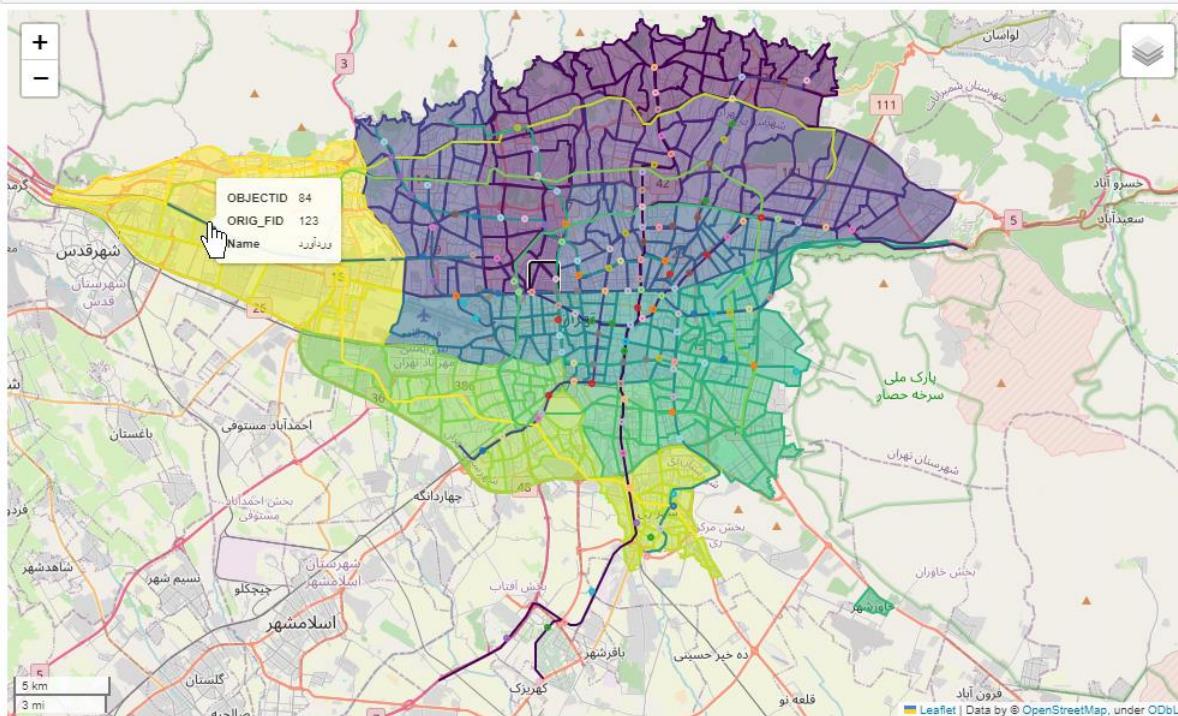
m = Metro\_Line.explore(m=m, name = "Metro", column= "Line", legend=False)

m = Metro\_St.explore(m=m, name = "Station", column= "Name", legend=False)

folium.LayerControl().add\_to(m)

m

```
m = Mahale.explore(column= "REGION", legend=False)
m = Metro_Line.explore(m=m, name = "Metro", column= "Line", legend=False)
m = Metro_St.explore(m=m, name = "Station", column= "Name", legend=False)
folium.LayerControl().add_to(m)
m
```



میتونی کل این لایه‌ها رو با هم به نام Web\_Layers مثل بالا ذخیره کنی.

`m.save(r"E:\ArcGIS Pro\bin\Python\envs\gisenv\Shp\Web_Layers.html")`

```
m.save (r"E:\ArcGIS Pro\bin\Python\envs\gisenv\Shp\Web_Layers.html")
```

« HD710 PRO (E:) » ArcGIS Pro » bin » Python » envs » gisenv » Shp		
	Name	Date modified
	name_m	11/27/2023 8:53 PM
	Traffic_M	11/27/2023 8:53 PM
	web.mahalat	11/28/2023 11:50 AM
	Web_Layers	11/27/2023 9:18 PM

تو جزوه بعدی دو تا کتابخونه مهم دیگه Geoplotlib و Rasterio رو با هم کار میکنیم.