Introduction to geospatial analysis using Python tools and Adzuna data

Transcript from webinar video recording

1

00:00:01,685 --> 00:00:04,346

So, in this webinar,

2

00:00:04,446 --> 00:00:09,178

we are going to use Adzuna datasets.

3

00:00:09,860 --> 00:00:13,402

I'll briefly show you all the information.

4

00:00:13,502 --> 00:00:17,555

So, Adzuna is a job aggregator website

5

00:00:18,767 --> 00:00:23,730

which has about 95 [percent of] job listing coverage in the UK.

6

00:00:23,830 --> 00:00:25,037

So, it's really easy to use.

7

00:00:25,137 --> 00:00:28,776

You just search by job, company, or location

8

00:00:29,597 --> 00:00:33,689

and you get an aggregation from different sites listed there.

9

00:00:35,470 --> 00:00:38,969

So, there are two datasets.

10

00:00:39,521 --> 00:00:42,982

One is open, collected using an Adzuna API.

11

00:00:43,476 --> 00:00:45,738

I added all the links in the presentation

12

00:00:45,916 --> 00:00:48,069

so you can follow along as well.

13

00:00:48,182 --> 00:00:52,473

So, if you wish to collect open Adzuna datasets,

14

00:00:52,573 --> 00:00:56,435

you can register for an API,

15

00:00:57,328 --> 00:00:59,408

receive the key, and just collect the data yourself.

16

00:01:00,398 --> 00:01:04,839

And this data, it's free and open data.

17

00:01:05,572 --> 00:01:09,549

To reduce the size of it, it has been subselected to location_area_1

18

00:01:09,779 --> 00:01:10,919

containing Scotland value.

19

00:01:11,019 --> 00:01:12,468

It's only sample data,

20

00:01:14,283 --> 00:01:15,882

mainly because of its size.

21

00:01:16,421 --> 00:01:21,480

But I'd also like to introduce a licensed dataset available

22

00:01:21,580 --> 00:01:25,596

for academic use at the Urban Big Data Centre website.

23

00:01:27,266 --> 00:01:30,093

So, click. So, this dataset is

24

00:01:30,462 --> 00:01:31,740

cleaned and duplicated.

25

00:01:32,678 --> 00:01:35,557

The only thing is you would need to apply for data to access it.

26

00:01:35,657 --> 00:01:39,716

So, the difference between these two data is one is open,

27

00:01:40,775 --> 00:01:44,113

it's messy, let's say,

28

00:01:44,793 --> 00:01:46,422

due to its nature of scraping,

29

00:01:46,691 --> 00:01:50,480

but there is work that could be done to derive insights.

30

00:01:50,580 --> 00:01:53,290

And the reason why this data is available for academic use,

31

00:01:53,508 --> 00:01:58,359

it has been deduplicated, cleaned up, and nicely formatted.

32

00:01:58,719 --> 00:02:01,489

The other thing is, in the data,

33

00:02:02,937 --> 00:02:07,171

because the description field containing information about the job ad,

34

00:02:07,629 --> 00:02:10,756

and in open data, it's truncated to 500 characters.

35

00:02:10,856 --> 00:02:16,946

So, quite often it's only a fraction of the job ad itself.

36

00:02:18,015 --> 00:02:22,384

Right. In this lab or training,

37

00:02:22,484 --> 00:02:24,922

we are going to use GeoPandas,

38

00:02:25,261 --> 00:02:27,438

which is an open source project

39

00:02:27,538 --> 00:02:32,436

which extends Pandas' frame as like a series of data frames

40

00:02:32,667 --> 00:02:34,967

to GeoSeries and GeoDataFrame.

41

00:02:35,137 --> 00:02:37,694

And the main difference between those two is that

42

00:02:38,660 --> 00:02:42,236

GeoDataFrame has a special column, geometry.

43

00:02:42,347 --> 00:02:45,705

Again, I've added the link for documentation

44

00:02:45,805 --> 00:02:47,794

if you want to explore further.

45

00:02:49,533 --> 00:02:52,533

So, let's start to open the data.

46

00:02:52,633 --> 00:02:55,582

And I'm going to run this session alongside with you.

47

00:02:55,790 --> 00:03:00,051

So, on this step, we are going to import Pandas

48

00:03:00,518 --> 00:03:01,795

to open our dataset,

49

00:03:02,263 --> 00:03:06,029

add the data provided, really the first data provided,

50

00:03:06,129 --> 00:03:07,696

using parquet format,

51

00:03:08,658 --> 00:03:10,835

which is an open source file format

52

00:03:11,234 --> 00:03:15,822

which is designed to efficiently store the information.

53

00:03:17,302 --> 00:03:20,039

I have included more information about

54

00:03:20,954 --> 00:03:23,822

how it works, its specifications, and other things.

55

00:03:25,295 --> 00:03:28,400

So, in this lab, we are using it because of

56

00:03:28,500 --> 00:03:29,741

the size of the data.

57

00:03:29,841 --> 00:03:32,722

So, it was just more convenient to pack it in a parquet

58

00:03:32,822 --> 00:03:34,344

and share it with you.

59

00:03:37,916 --> 00:03:40,525

So, I'm specifying my data source

60

00:03:41,536 --> 00:03:43,497

and I'm going to open it.

61

00:03:44,598 --> 00:03:45,968

Right. Let's do it like this.

62

00:03:46,709 --> 00:03:49,653

My source and load it to Pandas GeoDataFrame.

63

00:03:52,253 --> 00:03:54,734

Let's have a look at the data frame.

64

00:03:58,415 --> 00:04:02,330

So, this is what our Adzuna data looks like.

65

00:04:02,430 --> 00:04:04,492

It does have information about

66

00:04:05,393 --> 00:04:08,873

the job title, its category and tag.

67

00:04:08,973 --> 00:04:11,402

This is, by the way, created by Adzuna.

68

00:04:13,029 --> 00:04:18,298

There is some categorisation of the data available in open data.

69

00:04:19,169 --> 00:04:22,868

The dataset has information about contract type,

70

00:04:22,968 --> 00:04:25,186

well, in some cases.

71

00:04:25,697 --> 00:04:27,935

Information about minimum salary.

72

00:04:28,765 --> 00:04:29,966

About description.

73

00:04:30,066 --> 00:04:33,245

As I mentioned before, it's truncated to 500 characters.

74

00:04:34,595 --> 00:04:37,851

The date it was created and location information.

75

00:04:41,862 --> 00:04:45,570

So, in this lab, we are going to use longitude and latitude

76

00:04:45,805 --> 00:04:47,864

to make this data spatial.

77

00:04:50,014 --> 00:04:54,334

So, before we do any kind of GIS,

78

00:04:54,951 --> 00:04:57,399

analysis of GIS, we are just going to look at the data

79

00:04:57,588 --> 00:04:58,617

and try to import it.

80

00:04:58,717 --> 00:04:59,906

So, in this lab,

81

00:05:02,906 --> 00:05:05,965

on this step, I'm going to use Seaborn library

82

00:05:07,390 --> 00:05:10,088

for visualisation. You can see why shortly.

83

00:05:10,188 --> 00:05:14,446

It has very nice jointplot, type of visualisation

84

00:05:15,400 --> 00:05:19,429

to really showcase geographical data.

85

00:05:19,627 --> 00:05:21,210

And I'm going to use contextily,

86

00:05:21,310 --> 00:05:23,877

it's a library which provides background mapping.

87

00:05:24,214 --> 00:05:28,835

So, on this line, I specify that I want to create this type of plot,

88

00:05:28,935 --> 00:05:30,643

or the jointplot,

89

00:05:30,991 --> 00:05:35,727

where X would be longitude and Y latitude from my data frame.

90

00:05:36,227 --> 00:05:39,302

By default, I think, the colour is blue

91

00:05:39,402 --> 00:05:44,422

so I changed it to red to make the data more visible.

92

00:05:45,020 --> 00:05:49,515

On this line, I'm adding basemap from contextily

93

00:05:49,615 --> 00:05:52,159

and I'm saying that I'm adding it on the same plot.

94

00:05:52,259 --> 00:05:53,956

I'm specifying CRS.

95

00:05:54,078 --> 00:05:56,126

So, I'm going to talk about CRS,

96

00:05:56,226 --> 00:05:58,393

which stands for Coordinate Reference System

97

00:05:58,493 --> 00:06:00,091

in more detail later.

98

00:06:00,514 --> 00:06:02,083

And I'm specifying the source.

99

00:06:02,263 --> 00:06:07,471

Source is the type of background mapping I would like to see.

100

00:06:07,940 --> 00:06:10,958

Again, I'll talk more about it later

101

00:06:11,058 --> 00:06:14,366

and provide a few examples of what it could be.

102

00:06:14,576 --> 00:06:20,924

There, alpha is set to 0.4 and alpha is transparency.

103

00:06:22,234 --> 00:06:25,374

Again, on these lines I've made plot slightly bigger

104

00:06:25,738 --> 00:06:29,440

than it is by default so you can see the data better.

105

00:06:30,011 --> 00:06:34,395

So, if you're familiar with UK geography,

106

00:06:34,495 --> 00:06:38,347

you know that this area in red is Scotland.

107

00:06:38,447 --> 00:06:45,230

And several job ads, well, they have these kind of outliers,

108

00:06:45,330 --> 00:06:46,362

let's call them outliers,

109

00:06:46,714 --> 00:06:49,286

they do have Scotland as value and location

110

00:06:49,386 --> 00:06:52,577

but their coordinates are set to somewhere else

111

00:06:52,677 --> 00:06:53,939

so we'll remove them later.

112

00:06:54,577 --> 00:06:56,719

And if you see here,

113

00:06:56,800 --> 00:07:02,422

our jointplot shows the maximum distribution of that.

114

00:07:02,781 --> 00:07:08,192

And if you create intersection, you would see that it's Edinburgh

115

00:07:08,673 --> 00:07:10,294

and Glasgow,

116

00:07:10,404 --> 00:07:14,515

and most popular Third location is here in Aberdeenshire.

117

00:07:15,506 --> 00:07:17,534

Right. But it's important to say that,

118

00:07:17,634 --> 00:07:21,180

at this stage, this data is not spatial data yet.

119

00:07:22,331 --> 00:07:23,858

To make it spatial,

120

00:07:23,958 --> 00:07:26,744

we are going to convert the data frame into GeoDataFrame

121

00:07:26,844 --> 00:07:29,543

so we are able to perform geospatial analysis.

122

00:07:30,880 --> 00:07:32,575

We are going to import GeoPandas

123

00:07:32,675 --> 00:07:35,273

and the convention is to shorten the library names

124

00:07:35,373 --> 00:07:38,473

that we are going to import Pandas as GPD.

125

00:07:39,099 --> 00:07:46,043

Again, GeoPandas is a project which allows us to work with geospatial data

126

00:07:46,322 --> 00:07:47,813

and Python environment.

127

00:07:48,233 --> 00:07:53,106

On the background, it uses Shapely library.

128

00:07:53,206 --> 00:07:54,487

I've added documentation here.

129

00:07:54,587 --> 00:07:57,900

So, if you click the link, you can read about it later.

130

00:07:58,599 --> 00:08:02,756

And it uses matplotlib for plotting.

131

00:08:06,269 --> 00:08:07,683

So, let's import matplotlib

132

00:08:07,783 --> 00:08:10,645

as we are going to use it for plotting GeoPandas.

133

00:08:10,745 --> 00:08:15,315

And in this session, we are going to use Point and Polygon

134

00:08:15,415 --> 00:08:17,405

so I'm importing them as well.

135

00:08:20,325 --> 00:08:25,372

So, now we need to create our GeoDataFrame

136

00:08:25,472 --> 00:08:29,175

based on available X and Y.

137

00:08:29,275 --> 00:08:30,517

And the convention, really,

138

00:08:30,722 --> 00:08:35,389

the command is GeoPandas creates points from XY,

139

00:08:35,489 --> 00:08:40,979

which kind of says everything for itself.

140

00:08:41,600 --> 00:08:42,882

So, let's do this.

141

00:08:44,722 --> 00:08:47,314

So, on this step, I'm creating a GeoDataFrame

142

00:08:47,480 --> 00:08:49,178

and I'm calling it GDF.

143

00:08:49,278 --> 00:08:52,376

I'm saying GeoPandas data frame for command

144

00:08:52,974 --> 00:08:54,932

using my DF, data frame,

145

00:08:55,947 --> 00:08:59,734

the job ads data loaded on a previous step.

146

00:08:59,986 --> 00:09:01,704

And for geometry column,

147

00:09:01,912 --> 00:09:06,681

we are going to use a longitude column and a latitude column

148

00:09:06,781 --> 00:09:08,810

and create points based on them.

149

00:09:11,109 --> 00:09:13,661

So, let's see our data.

150

00:09:14,469 --> 00:09:17,378

So, we have exactly the same data as before.

151

00:09:18,307 --> 00:09:20,476

If you noticed on the previous steps,

152

00:09:20,576 --> 00:09:22,916

when we looked at the data we had only 23 columns

153

00:09:23,016 --> 00:09:27,264

and now we have one extra column called geometry

154

00:09:27,433 --> 00:09:29,231

which is a Point

155

00:09:29,761 --> 00:09:33,349

and it has information similar to what we have here.

156

00:09:34,564 --> 00:09:37,114

So, now we have a spatial dataset.

157

00:09:39,075 --> 00:09:40,754

Let's check our Coordinate Reference system.

158

00:09:42,472 --> 00:09:45,860

We could have specified it on a previous step.

159

00:09:46,336 --> 00:09:49,053

If it's not specified, by default it's set to none.

160

00:09:51,432 --> 00:09:54,601

And in this section, I'm going to talk very, very briefly about

161

00:09:54,701 --> 00:09:56,847

map projections and Coordinate Reference System

162

00:09:56,947 --> 00:10:01,266

for those who are not familiar with this kind of concept,

163

00:10:01,484 --> 00:10:04,521

which is very, very crucial in the GIS world.

164

00:10:04,701 --> 00:10:08,649

Map projection, in very simple terms,

165

00:10:08,749 --> 00:10:11,745

tries to transform the Earth from its spherical shape

166

00:10:12,053 --> 00:10:16,091

to a flat, planar shape or flat map.

167

00:10:16,281 --> 00:10:18,371

And Coordinate Reference System,

168

00:10:18,471 --> 00:10:22,538

CRS for short, defines how this two-dimensional space

169

00:10:22,638 --> 00:10:25,076

projected on a map in your geographical system

170

00:10:25,176 --> 00:10:27,655

relates to real places on Earth.

171

00:10:28,215 --> 00:10:34,041

So, the decision of which map projection and Coordinate Reference System to use

172

00:10:34,141 --> 00:10:37,511

depends on the region and country.

173

00:10:40,688 --> 00:10:43,866

I've included a source link where you can read more.

174

00:10:43,976 --> 00:10:49,389

But please bear in mind it's very, very important for GIS.

175

00:10:50,630 --> 00:10:53,158

So, I have included a few visual examples

176

00:10:53,258 --> 00:10:55,706

just to showcase it.

177

00:10:56,105 --> 00:10:58,182

Depending on the choice, you could have

178

00:10:58,559 --> 00:11:01,537

different distortions.

179

00:11:01,637 --> 00:11:04,275

In this case, it's distortion by area.

180

00:11:05,706 --> 00:11:11,694

And this is another example just to illustrate how the shape changes.

181

00:11:11,794 --> 00:11:14,122

But it could also change distances and angles

182

00:11:14,222 --> 00:11:15,350

depending on the projection.

183

00:11:15,450 --> 00:11:21,659

But what you see on the screen would change based on what you specified.

184

00:11:22,040 --> 00:11:25,956

Again, there is a source to check

185

00:11:26,056 --> 00:11:30,885

and maybe learn a bit more about it on your own.

186

00:11:31,114 --> 00:11:36,683

So, now we are going to set coordinates to WGS84

187

00:11:36,996 --> 00:11:39,130

using the set_crs function,

188

00:11:40,080 --> 00:11:42,637

which takes EPSG codes,

189

00:11:42,737 --> 00:11:46,196

which stands for European Petroleum Survey Group,

190

00:11:47,111 --> 00:11:49,629

and value of the system.

191

00:11:50,129 --> 00:11:55,577

So, for WGS84, the value of EPSG is 4326.

192

00:11:57,407 --> 00:12:01,457

And I'm printing it as well just to check that it worked.

193

00:12:02,677 --> 00:12:07,066

I've also included for you several quite useful links, I think.

194

00:12:08,175 --> 00:12:10,903

So, one is Spatial Reference where you can find

195

00:12:12,063 --> 00:12:15,332

many, many, many codes and references,

196

00:12:15,432 --> 00:12:17,715

coordinate references. So, we'll just open one.

197

00:12:18,385 --> 00:12:23,697

So, it does contain Bounds or a link

198

00:12:25,047 --> 00:12:26,220

and a Google.

199

00:12:26,334 --> 00:12:29,088

And the other one is also quite useful.

200

00:12:30,241 --> 00:12:31,622

Let's just open it.

201

00:12:31,892 --> 00:12:34,574

It does also have visual representations of

202

00:12:34,654 --> 00:12:38,587

what you would expect your data to be like

203

00:12:38,687 --> 00:12:39,849

after you project it.

204

00:12:40,079 --> 00:12:43,451

And also, it does contain transformational values.

205

00:12:44,990 --> 00:12:46,082

Let's move on.

206

00:12:46,853 --> 00:12:50,195

So, now we've set up our coordinate reference system

207

00:12:50,295 --> 00:12:52,104

and we have our geospatial data,

208

00:12:52,204 --> 00:12:54,895

let's do some visualisation.

209

00:12:54,995 --> 00:12:57,374

So, we are going to use matplotlib

210

00:12:57,474 --> 00:12:59,963

and the function dot plot.

211

00:13:00,703 --> 00:13:01,732

And by default,

212

00:13:03,471 --> 00:13:07,010

the map in GeoPandas is created based on the geometry column.

213

00:13:07,479 --> 00:13:08,587

Let's run it.

214

00:13:10,978 --> 00:13:12,197

So, it might take a while.

215

00:13:12,297 --> 00:13:15,757

What I'm saying is I want to plot my GeoDataFrame,

216

00:13:16,566 --> 00:13:19,556

I want my dots to be red,

217

00:13:19,656 --> 00:13:20,835

and I want it to show.

218

00:13:20,935 --> 00:13:23,123

So, if you see here, it's a small plot

219

00:13:23,223 --> 00:13:25,701

but you can see the outline of Scotland here.

220

00:13:25,999 --> 00:13:27,207

Similar to what we've seen

221

00:13:29,128 --> 00:13:34,627

when we were visualising data using just X and Y and Seaborn.

222

00:13:36,152 --> 00:13:40,970

On this step, I would like to show how to open different file formats of

223

00:13:41,070 --> 00:13:42,725

spatial data in Pandas.

224

00:13:43,154 --> 00:13:47,323

So, in this step, we are going to load the boundary of Scotland

225

00:13:48,879 --> 00:13:53,933

using our open and free Ordinance Survey data BoundaryLine,

226

00:13:54,839 --> 00:13:57,787

which is available for download for everybody.

227

00:13:57,887 --> 00:13:59,195

So, it's free to download here.

228

00:13:59,525 --> 00:14:04,141

And it is a collection of data available.

229

00:14:04,711 --> 00:14:07,409

Trying to find the description.

230

00:14:07,509 --> 00:14:09,318

So, it does contain information about

231

00:14:09,418 --> 00:14:12,135

boundaries of civil parishes, wards, communities,

232

00:14:12,712 --> 00:14:14,159

electoral divisions.

233

00:14:14,779 --> 00:14:16,456

So, it does contain a lot of information

234

00:14:16,556 --> 00:14:19,305

and it's available for download.

235

00:14:21,345 --> 00:14:24,533

So, to read the file,

236

00:14:25,377 --> 00:14:27,726

you just need to specify GeoPandas, read file,

237

00:14:28,285 --> 00:14:31,570

and what it does, the library reads the data

238

00:14:31,670 --> 00:14:33,867

and it returns it as a GeoDataFrame object.

239

00:14:35,248 --> 00:14:36,407

Run it here.

240

00:14:36,596 --> 00:14:39,995

So, in mine, I used GeoDataFrame OS data

241

00:14:40,095 --> 00:14:41,323

and I'm saying read file,

242

00:14:42,174 --> 00:14:44,212

which is saved in the data folder.

243

00:14:44,440 --> 00:14:46,701

So, it's district_borough_unitary shape.

244

00:14:47,433 --> 00:14:52,442

So, the full dataset provided is GeoPackage with many layers

245

00:14:52,542 --> 00:14:55,181

so that it's packaged to reduce the size

246

00:14:55,281 --> 00:14:58,980

and make it easier for you to work with the data.

247

00:14:59,090 --> 00:15:01,530

I preloaded it into a shapefile.

248

00:15:02,409 --> 00:15:04,458

If you decide to load the full dataset,

249

00:15:04,558 --> 00:15:07,057

please command on this line and use this.

250

00:15:07,314 --> 00:15:09,942

So, very similar, we are reading the file

251

00:15:10,723 --> 00:15:12,231

and for that, set it to GeoPackage

252

00:15:12,431 --> 00:15:15,960

and its name and specify the way we need it to be loaded

253

00:15:16,060 --> 00:15:17,988

so it would provide the same result.

254

00:15:19,969 --> 00:15:22,596

Let's check the Coordinate Reference System.

255

00:15:23,145 --> 00:15:27,154

So, this data is projected dataset to the British National Grid,

256

00:15:27,254 --> 00:15:30,302

EPSG code 27770.

257

00:15:32,314 --> 00:15:35,712

Right. Let's visualise our datasets together.

258

00:15:35,812 --> 00:15:40,220

So, I'm going to load my OS data

259

00:15:40,320 --> 00:15:43,387

and my Adzuna and job listings together.

260

00:15:44,529 --> 00:15:47,133

It might take some time.

261

00:15:48,814 --> 00:15:51,789

Again, I specified my Adzuna data to be red.

262

00:15:51,889 --> 00:15:54,458

And if you see it, it doesn't look right.

263

00:15:54,558 --> 00:16:01,297

So, we have our UK and our Adzuna data here,

264

00:16:02,154 --> 00:16:04,122

over here, but we would expect it there.

265

00:16:07,128 --> 00:16:11,468

This is one of the typical and common problems which could

266

00:16:11,595 --> 00:16:14,123

occur when you use GeoPandas.

267

00:16:14,223 --> 00:16:16,637

It is mismatch of projection.

268

00:16:17,078 --> 00:16:20,406

Do you remember our GeoDataFrame was WGS84?

269

00:16:21,035 --> 00:16:25,594

And OS data is British National Grid.

270

00:16:25,694 --> 00:16:28,633

So, if you visualise your data and it's something like this,

271

00:16:28,982 --> 00:16:30,909

the first thing would be to check the projection.

272

00:16:31,009 --> 00:16:34,617

So, on this step, let's make the plot bigger

273

00:16:34,717 --> 00:16:39,487

and fix the projection issue using to_crs Scotland.

274

00:16:40,576 --> 00:16:43,454

So, I'm going to run it.

275

00:16:44,205 --> 00:16:49,494

So, on this line, I'm setting my plot to a bigger size

276

00:16:49,594 --> 00:16:54,963

and I'm going to reproject my both datasets to Web Mercator

277

00:16:55,095 --> 00:16:58,961

to align it with the background map. So, on this line, I'm reprojecting

278

00:16:59,061 --> 00:17:01,660

my OS data using command to_crs,

279

00:17:01,760 --> 00:17:04,259

specifying a new Coordinate Reference System.

280

00:17:05,389 --> 00:17:10,498

Similarly, on this line, I'm doing the same with my Adzuna data

281

00:17:10,598 --> 00:17:13,276

from the GeoDataFrame and I'm setting the colour as red

282

00:17:13,376 --> 00:17:14,504

so we can clearly see it.

283

00:17:14,704 --> 00:17:17,012

And on this line, I'm adding basemap.

284

00:17:18,772 --> 00:17:22,802

So, now our visualisation is better.

285

00:17:22,902 --> 00:17:24,301

So, data aligned.

286

00:17:26,511 --> 00:17:31,730

And what I want to talk about next is

287

00:17:31,830 --> 00:17:33,488

contextily basemaps.

288

00:17:34,305 --> 00:17:40,767

So, there are different ways you can use reprojection

289

00:17:40,867 --> 00:17:42,455

and align with this basemap.

290

00:17:42,555 --> 00:17:44,283

So, just a reminder that,

291

00:17:44,590 --> 00:17:47,894

on this step, we projected both layers

292

00:17:48,382 --> 00:17:51,341

and didn't do any reprojection of the basemap file.

293

00:17:51,441 --> 00:17:53,379

It could have been done in different ways.

294

00:17:53,499 --> 00:17:55,967

So, we could have reprojected only one layer

295

00:17:56,067 --> 00:17:58,882

to match the Coordinate Reference System

296

00:17:58,982 --> 00:18:00,120

of another one

297

00:18:00,890 --> 00:18:05,478

and then specify the same EPSG code in our basemap.

298

00:18:05,888 --> 00:18:08,936

And it's the output we would expect.

299

00:18:09,036 --> 00:18:12,543

So, the data is, it's the same location

300

00:18:12,643 --> 00:18:15,778

but it is visualised in a slightly different way.

301

00:18:16,289 --> 00:18:20,007

What I mean by this is, try to compare this picture

302

00:18:20,107 --> 00:18:21,795

and this.

303

00:18:22,093 --> 00:18:24,959

So, the data is the same, just a different Coordinate Reference System

304

00:18:25,059 --> 00:18:28,905

and a slightly different visualisation.

305

00:18:29,023 --> 00:18:33,810

So, now I'd like to show you available backgrounds.

306

00:18:33,910 --> 00:18:38,719

So, if you specify contextily providing this,

307

00:18:38,819 --> 00:18:43,663

you would get a list of all available background map providers.

308

00:18:45,350 --> 00:18:51,279

And then, if you add contextily providers

309

00:18:51,379 --> 00:18:54,348

and instead of keyname add something like "HERE"

310

00:18:54,448 --> 00:18:56,456

and said dot keys,

311

00:18:56,925 --> 00:18:59,692

you would get a list of

312

00:19:00,159 --> 00:19:01,889

these background maps and visualisations of

313

00:19:01,989 --> 00:19:04,437

different styles available from these providers.

314

00:19:04,537 --> 00:19:06,987

So, in this example for provider HERE,

315

00:19:07,087 --> 00:19:11,902

we have this many different styles

316

00:19:12,630 --> 00:19:14,127

available for you to use.

317

00:19:14,687 --> 00:19:16,707

Right. Spatial Join.

318

00:19:16,997 --> 00:19:22,076

This is quite a popular operation in GIS.

319

00:19:22,366 --> 00:19:26,724

I've added a link for how it works in GeoPandas.

320

00:19:26,954 --> 00:19:28,723

So, here it is, spatial join.

321

00:19:29,084 --> 00:19:30,904

It's different arguments.

322

00:19:31,744 --> 00:19:35,523

Again, it's something for you to have.

323

00:19:38,839 --> 00:19:42,378

As always, let's look at the data.

324

00:19:42,478 --> 00:19:45,604

And on this step, we are going to remove outliers

325

00:19:45,873 --> 00:19:49,292

but outside of Scotland

326

00:19:49,392 --> 00:19:52,236

and limit it to make it smaller and faster.

327

00:19:52,336 --> 00:19:55,275

We're going to remove everything outside of Glasgow.

328

00:19:55,855 --> 00:19:57,988

So, just a quick recap.

329

00:19:58,000 --> 00:20:00,749

So, our GeoDataFrame with Adzuna listings

330

00:20:02,007 --> 00:20:04,386

contains 24 columns now.

331

00:20:04,586 --> 00:20:07,875

23 of which are the Adzuna data itself

332

00:20:08,083 --> 00:20:11,523

and one is the geometry that we created.

333

00:20:14,140 --> 00:20:18,860

I'm going to subselect my Ordinance Survey data

334

00:20:19,319 --> 00:20:21,796

only to have Glasgow.

335

00:20:22,656 --> 00:20:24,044

Sorry, Glasgow.

336

00:20:24,244 --> 00:20:26,392

That's set and I'm going to go with Glasgow.

337

00:20:28,481 --> 00:20:30,169

Will it just quickly work?

338

00:20:30,998 --> 00:20:33,157

What it looks like. Again, if you're familiar,

339

00:20:33,257 --> 00:20:35,924

this is a Glasgow boundary.

340

00:20:36,355 --> 00:20:37,822

Just a visual check.

341

00:20:38,802 --> 00:20:41,982

Let's check what Coordinate Reference System it is in.

342

00:20:42,201 --> 00:20:45,459

So, it is in Web Mercator.

343

00:20:45,827 --> 00:20:47,636

If you remember, on the previous step,

344

00:20:47,823 --> 00:20:50,218

we reprojected OS data

345

00:20:51,189 --> 00:20:56,725

and Glasgow GeoDataFrame inherited it.

346

00:20:57,885 --> 00:21:00,154

Right. Let's make spatial join.

347

00:21:01,004 --> 00:21:06,001

So, to make a join, we need to specify our GeoDataFrame

348

00:21:06,341 --> 00:21:07,449

we'd like to join.

349

00:21:07,549 --> 00:21:09,788

And in this example, we are going to select

350

00:21:09,888 --> 00:21:11,497

everything which falls within.

351

00:21:11,797 --> 00:21:14,627

And, if you can see, we've got an error.

352

00:21:15,705 --> 00:21:17,885

What do you think caused an error?

353

00:21:21,354 --> 00:21:22,412

Sorry.

354

00:21:22,583 --> 00:21:24,641

So, if you read here the user warning,

355

00:21:24,829 --> 00:21:28,628

it says it's a mismatch between left and right geometry.

356

00:21:29,728 --> 00:21:36,575

And it says that our left GeoDataFrame is in WGS84, 4326,

357

00:21:36,675 --> 00:21:39,100

and our Glasgow data frame has

358

00:21:39,200 --> 00:21:41,858

the EPSG code 3857, which is Web Mercator.

359

00:21:41,958 --> 00:21:46,696

So, because there are different projections,

360

00:21:46,796 --> 00:21:49,414

GeoPandas cannot figure it out and throw an exception.

361

00:21:49,514 --> 00:21:52,952

So, what we need to do is to reproject one of the layers

362

00:21:53,052 --> 00:21:54,911

to match the projection of another.

363

00:21:56,099 --> 00:21:57,299

Let's run it.

364

00:22:00,621 --> 00:22:03,611

It might take some time if you run it on your computer.

365

00:22:05,111 --> 00:22:08,407

So, again, to remind you,

366

00:22:08,507 --> 00:22:12,747

we are going to select all the job ads within Glasgow.

367

00:22:12,985 --> 00:22:14,670

Let's have a look at what we've got.

368

00:22:15,249 --> 00:22:17,138

We're going to spatially join

369

00:22:19,537 --> 00:22:22,667

our Adzuna dataset and Glasgow boundary.

370

00:22:23,948 --> 00:22:25,015

Right.

371

00:22:26,085 --> 00:22:29,273

I'm going to check the length of my join.

372

00:22:29,443 --> 00:22:32,021

So, now it's quite a small and manageable dataset.

373

00:22:33,815 --> 00:22:35,695

So, I'm going to remember this number

374

00:22:35,795 --> 00:22:38,005

because I'm going to use it for something else as well.

375

00:22:38,535 --> 00:22:42,203

Right. Let's check what we have inside.

376

00:22:42,432 --> 00:22:46,913

And the first thing you would notice is now we have 41 columns.

377

00:22:47,280 --> 00:22:49,059

So, what happened here,

378

00:22:49,889 --> 00:22:54,826

our Adzuna dataset has been spatially joined with Glasgow

379

00:22:54,926 --> 00:22:58,132

and all the records are saved information

380

00:22:58,506 --> 00:23:01,455

from the data layer related to this location.

381

00:23:01,743 --> 00:23:05,133

In this case, it's only information about

382

00:23:05,792 --> 00:23:06,871

our Glasgow...

383

00:23:08,319 --> 00:23:10,688

Sorry. The same information for all the points

384

00:23:10,788 --> 00:23:13,427

but it could be used in a more powerful way,

385

00:23:13,527 --> 00:23:14,914

which I'll show you later.

386

00:23:17,204 --> 00:23:23,604

But an important thing to note and to take from it is

387

00:23:23,704 --> 00:23:28,681

that by spatially joining two of our GeoDataFrames,

388

00:23:28,781 --> 00:23:31,726

we receive information from both of them.

389

00:23:33,984 --> 00:23:36,303

In this step, I really wanted to show that

390

00:23:36,862 --> 00:23:41,162

you can use your GeoDataFrame in a similar way as your data frame

391

00:23:41,262 --> 00:23:43,807

and you can use typical functions.

392

00:23:43,907 --> 00:23:45,013

So, here, for instance,

393

00:23:45,213 --> 00:23:49,037

I used a group by column to find

394

00:23:49,325 --> 00:23:55,250

the most paid categories

395

00:23:56,037 --> 00:24:00,458

by finding the average minimum salary

396

00:24:00,736 --> 00:24:03,904

and then sorting values by the salary.

397

00:24:04,254 --> 00:24:06,812

So, in this step,

398

00:24:06,910 --> 00:24:11,538

we just did a summary of our GeoSpatial data frame of

399

00:24:11,638 --> 00:24:12,795

geographical data

400

00:24:15,229 --> 00:24:16,810

and received some statistics.

401

00:24:17,519 --> 00:24:20,077

Right. Let's quickly visualise data.

402

00:24:20,177 --> 00:24:24,236

And I would like to introduce you to, in this step, to geoplot,

403

00:24:26,105 --> 00:24:30,161

which is a fantastic library for geospatial data visualisation.

404

00:24:30,650 --> 00:24:35,724

I will quickly show you the gallery, just to introduce

405

00:24:36,221 --> 00:24:41,880

the different visualisation styles and techniques available from here.

406

00:24:42,859 --> 00:24:46,968

Again, please have a look when you have some time.

407

00:24:47,630 --> 00:24:50,846

And, as always, we need to import this library.

408

00:24:54,355 --> 00:24:56,695

And here I'm going to use pointplot,

409

00:24:58,854 --> 00:25:03,812

where my colour would depend on the value in the salary min column.

410

00:25:05,362 --> 00:25:10,101

So, the data parameter is just legend true to have the legend here.

411

00:25:10,220 --> 00:25:13,067

Edge colour is set to light grey but you can actually see it

412

00:25:13,167 --> 00:25:15,111

so it's edge of the point.

413

00:25:15,575 --> 00:25:17,420

And its line width.

414

00:25:17,770 --> 00:25:20,450

So, we do have visualisation

415

00:25:20,550 --> 00:25:26,758

but clearly most of the values fell in this region

416

00:25:26,858 --> 00:25:29,386

and we can't actually see a difference between

417

00:25:30,429 --> 00:25:32,497

different areas of Glasgow in terms of salary min.

418

00:25:34,216 --> 00:25:36,024

To fix this or to enhance it,

419

00:25:36,272 --> 00:25:38,873

I'm going to use mapclassify.

420

00:25:41,440 --> 00:25:44,909

Mapclassify is a library which provides classification schemes

421

00:25:45,158 --> 00:25:47,017

there are different parameters.

422

00:25:48,916 --> 00:25:50,044

You will see.

423

00:25:50,474 --> 00:25:52,732

So, I'm going to import this mapclassify.

424

00:25:52,832 --> 00:25:54,480

The library is mc.

425

00:25:54,562 --> 00:25:58,182

And I'm going to use scheme Quantiles

426

00:25:58,550 --> 00:26:02,679

and my salary min column from join.

427

00:26:02,769 --> 00:26:05,617

Now, as I've said before,

428

00:26:06,506 --> 00:26:08,314

this part is the same.

429

00:26:08,414 --> 00:26:10,683

The only change I made in here is

430

00:26:10,783 --> 00:26:12,402

to specify the scheme I'd like to use.

431

00:26:16,655 --> 00:26:18,285

So, the legend changed.

432

00:26:18,464 --> 00:26:22,201

But now you can see that the data has been reclassified

433

00:26:22,301 --> 00:26:28,383

and now we can see the distribution based on the salary min column.

434

00:26:28,781 --> 00:26:33,653

Again, there are many schemes and classification types available.

435

00:26:33,753 --> 00:26:37,971

It's a very, very powerful to create nice looking visualisations.

436

00:26:39,626 --> 00:26:40,714

Right.

437

00:26:40,941 --> 00:26:46,169

The other geospatial function I'd like to introduce is clip.

438

00:26:50,530 --> 00:26:55,237

GeoPandas does clip to what you would expect to get

439

00:26:55,337 --> 00:26:57,456

in traditional GIS.

440

00:26:57,556 --> 00:27:00,554

So, one layer is clipped by another

441

00:27:01,143 --> 00:27:03,383

based on the geometries.

442

00:27:03,562 --> 00:27:05,931

Again, very important, both layers must be

443

00:27:06,031 --> 00:27:08,099

the same Coordinate Reference System.

444

00:27:11,890 --> 00:27:15,590

So, for that reason, let's check what we have now.

445

00:27:15,690 --> 00:27:19,888

So, one of our layers is in Web Mercator

446

00:27:19,988 --> 00:27:23,056

and the other one is in WGS84.

447

00:27:25,317 --> 00:27:30,025

So, let's reproject both layers to Web Mercator.

448

00:27:32,031 --> 00:27:34,801

Actually, now I'm thinking we do not need to reproject one of them.

449

00:27:35,270 --> 00:27:37,089

We need to reproject only one of them.

450

00:27:39,299 --> 00:27:41,225

And let's do the clip.

451

00:27:41,402 --> 00:27:44,641

So, we are going to clip

452

00:27:44,741 --> 00:27:47,680

our Adzuna job listings stored in GDF

453

00:27:47,780 --> 00:27:49,389

by the Glasgow boundary.

454

00:27:49,786 --> 00:27:52,406

And, again, remove everything else outside.

455

00:27:52,694 --> 00:27:56,726

So, let's check the length of our dataset.

456

00:27:56,826 --> 00:28:00,208

So, if you remember, it is exactly the same number

457

00:28:00,308 --> 00:28:04,355

as we had when we were joining the data.

458

00:28:04,576 --> 00:28:06,173

The length looked quite similar.

459

00:28:06,373 --> 00:28:09,102

Let's look at the clipped GeoDataFrame.

460

00:28:11,159 --> 00:28:14,187

And, again, there are only 24 columns.

461

00:28:14,287 --> 00:28:16,220

So, the way clip works is

462

00:28:16,320 --> 00:28:22,339

it subselects all the features

463

00:28:22,439 --> 00:28:23,986

within other features.

464

00:28:24,118 --> 00:28:29,225

So, what we have got here, we've got our Adzuna job ads

465

00:28:29,651 --> 00:28:32,070

subselected or clipped only

466

00:28:33,163 --> 00:28:34,643

to be within Glasgow.

467

00:28:35,542 --> 00:28:38,990

So, no extra additional information is added

468

00:28:39,567 --> 00:28:40,917

when using clip.

469

00:28:41,858 --> 00:28:43,376

Now, let's plot the data.

470

00:28:45,983 --> 00:28:49,813

Again, I'm specifying my figure size on this line.

471

00:28:50,427 --> 00:28:52,925

I'm plotting my clipped GeoDataFrame,

472

00:28:53,025 --> 00:28:54,163

my Glasgow GeoDataframe,

473

00:28:54,263 --> 00:28:58,940

set in some transparency so we can see both datasets.

474

00:28:59,040 --> 00:29:01,588

And I'm adding my basemap.

475

00:29:01,993 --> 00:29:04,511

Now, I'm using OpenStreetMap Mapnik.

476

00:29:04,864 --> 00:29:06,306

This is what we've got.

477

00:29:07,879 --> 00:29:12,281

If you can see, no point is outside of the Glasgow boundary.

478

00:29:15,518 --> 00:29:19,184

So, on this step, I just really wanted to show you that

479

00:29:19,994 --> 00:29:23,093

using join and clip in this way

480

00:29:23,717 --> 00:29:25,006

provided us with the same result

481

00:29:25,106 --> 00:29:29,844

and show the way it works if you run assert statement

482

00:29:29,944 --> 00:29:32,432

and it doesn't return anything or returns true.

483

00:29:33,821 --> 00:29:35,689

So, if it doesn't return any output,

484

00:29:35,789 --> 00:29:37,947

it means the statement is true.

485

00:29:38,047 --> 00:29:40,754

And if it throws an error, it means it's false.

486

00:29:41,086 --> 00:29:42,545

So, in our case,

487

00:29:43,235 --> 00:29:46,722

both geospatial functions provided the same result.

488

00:29:49,622 --> 00:29:51,139

I don't have to explain this slide.

489

00:29:51,239 --> 00:29:53,337

So, when you work with geoplot,

490

00:29:53,437 --> 00:29:56,775

what you could often come across is

491

00:29:57,481 --> 00:30:03,529

some clearly wrong result but you do not get any errors.

492

00:30:04,209 --> 00:30:06,997

And it's a little helpful hint.

493

00:30:07,097 --> 00:30:08,601

So, in geoplot,

494

00:30:11,144 --> 00:30:15,542

coordinates need to be within this range.

495

00:30:15,642 --> 00:30:20,830

So, they should be between minus 180 and plus 180.

496

00:30:20,930 --> 00:30:22,548

I'll showcase what I mean here.

497

00:30:24,098 --> 00:30:27,464

So, I'm going to create, clip, and reproject it.

498

00:30:28,051 --> 00:30:30,472

I'm going to create a new GeoDataFrame

499

00:30:31,660 --> 00:30:34,658

by reprojecting an available clipped data frame.

500

00:30:36,626 --> 00:30:38,764

And I want to show you

501

00:30:39,672 --> 00:30:43,546

bounding boxes of our newly reprojected data frame

502

00:30:44,315 --> 00:30:46,783

and our existing clipped data frame.

503

00:30:46,883 --> 00:30:48,057

So, if you can see,

504

00:30:49,125 --> 00:30:53,212

this GeoDataFrame and bounding box are

505

00:30:53,312 --> 00:30:56,779

within minus 180 and plus 180.

506

00:30:57,259 --> 00:30:59,156

And because of the different projections of

507

00:30:59,256 --> 00:31:00,311

the clipped data frame,

508

00:31:00,411 --> 00:31:05,310

they are out of these limits.

509

00:31:05,804 --> 00:31:07,977

Again, a bit of an explanation.

510

00:31:08,077 --> 00:31:12,605

So, if you plot our clipped GeoDataFrame,

511

00:31:14,323 --> 00:31:16,272

because it does fall within those bounds,

512

00:31:16,372 --> 00:31:17,741

you would get some visualisation.

513

00:31:17,841 --> 00:31:19,949

So, in this case,

514

00:31:20,713 --> 00:31:26,550

this is our clipped GeoDataFrame

515

00:31:26,650 --> 00:31:30,007

where the size of the points depends on

516

00:31:30,107 --> 00:31:33,114

the information stored in the salary min column.

517

00:31:33,394 --> 00:31:37,091

Limits just specify the minimum and maximum size of the points.

518

00:31:38,005 --> 00:31:40,413

So, figure size is just the size of the plot.

519

00:31:40,513 --> 00:31:42,409

Alpha is the transparency.

520

00:31:42,946 --> 00:31:44,621

Legend is set to true again.

521

00:31:46,851 --> 00:31:50,085

Probably, it's better to reduce the size from 50 to something else

522

00:31:50,185 --> 00:31:51,591

so it does fit better.

523

00:31:53,472 --> 00:31:58,759

And, on this line, I'm also adding my Glasgow boundary on this plot.

524

00:31:59,309 --> 00:32:04,288

Now, let's take a look if we use the clipped GeoDataFrame

525

00:32:04,572 --> 00:32:08,743

which bounding boxes are outside of those values.

526

00:32:08,843 --> 00:32:12,562

And see, you only have Glasgow,

527

00:32:14,690 --> 00:32:18,308

the Glasgow outline but nothing from the clipped data frame.

528

00:32:18,408 --> 00:32:21,548

And it doesn't throw you any errors.

529

00:32:21,918 --> 00:32:26,068

So, if anything like this happens when you're using geoplot,

530

00:32:26,168 --> 00:32:29,256

just bear in mind that you might need to use different projections

531

00:32:29,356 --> 00:32:33,955

so your bounding box is within those limits.

532

00:32:34,205 --> 00:32:36,703

Just some hints I wanted to share.

533

00:32:39,258 --> 00:32:42,975

In these many examples,

534

00:32:43,353 --> 00:32:47,291

we would look into transformation between GeoDataFrame and data frame.

535

00:32:47,995 --> 00:32:50,951

As I mentioned before, you can use

536

00:32:52,125 --> 00:32:56,622

most Pandas functions in your GeoDataFrame

537

00:32:57,939 --> 00:33:00,280

And I'm going to add in another dataset.

538

00:33:01,018 --> 00:33:03,018

It is Data Zone boundaries.

539

00:33:03,854 --> 00:33:05,712

I've added the link here.

540

00:33:05,912 --> 00:33:10,027

So, it's an open dataset containing Data Zones,

541

00:33:10,216 --> 00:33:11,835

which, if you're not familiar,

542

00:33:11,935 --> 00:33:16,042

it's a kind of statistical area in Scotland.

543

00:33:16,974 --> 00:33:20,754

So, the data provided on the website comes in a shapefile

544

00:33:20,854 --> 00:33:23,533

but I have converted it to geojson

545

00:33:24,150 --> 00:33:27,097

just to show how easy it is to open this file format.

546

00:33:27,197 --> 00:33:30,266

And it's quite a typical file format

547

00:33:31,704 --> 00:33:33,115

available and used.

548

00:33:33,845 --> 00:33:38,127

So, again, the Data Zone boundaries are clipped for Glasgow,

549

00:33:38,227 --> 00:33:40,438

just to give you the size, so let's open it.

550

00:33:41,799 --> 00:33:45,000

So, similarly, we just say in GeoPandas

551

00:33:45,100 --> 00:33:47,440

to read file and specify the file location.

552

00:33:48,740 --> 00:33:51,472

Let's check the Coordinate Reference System.

553

00:33:51,572 --> 00:33:53,214

So, it's British National Grid,

554

00:33:53,964 --> 00:33:56,631

code 27700.

555

00:34:00,196 --> 00:34:02,846

Because I'm going to use it with my GeoDataFrame

556

00:34:02,946 --> 00:34:04,415

containing Adzuna ads,

557

00:34:04,580 --> 00:34:07,296

I want to check what it is now,

558

00:34:07,396 --> 00:34:09,029

after the several reprojections we made.

559

00:34:09,129 --> 00:34:11,407

So, it is in Web Mercator code

560

00:34:12,386 --> 00:34:14,172

3857.

561

00:34:14,752 --> 00:34:20,450

So, let's convert our GeoDataFrame to match Data Zones,

562

00:34:20,869 --> 00:34:23,477

specifying CRS Scotland

563

00:34:23,577 --> 00:34:26,175

and the British National Grid code.

564

00:34:29,975 --> 00:34:32,273

Let's look at our Data Zones.

565

00:34:34,191 --> 00:34:36,835

So, you can see this is a Glasgow boundary

566

00:34:36,935 --> 00:34:40,923

and small, tiny polygons inside which we are going to use.

567

00:34:43,009 --> 00:34:44,738

So, what I want to do is

568

00:34:44,838 --> 00:34:47,615

to make a spatial join again.

569

00:34:47,715 --> 00:34:49,189

But this time I'm going to use

570

00:34:49,289 --> 00:34:52,919

my Adzuna data and my Data Zone,

571

00:34:54,444 --> 00:34:57,863

assigning values of a Data Zone to

572

00:34:57,963 --> 00:35:02,342

every point of Adzuna job ads based on their location.

573

00:35:03,192 --> 00:35:06,571

Saying that the operation we're going to use for this is

574

00:35:06,840 --> 00:35:09,428

the point should be within the border, completely within.

575

00:35:11,898 --> 00:35:16,067

So, depending on the spec of your computer,

576

00:35:16,336 --> 00:35:18,623

it might be faster or take longer

577

00:35:18,723 --> 00:35:20,153

so be patient.

578

00:35:21,399 --> 00:35:22,905

Let's have a look at the data.

579

00:35:23,896 --> 00:35:28,721

So, we do see that we have received extra columns.

580

00:35:29,321 --> 00:35:30,753

Let's have a look.

581

00:35:31,892 --> 00:35:36,078

So, we have our Adzuna data here

582

00:35:36,499 --> 00:35:41,310

and Data Zone information attached to it.

583

00:35:43,226 --> 00:35:46,364

But, just an important difference,

584

00:35:46,464 --> 00:35:48,131

when we did the join with Glasgow,

585

00:35:48,231 --> 00:35:52,119

we only received one set of values for every Adzuna point

586

00:35:52,219 --> 00:35:55,927

so it also going to be different for every point,

587

00:35:56,077 --> 00:35:57,206

or for most of the points.

588

00:35:58,963 --> 00:36:02,533

Let's find average salary advertised per Data Zone.

589

00:36:02,633 --> 00:36:06,091

And to do this, I'm going to use group by.

590

00:36:06,735 --> 00:36:12,008

So, I'm going to group my dataset by Data Zone name

591

00:36:12,235 --> 00:36:16,257

and I'm going to find the average value of salary min column.

592

00:36:18,848 --> 00:36:20,497

Right. Let's have a look.

593

00:36:21,100 --> 00:36:23,256

So, our data would look like this.

594

00:36:23,356 --> 00:36:28,174

They would have Data Zone name and information on average salary min.

595

00:36:31,145 --> 00:36:33,036

On this step, I'm going to rename my columns

596

00:36:33,136 --> 00:36:36,304

because it's not really salary minimum anymore.

597

00:36:36,404 --> 00:36:38,702

It's average salary - mean.

598

00:36:39,132 --> 00:36:40,582

So, I'm going to rename the column,

599

00:36:41,832 --> 00:36:43,101

just to make it clear.

600

00:36:44,398 --> 00:36:48,507

And now, let's join our GeoDataFrame Data Zone

601

00:36:48,607 --> 00:36:53,855

with this statistical information of salaries.

602

00:36:54,111 --> 00:36:57,839

So, I'm going to, as I just did, based on the name.

603

00:36:59,130 --> 00:37:00,348

Let's have a look at the data.

604

00:37:00,639 --> 00:37:04,838

So, now we do have the name of the Data Zone,

605

00:37:05,266 --> 00:37:09,927

average salary, and the information of the Data Zone.

606

00:37:10,902 --> 00:37:13,043

Now here it does contain a geometry column.

607

00:37:14,422 --> 00:37:18,365

Let's check the type of salaries by Data Zone.

608

00:37:19,687 --> 00:37:24,017

And Pandas says, actually, it's a data frame,

609

00:37:24,117 --> 00:37:25,969

despite having a geometry column.

610

00:37:26,069 --> 00:37:29,091

And it is recognised as a data frame.

611

00:37:29,520 --> 00:37:31,873

So, let's plot it to verify that.

612

00:37:33,020 --> 00:37:37,209

And, you see, it's not geographical representation

613

00:37:37,309 --> 00:37:40,308

or a map we see with some plots.

614

00:37:41,626 --> 00:37:47,599

Pandas clearly recognised our salaries dz as a data frame.

615

00:37:47,699 --> 00:37:49,407

So, to do that, we need to

616

00:37:50,765 --> 00:37:55,375

say that or create a new GeoDataFrame salaries dz

617

00:37:55,796 --> 00:38:00,407

saying that our salaries dz data frame we created in the previous step

618

00:38:00,978 --> 00:38:05,179

has got geometry, and this geometry is called geometry.

619

00:38:06,259 --> 00:38:09,659

But pretty much, please use this column geometry

620

00:38:09,759 --> 00:38:11,050

as our geometry.

621

00:38:12,251 --> 00:38:14,421

Now, let's see the columns.

622

00:38:15,432 --> 00:38:17,303

Yeah. So, we still have geometry.

623

00:38:17,403 --> 00:38:18,444

Just to check.

624

00:38:19,060 --> 00:38:22,902

And let's plot our new GeoDataFrame

625

00:38:23,635 --> 00:38:26,516

to check that we do have a map now.

626

00:38:26,616 --> 00:38:28,022

So, we do have a map now

627

00:38:29,066 --> 00:38:31,775

with some empty values.

628

00:38:31,875 --> 00:38:35,631

So, these empty values or empty spaces are

629

00:38:35,731 --> 00:38:41,999

the Data Zones which did not have any Adzuna job listings

630

00:38:42,099 --> 00:38:45,536

so there is no average salary minimum,

631

00:38:46,444 --> 00:38:49,312

sorry, average salary available for those Data Zones.

632

00:38:49,412 --> 00:38:52,161

So, they just disappear from the dataset because

633

00:38:52,261 --> 00:38:53,519

there are no values for them.

634

00:38:55,578 --> 00:38:59,177

Right. Let's do some choropleth mapping.

635

00:39:01,128 --> 00:39:05,446

So, just wanted to say it's not normalised data for analysis,

636

00:39:05,765 --> 00:39:08,903

just an example. So, for our plot,

637

00:39:09,003 --> 00:39:12,160

we are going to use the salary mean column

638

00:39:12,260 --> 00:39:13,799

Colourmap is yellow, green, blue,

639

00:39:13,899 --> 00:39:16,948

but you can use anything else.

640

00:39:17,144 --> 00:39:22,100

Figure size is set up on this line.

641

00:39:22,300 --> 00:39:24,978

And the scheme I'm using is Quantiles.

642

00:39:27,088 --> 00:39:29,696

So, this is our distribution of

643

00:39:29,796 --> 00:39:34,434

average salaries between Data Zones in Glasgow.

644

00:39:34,534 --> 00:39:38,512

Again, blank spaces are those Data Zones where

645

00:39:38,612 --> 00:39:40,550

no information was available.

646

00:39:41,830 --> 00:39:47,096

And here, I would like to show you a link for Colourmaps available.

647

00:39:47,944 --> 00:39:49,484

Matplotlib.

648

00:39:50,964 --> 00:39:52,114

Hold on. Give me a second.

649

00:39:52,214 --> 00:39:54,733

So, this is how they will look,

650

00:39:55,012 --> 00:39:57,390

what's available, and the names of them

651

00:39:57,490 --> 00:40:00,528

which you could use in your code are just provided on the left-hand side.

652

00:40:04,549 --> 00:40:08,418

Right. Let's do some cartogram

653

00:40:09,155 --> 00:40:10,795

using geoplot.

654

00:40:11,065 --> 00:40:15,189

Just to showcase that it does contain lots of visualisation.

655

00:40:15,931 --> 00:40:17,490

So, I'm going to

656

00:40:19,168 --> 00:40:22,977

convert my salaries Data Zone GeoDataFrame

657

00:40:24,100 --> 00:40:26,294

to WGS84.

658

00:40:27,033 --> 00:40:30,353

The scale of my data would depend on

659

00:40:30,453 --> 00:40:34,460

the information on the values stored in the salary mean column.

660

00:40:36,142 --> 00:40:41,171

I guess the edge colour is the colour of those polygons inside.

661

00:40:41,948 --> 00:40:44,737

That colour also depends on the value.

662

00:40:44,837 --> 00:40:47,504

And the colourmap used here is red.

663

00:40:47,696 --> 00:40:49,755

Let's try to do something.

664

00:40:50,664 --> 00:40:51,782

This should work.

665

00:40:52,151 --> 00:40:54,207

No. Purples.

666

00:40:57,795 --> 00:41:00,145

No. Not much better.

667

00:41:00,505 --> 00:41:03,214

Well, yeah, you can play with it to make it better.

668

00:41:03,314 --> 00:41:05,013

I'm not sure what you see on your screen.

669

00:41:05,113 --> 00:41:07,301

Mine, it's quite pale.

670

00:41:09,164 --> 00:41:10,596

Let's try something else.

671

00:41:13,544 --> 00:41:16,674

I think this one is better. So, you can see that

672

00:41:17,828 --> 00:41:23,446

polygon size and its colour depends on the value in the salary mean column.

673

00:41:25,304 --> 00:41:28,039

Another quite popular type of visualisation

674

00:41:28,139 --> 00:41:31,766

you probably could have seen on the internet.

675

00:41:32,823 --> 00:41:38,977

It just shows the dependency of the true size and of some value,

676

00:41:39,657 --> 00:41:41,743

and the size represented by some value.

677

00:41:42,484 --> 00:41:44,272

Right. And as a last step,

678

00:41:44,469 --> 00:41:47,296

I want to show how to export your data

679

00:41:47,609 --> 00:41:49,418

on your GeoDataFrame outside.

680

00:41:50,103 --> 00:41:54,811

And really, it is just using the command to_file

681

00:41:54,911 --> 00:41:58,520

and setting the destination.

682

00:41:59,383 --> 00:42:02,511

Or if you're exporting it to a database,

683

00:42:02,611 --> 00:42:05,790

you would need to specify the connection details.

684

00:42:06,456 --> 00:42:08,792

Location, connection details, schema,

685

00:42:09,339 --> 00:42:11,389

and, really, what you want to export.

686

00:42:11,589 --> 00:42:15,677

Again, more information is available if you follow this link.

687

00:42:16,638 --> 00:42:20,956

So, I'm going to export my clipped GeoDataFrame.

688

00:42:21,765 --> 00:42:24,903

to_file. By default, it's a shapefile.

689

00:42:25,111 --> 00:42:27,731

And there is an important warning that

690

00:42:27,831 --> 00:42:32,611

shapefile does only support 10 characters in the column names

691

00:42:32,711 --> 00:42:35,500

so all the names would be truncated.

692

00:42:37,434 --> 00:42:39,394

It is known limitation of a shapefile.

693

00:42:39,523 --> 00:42:42,618

So, please bear in mind that

694

00:42:42,718 --> 00:42:47,568

if your GeoDataFrame has a very descriptive column,

695

00:42:47,825 --> 00:42:52,156

it probably would be better to export it into some other format.

696

00:42:59,113 --> 00:43:05,762

I'll probably show you how our data looks.

697

00:43:07,921 --> 00:43:09,950

Okay. Give me a second.

698

00:43:14,258 --> 00:43:16,617

So, what I'll do is, I'm going to

699

00:43:19,047 --> 00:43:21,666

share some other screen.

700

00:43:22,266 --> 00:43:24,225

So, it's going to be QGIS.

701

00:43:25,335 --> 00:43:28,784

A kind of open source traditional GIS software.

702

00:43:29,307 --> 00:43:30,848

And I'm going to open

703

00:43:36,379 --> 00:43:39,387

my exported clipped GeoDataFrame.

704

00:43:39,487 --> 00:43:41,136

So, by just dragging and dropping.

705

00:43:41,886 --> 00:43:47,495

It's similar to what we've seen in our notebook.

706

00:43:48,145 --> 00:43:49,833

I'm going to share

707

00:43:51,682 --> 00:43:57,450

my notebook again.

708

00:43:57,708 --> 00:43:58,963

And this is it.

709

00:43:59,065 --> 00:44:01,574

Thank you very much for your participation.

710

00:44:02,880 --> 00:44:05,368

Please visit our website to find out more about

711

00:44:05,468 --> 00:44:07,095

our upcoming webinars.

712

00:44:08,154 --> 00:44:11,354

And I'm ready for Questions and Answers.

713

00:44:12,072 --> 00:44:13,250

If you have any.

714

00:44:13,350 --> 00:44:14,458

Thank you.

715

00:44:16,386 --> 00:44:18,816

So, I'm going to open chat

716

00:44:19,642 --> 00:44:22,652

and check if there are any questions.

717

00:44:25,880 --> 00:44:29,869

I see that quite a few people are asking where to get the data.

718

00:44:30,768 --> 00:44:32,737

It has been uploaded to GitHub.

719

00:44:33,676 --> 00:44:34,923

It's available for you.

720

00:44:35,023 --> 00:44:39,417

And I'm going to add this presentation, this slideshow presentation,

721

00:44:39,976 --> 00:44:43,915

to what you have on GitHub.

722

00:44:44,431 --> 00:44:47,326

So, I'm going to at it on a later date.

723

00:44:47,664 --> 00:44:52,413

So, it can be accessible for you quite soon.

724

00:44:56,003 --> 00:44:57,032

Thanks, Nadiia.

725

00:44:57,132 --> 00:45:03,371

Yeah, and I am also posting the link that they usually put

726

00:45:04,300 --> 00:45:09,788

for past presentations and resources afterwards.

727

00:45:10,158 --> 00:45:11,974

You wait a few days, I think.

728

00:45:12,074 --> 00:45:13,343

Yeah, it's here.

729

00:45:14,338 --> 00:45:17,700

Okay, yeah. Maxwell has a question

730

00:45:17,800 --> 00:45:21,148

on choropleth maps.

731

00:45:21,248 --> 00:45:23,076

Cool. Choropleth maps, yes.

732

00:45:24,344 --> 00:45:27,309

Can these choropleth maps be made interactive?

733

00:45:27,409 --> 00:45:29,317

Yeah, I would also want to know that.

734

00:45:29,766 --> 00:45:35,465

Yes, but it does require different libraries

735

00:45:35,565 --> 00:45:37,321

and it's, kind of, another complexity.

736

00:45:37,439 --> 00:45:40,368

I did not show anything in this lab

737

00:45:40,586 --> 00:45:42,646

just to keep it simple and introductive.

738

00:45:42,879 --> 00:45:44,399

But yeah, it is possible.

739

00:45:45,939 --> 00:45:47,939

Just a different set of libraries to use.

740

00:45:48,148 --> 00:45:50,305

So, something like Bokeh maybe

741

00:45:50,776 --> 00:45:52,162

or Dash.

742

00:45:53,279 --> 00:45:58,229

So, both of them can display geographical information easily

743

00:45:58,329 --> 00:46:00,534

but do provide some interactivity.

744

00:46:01,784 --> 00:46:04,089

Yeah, and I was wondering,

745

00:46:04,233 --> 00:46:08,741

earlier you mentioned something about

746

00:46:08,948 --> 00:46:11,210

the boundary files.

747

00:46:11,606 --> 00:46:13,846

Pretty early on in the presentation.

748

00:46:13,946 --> 00:46:16,784

Could you also share those links?

749

00:46:17,714 --> 00:46:20,845

Right. I'm going to share my screen again.

750

00:46:21,854 --> 00:46:27,801

So, in the chat, Chau Man has sent you a link to GitHub

751

00:46:28,000 --> 00:46:33,069

where you can find the data, the exercise itself.

752

00:46:33,169 --> 00:46:35,019

So, it's an exercise folder.

753

00:46:36,169 --> 00:46:39,207

So, if you open this GeoPython lab,

754

00:46:39,307 --> 00:46:40,405

you would have...

755

00:46:41,335 --> 00:46:44,103

So, because I have it here,

756

00:46:44,211 --> 00:46:48,060

you would have this notebook with all the links attached.

757

00:46:49,378 --> 00:46:50,875

So, please have a look there.

758

00:46:50,975 --> 00:46:52,273

I'm trying to find my...

759

00:46:52,685 --> 00:46:59,330

This presentation and notebook, I'll update on GitHub later.

760

00:46:59,430 --> 00:47:03,615

And it is pretty much the same as what you have received,

761

00:47:03,777 --> 00:47:06,260

with very tiny modifications and additions.

762

00:47:07,930 --> 00:47:12,684

So, please see GitHub and the notebook, if you scroll down.

763

00:47:13,523 --> 00:47:15,931

you will have all the links,

764

00:47:17,010 --> 00:47:18,160

all the datasets.

765

00:47:18,260 --> 00:47:19,448

So, it's BoundaryLine.

766

00:47:20,488 --> 00:47:22,317

I think this is the one mentioned.

767

00:47:23,175 --> 00:47:27,634

As well, I can send those in the chat.

768

00:47:29,393 --> 00:47:30,422

I need to stop sharing.

769

00:47:30,522 --> 00:47:31,551

This is it.

770

00:47:33,170 --> 00:47:37,488

And there are many open datasets available on the Ordnance Survey website,

771

00:47:37,588 --> 00:47:44,136

something like Open Roads, Code-Point,

772

00:47:45,484 --> 00:47:48,881

and Zoomstack, which are fantastic for visualisation

773

00:47:49,735 --> 00:47:54,355

and to work with for geospatial analysis.

774

00:47:54,913 --> 00:47:57,903

They're not very detailed in some respects

775

00:47:58,003 --> 00:48:02,471

but may be useful as datasets to use

776

00:48:03,052 --> 00:48:05,480

when you're learning new libraries or a tool.

777

00:48:06,119 --> 00:48:07,709

Are there any more questions?

778

00:48:08,515 --> 00:48:10,874

Does the projection used tend to vary by country?

779

00:48:12,933 --> 00:48:15,191

Well... Right.

780

00:48:15,291 --> 00:48:17,389

The question is, does the projection used

781

00:48:17,489 --> 00:48:19,188

tend to vary by country?

782

00:48:19,307 --> 00:48:20,375

Well, yes, it does.

783

00:48:20,475 --> 00:48:24,694

Typically, every country has its own projection

784

00:48:28,502 --> 00:48:32,840

which is selected based on their best fit of the area.

785

00:48:34,692 --> 00:48:39,782

And a different projection which preserve either directions, angle

786

00:48:40,770 --> 00:48:42,227

or areas.

787

00:48:42,327 --> 00:48:48,716

And again, those three would be used depending on what you want

788

00:48:49,016 --> 00:48:52,331

to have less distortion of, let's say.

789

00:48:52,784 --> 00:48:54,592

And the second part of...

790

00:48:54,692 --> 00:48:59,815

So, yes, different countries tend to have different projections.

791

00:48:59,915 --> 00:49:03,581

Or does that tend to be a standard reference?

792

00:49:04,400 --> 00:49:08,000

So, data collected, via

793

00:49:09,160 --> 00:49:16,823

GPS devices, so mobile phones, are provided in WGS84.

794

00:49:18,677 --> 00:49:21,205

Absolutely forgotten what this stands for.

795

00:49:25,575 --> 00:49:28,993

So, it's World Geodetic System.

796

00:49:29,723 --> 00:49:32,441

1984 is the year it was created.

797

00:49:35,467 --> 00:49:39,276

So, this is quite a popular CRS to use

798

00:49:39,376 --> 00:49:40,875

for this type of data.

799

00:49:41,843 --> 00:49:45,873

The type of data which you see, something like Google Maps,

800

00:49:45,973 --> 00:49:47,503

it's called Web Mercator.

801

00:49:47,603 --> 00:49:51,252

It's a slightly different transformation than the plot.

802

00:49:52,062 --> 00:49:53,501

But what I wanted to say.

803

00:49:54,571 --> 00:49:58,629

If your data is projected to some CRS,

804

00:49:58,778 --> 00:50:02,287

it's very easy to reproject it to some other.

805

00:50:02,967 --> 00:50:06,356

If you know where your data is and what projection it is,

806

00:50:06,456 --> 00:50:09,062

you can easily reproject it to something else.

807

00:50:10,472 --> 00:50:15,641

So, for Britain or, well, government

808

00:50:15,869 --> 00:50:18,417

datasets are provided in the British National Grid.

809

00:50:18,517 --> 00:50:19,875

But again, if you're new to it,

810

00:50:19,975 --> 00:50:21,781

if you show information from the internet,

811

00:50:21,881 --> 00:50:23,636

you can easily reproject them,

812

00:50:23,736 --> 00:50:28,436

those data to WGS84 or the other way round to British National Grid.

813

00:50:29,017 --> 00:50:30,866

I hope I explained it well.

814

00:50:30,966 --> 00:50:34,224

It's quite a complex topic so I would really suggest you

815

00:50:34,802 --> 00:50:36,662

scroll through the links I've provided

816

00:50:36,762 --> 00:50:40,781

and do additional reading to understand that better.

817

00:50:43,441 --> 00:50:44,619

It's quite a complex topic.

818

00:50:44,719 --> 00:50:45,757

Thank you.

819

00:50:46,055 --> 00:50:47,413

So, as I mentioned before,

820

00:50:48,534 --> 00:50:52,341

there's a video recording of this webinar

821

00:50:52,640 --> 00:50:58,368

which will be available in an accessible format at a later date.

822

00:50:58,468 --> 00:51:03,912

So, you can view it. And my presentation/notebook will be

823

00:51:04,012 --> 00:51:06,049

uploaded to GitHub as well.

824

00:51:07,309 --> 00:51:09,147

So, I think this is it.

825

00:51:09,247 --> 00:51:11,964

Thank you everybody for joining. Thank you for all these questions.

826

00:51:13,552 --> 00:51:15,231

If you're interested in the data,

827

00:51:15,331 --> 00:51:21,590

you can contact UBDC or register with Adzuna for the API.

828

00:51:22,039 --> 00:51:24,456

And I think in a few weeks' time

829

00:51:24,556 --> 00:51:28,013

there's going to be a call for interest

830

00:51:28,113 --> 00:51:30,242

in the licensed Adzuna dataset.

831

00:51:30,629 --> 00:51:31,757

Thank you everybody.