Deena 20104016

Basic Analysis using Numpy and Pandas

Import Libraries

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Importing Dataset

In [2]:
```
df=pd.read_csv("8_BreastCancerPrediction.csv")
df
```

Out[2]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_
0	842302	M	17.99	10.38	122.80	1001.0	0.
1	842517	M	20.57	17.77	132.90	1326.0	0.(
2	84300903	M	19.69	21.25	130.00	1203.0	0.1
3	84348301	M	11.42	20.38	77.58	386.1	0.1
4	84358402	M	20.29	14.34	135.10	1297.0	0.1
...	...	...	...	...	...	...	
564	926424	M	21.56	22.39	142.00	1479.0	0.
565	926682	M	20.13	28.25	131.20	1261.0	0.(
566	926954	M	16.60	28.08	108.30	858.1	0.(
567	927241	M	20.60	29.33	140.10	1265.0	0.
568	92751	B	7.76	24.54	47.92	181.0	0.(

569 rows × 33 columns

To display first 10 rows

In [3]:
```
df.head(10)
```

Out[3]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_me
0	842302	M	17.99	10.38	122.80	1001.0	0.118
1	842517	M	20.57	17.77	132.90	1326.0	0.084
2	84300903	M	19.69	21.25	130.00	1203.0	0.109
3	84348301	M	11.42	20.38	77.58	386.1	0.142
4	84358402	M	20.29	14.34	135.10	1297.0	0.100
5	843786	M	12.45	15.70	82.57	477.1	0.127
6	844359	M	18.25	19.98	119.60	1040.0	0.094
7	84458202	M	13.71	20.83	90.20	577.9	0.118
8	844981	M	13.00	21.82	87.50	519.8	0.127
9	84501001	M	12.46	24.04	83.97	475.9	0.118

10 rows × 33 columns

To display last 5 rows

In [4]:
```
df.tail(5)
```

Out[4]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_me
564	926424	M	21.56	22.39	142.00	1479.0	0.111
565	926682	M	20.13	28.25	131.20	1261.0	0.097
566	926954	M	16.60	28.08	108.30	858.1	0.084
567	927241	M	20.60	29.33	140.10	1265.0	0.117
568	92751	B	7.76	24.54	47.92	181.0	0.052

5 rows × 33 columns

Satistical Summary

In [5]:
```
df.describe()
```

Out[5]:

	id	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mea
count	5.690000e+02	569.000000	569.000000	569.000000	569.000000	569.00000
mean	3.037183e+07	14.127292	19.289649	91.969033	654.889104	0.09636
std	1.250206e+08	3.524049	4.301036	24.298981	351.914129	0.01406
min	8.670000e+03	6.981000	9.710000	43.790000	143.500000	0.05263
25%	8.692180e+05	11.700000	16.170000	75.170000	420.300000	0.08637
50%	9.060240e+05	13.370000	18.840000	86.240000	551.100000	0.09587
75%	8.813129e+06	15.780000	21.800000	104.100000	782.700000	0.10530
max	9.113205e+08	28.110000	39.280000	188.500000	2501.000000	0.16340

8 rows × 32 columns

To find shape and size

In [6]:
```
df.shape
```

Out[6]: (569, 33)

In [7]:
```
df.size
```

Out[7]: 18777

To fill the null values

In [8]:
```
df.isna()
```

Out[8]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mea
0	False	False	False	False	False	False	Fals
1	False	False	False	False	False	False	Fals
2	False	False	False	False	False	False	Fals
3	False	False	False	False	False	False	Fals
4	False	False	False	False	False	False	Fals
...	...	...	...	...	...	...	.
564	False	False	False	False	False	False	Fals
565	False	False	False	False	False	False	Fals
566	False	False	False	False	False	False	Fals
567	False	False	False	False	False	False	Fals
568	False	False	False	False	False	False	Fals

569 rows × 33 columns

To fill missing values

In [9]:
```
df.dropna()
```

Out[9]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	con

0 rows × 33 columns

coloumns

In [10]:
```
df.columns
```

Out[10]:
```
Index(['id', 'diagnosis', 'radius_mean', 'texture_mean', 'perimeter_mean',
       'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mean',
       'concave points_mean', 'symmetry_mean', 'fractal_dimension_mean',
       'radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_se',
       'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_se',
       'fractal_dimension_se', 'radius_worst', 'texture_worst',
       'perimeter_worst', 'area_worst', 'smoothness_worst',
       'compactness_worst', 'concavity_worst', 'concave points_worst',
       'symmetry_worst', 'fractal_dimension_worst', 'Unnamed: 32'],
      dtype='object')
```

to print a particular coloumn

In [11]:
```
data=df[["radius_mean","texture_mean"]]
data
```

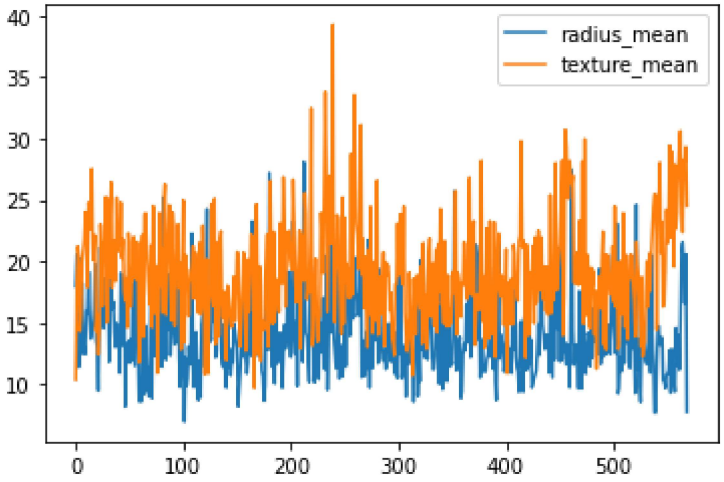
	radius_mean	texture_mean
0	17.99	10.38
1	20.57	17.77
2	19.69	21.25
3	11.42	20.38
4	20.29	14.34
...	...	...
564	21.56	22.39
565	20.13	28.25
566	16.60	28.08
567	20.60	29.33
568	7.76	24.54

569 rows × 2 columns

line plot

In [12]:
```
data.plot.line()
```

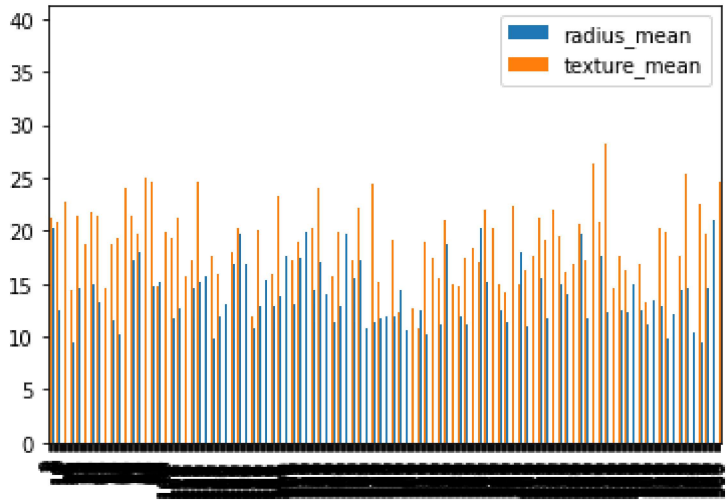
Out[12]: <AxesSubplot:>



bar plot

In [13]:
```
data.plot.bar()
```

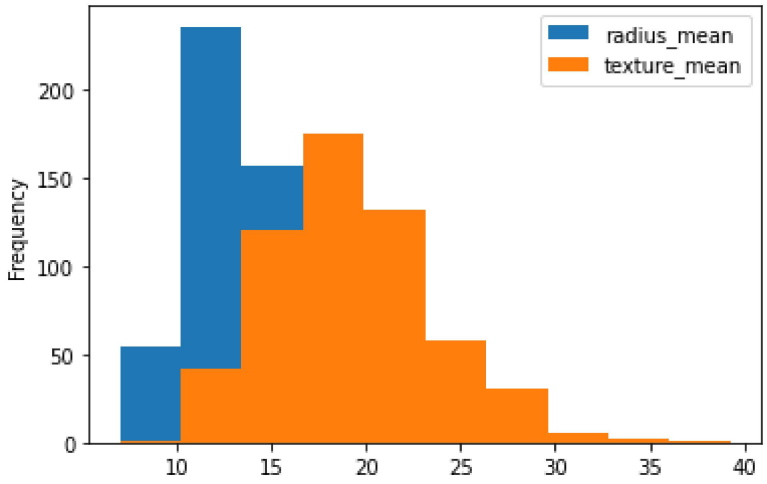
Out[13]: <AxesSubplot:>



hist plot

In [14]:
```
data.plot.hist()
```

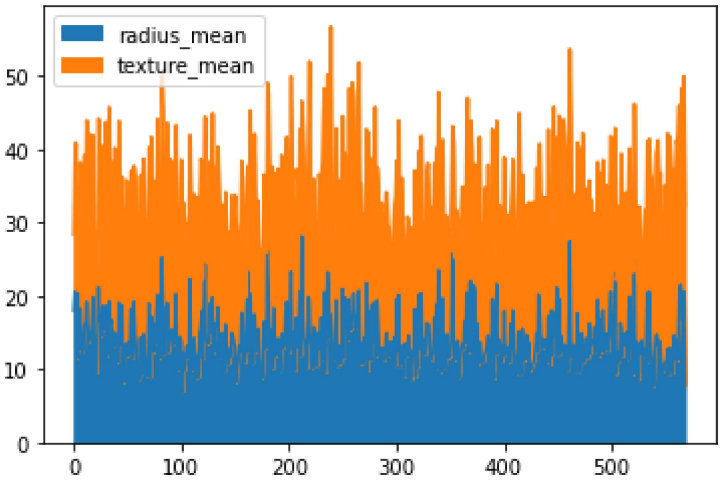
Out[14]: <AxesSubplot:ylabel='Frequency'>



Area plot

In [15]:
```
data.plot.area()
```

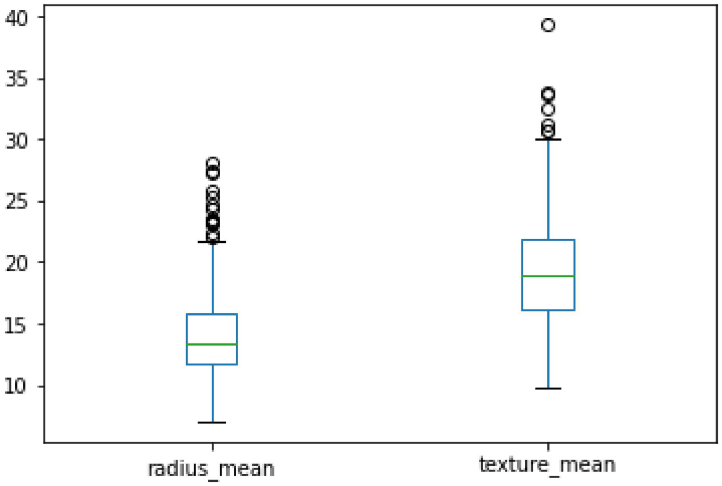
Out[15]: <AxesSubplot:>



Box plot

In [16]:
```
data.plot.box()
```

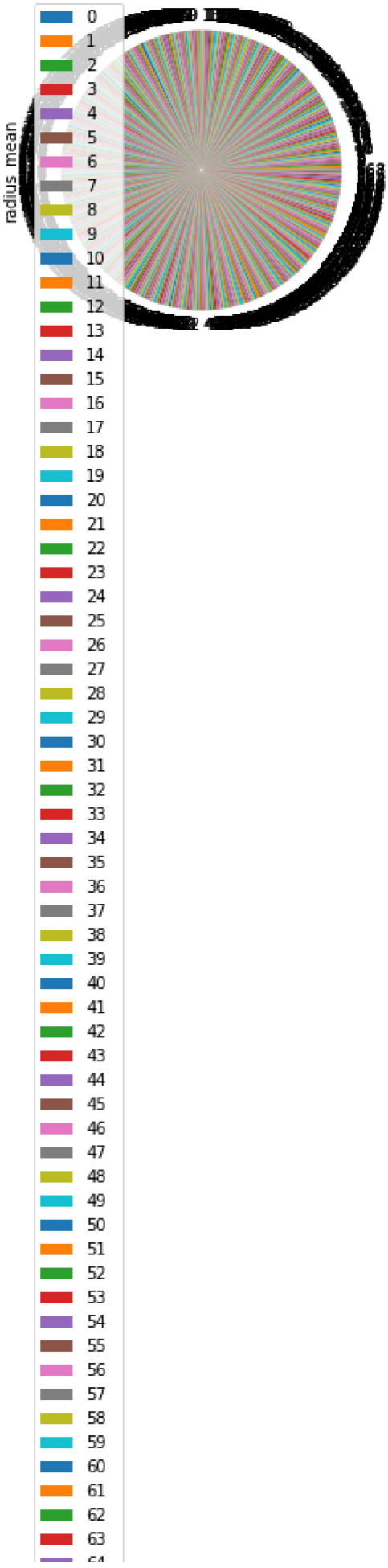
Out[16]: <AxesSubplot:>

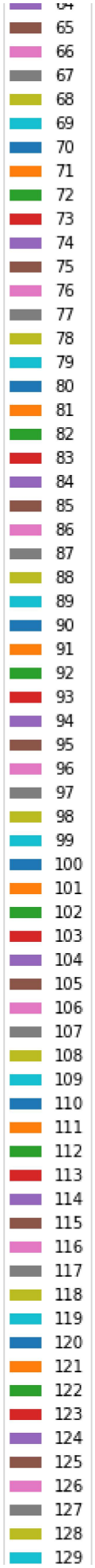


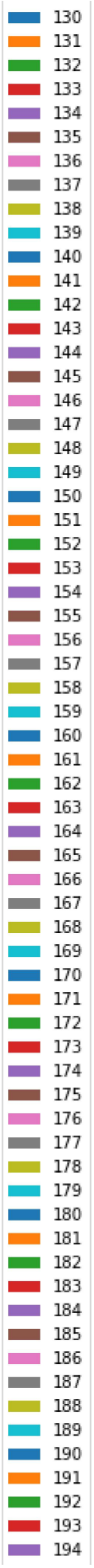
pie plot

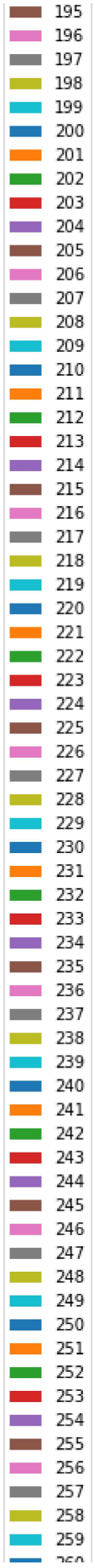
In [17]:
```
data.plot.pie(y="radius_mean")
```

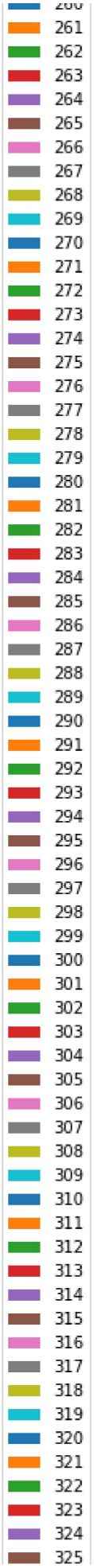
Out[17]:
```
<AxesSubplot:ylabel='radius_mean'>
```

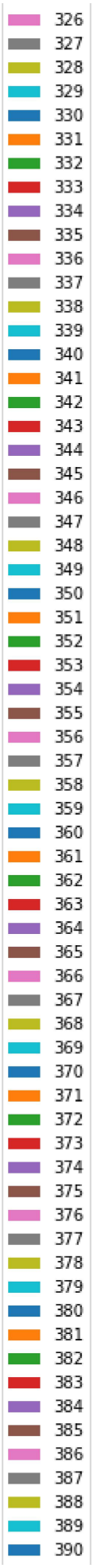

radius_mean
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63

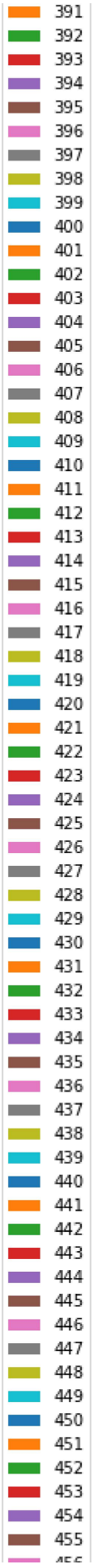

64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129

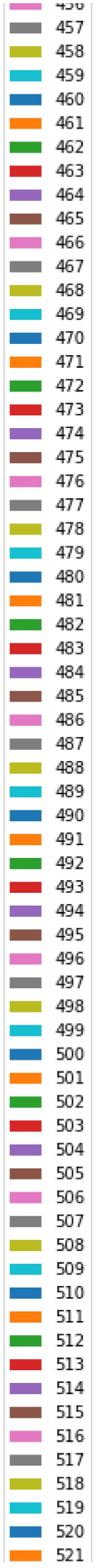

130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194

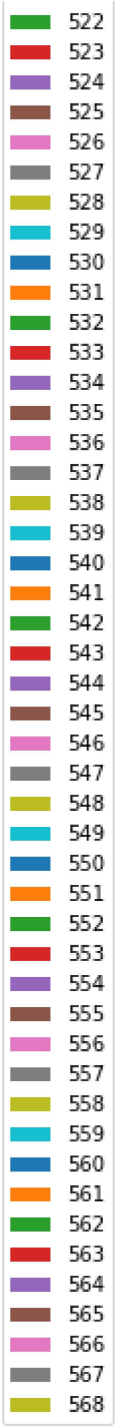

195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259


260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325


326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390


391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455


456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521


522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568

In [18]:
```
data.plot.scatter(x="radius_mean",y="texture_mean")
```

Out[18]:
```
<AxesSubplot:xlabel='radius_mean', ylabel='texture_mean'>
```