

▼ Sahithi Aharam Assignment 2

Inferences are given in print statements

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```



```
print(sns.get_dataset_names())
```

```
['anagrams', 'anscombe', 'attention', 'brain_networks', 'car_crashes', 'diamonds', 'dots', 'dowjones', 'exercise', 'flights', 'fmri', 'geyser', 'glue', 'healthexp', 'iris', 'mpg', 'nba', 'outliers', 'penguins', 'planets', 'seaborn', 'sp500', 'tips', 'volcanoes']
```

```
df=sns.load_dataset('car_crashes')
```

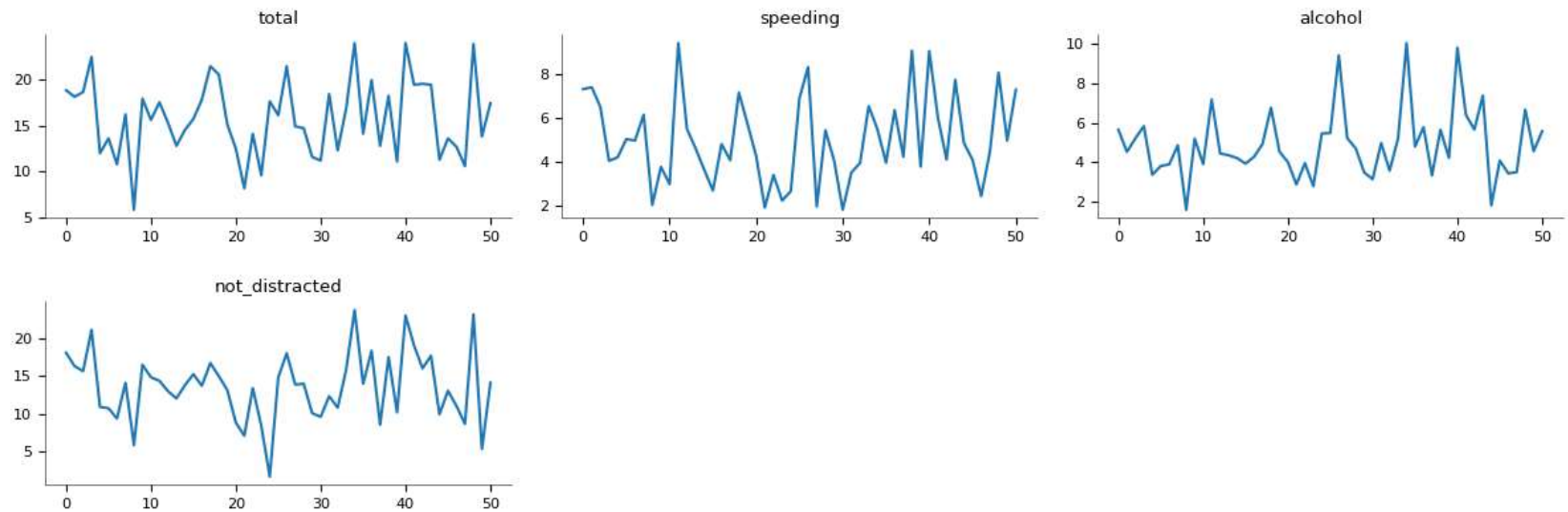
```
df
```



	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses	abbrev	
0	18.8	7.332	5.640	18.048	15.040	784.55	145.08	AL	
1	18.1	7.421	4.525	16.290	17.014	1053.48	133.93	AK	
2	18.6	6.510	5.208	15.624	17.856	899.47	110.35	AZ	
3	22.4	4.032	5.824	21.056	21.280	827.34	142.39	AR	
4	12.0	4.200	3.360	10.920	10.680	878.41	165.63	CA	
5	13.6	5.032	3.808	10.744	12.920	835.50	139.91	CO	
6	10.8	4.968	3.888	9.396	8.856	1068.73	167.02	CT	
7	16.2	6.156	4.860	14.094	16.038	1137.87	151.48	DE	
8	5.9	2.006	1.593	5.900	5.900	1273.89	136.05	DC	
9	17.9	3.759	5.191	16.468	16.826	1160.13	144.18	FL	
10	15.6	2.964	3.900	14.820	14.508	913.15	142.80	GA	
11	17.5	9.450	7.175	14.350	15.225	861.18	120.92	HI	
12	15.3	5.508	4.437	13.005	14.994	641.96	82.75	ID	
13	12.8	4.608	4.352	12.032	12.288	803.11	139.15	IL	
14	14.5	3.625	4.205	13.775	13.775	710.46	108.92	IN	
15	15.7	2.669	3.925	15.229	13.659	649.06	114.47	IA	
16	17.8	4.806	4.272	13.706	15.130	780.45	133.80	KS	
17	21.4	4.066	4.922	16.692	16.264	872.51	137.13	KY	
18	20.5	7.175	6.765	14.965	20.090	1281.55	194.78	LA	
19	15.1	5.738	4.530	13.137	12.684	661.88	96.57	ME	
20	12.5	4.250	4.000	8.875	12.375	1048.78	192.70	MD	
21	8.2	1.886	2.870	7.134	6.560	1011.14	135.63	MA	
22	14.1	3.384	3.948	13.395	10.857	1110.61	152.26	MI	
23	9.6	2.208	2.784	8.448	8.448	777.18	133.35	MN	
24	17.6	2.640	5.456	1.760	17.600	896.07	155.77	MS	
25	16.1	6.923	5.474	14.812	13.524	790.32	144.45	MO	
26	21.4	8.346	9.416	17.976	18.190	816.21	85.15	MT	
27	14.9	1.937	5.215	13.857	13.410	732.28	114.82	NE	
28	14.7	5.439	4.704	13.965	14.553	1029.87	138.71	NV	
29	11.6	4.060	3.480	10.092	9.628	746.54	120.21	NH	
30	11.2	1.792	3.136	9.632	8.736	1301.52	159.85	NJ	
31	18.4	3.496	4.968	12.328	18.032	869.85	120.75	NM	
32	12.3	3.936	3.567	10.824	9.840	1234.31	150.01	NY	

33	16.8	6.552	5.208	15.792	13.608	708.24	127.82	NC
34	23.9	5.497	10.038	23.661	20.554	688.75	109.72	ND
35	14.1	3.948	4.794	13.959	11.562	697.73	133.52	OH
36	19.9	6.368	5.771	18.308	18.706	881.51	178.86	OK
37	12.8	4.224	3.328	8.576	11.520	804.71	104.61	OR
38	18.2	9.100	5.642	17.472	16.016	905.99	153.86	PA
39	11.1	3.774	4.218	10.212	8.769	1148.99	148.58	RI
40	23.9	9.082	9.799	22.944	19.359	858.97	116.29	SC
41	19.4	6.014	6.402	19.012	16.684	669.31	96.87	SD
42	19.5	4.095	5.655	15.990	15.795	767.91	155.57	TN
43	19.4	7.760	7.372	17.654	16.878	1004.75	156.83	TX
44	11.3	4.859	1.808	9.944	10.848	809.38	109.48	UT
45	13.6	4.080	4.080	13.056	12.920	716.20	109.61	VT
46	12.7	2.413	3.429	11.049	11.176	768.95	153.72	VA
47	10.6	4.452	3.498	8.692	9.116	890.03	111.62	WA
48	23.8	8.092	6.664	23.086	20.706	992.61	152.56	WV
49	13.8	4.968	4.554	5.382	11.592	670.31	106.62	WI
50	17.4	7.308	5.568	14.094	15.660	791.14	122.04	WY

Values



Distributions





`sns.__version__`

'0.12.2'

8

`df.info()`

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 51 entries, 0 to 50

Data columns (total 8 columns):

#	Column	Non-Null Count	Dtype
0	total	51 non-null	float64
1	speeding	51 non-null	float64
2	alcohol	51 non-null	float64
3	not_distracted	51 non-null	float64
4	no_previous	51 non-null	float64
5	ins_premium	51 non-null	float64
6	ins_losses	51 non-null	float64
7	abbrev	51 non-null	object

dtypes: float64(7), object(1)

memory usage: 3.3+ KB

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

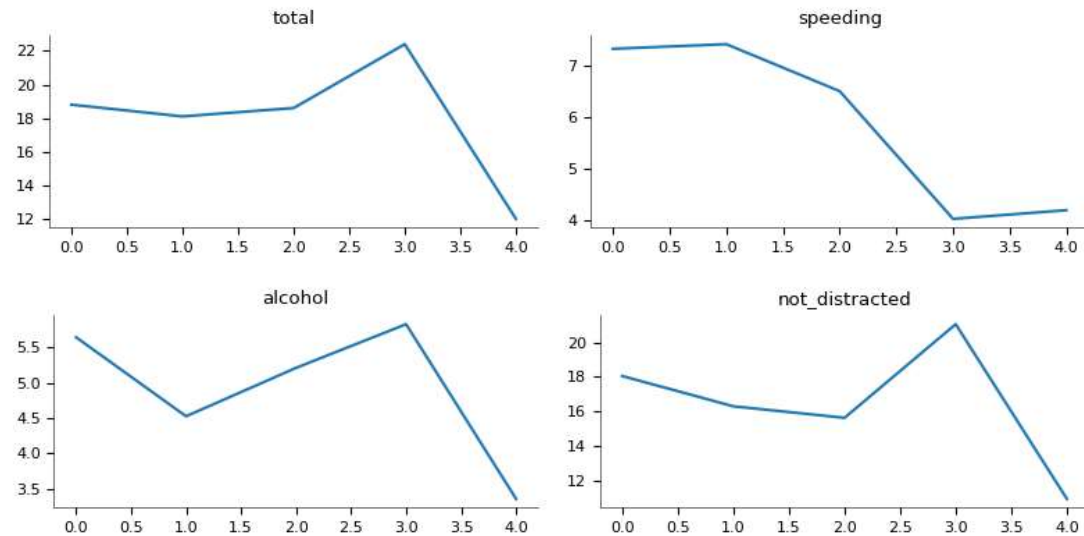
49

50

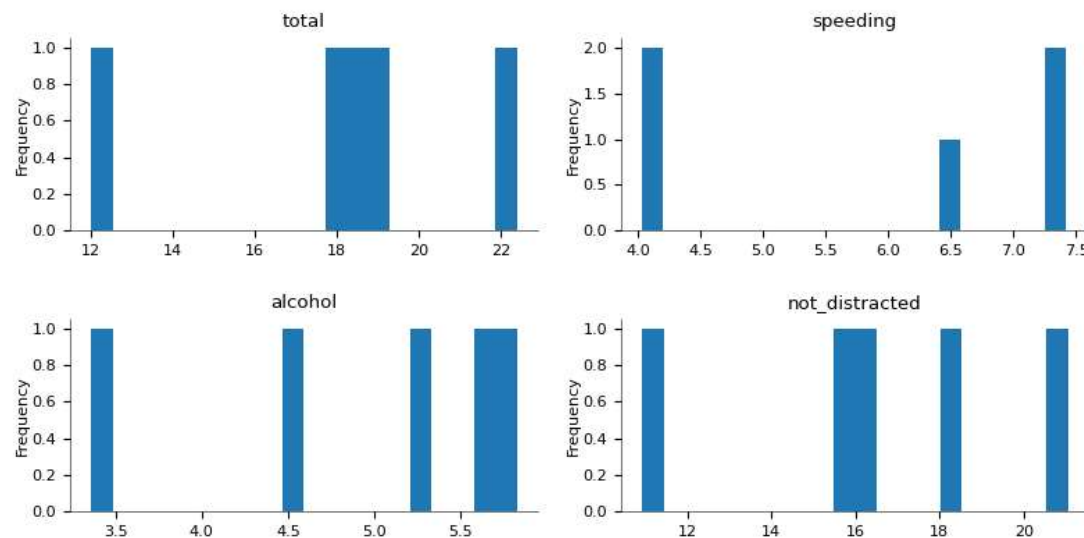
`df.head()`

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses	abbrev	
0	18.8	7.332	5.640	18.048	15.040	784.55	145.08	AL	
1	18.1	7.421	4.525	16.290	17.014	1053.48	133.93	AK	
2	18.6	6.510	5.208	15.624	17.856	899.47	110.35	AZ	
3	22.4	4.032	5.824	21.056	21.280	827.34	142.39	AR	
4	12.0	4.200	3.360	10.920	10.680	878.41	165.63	CA	

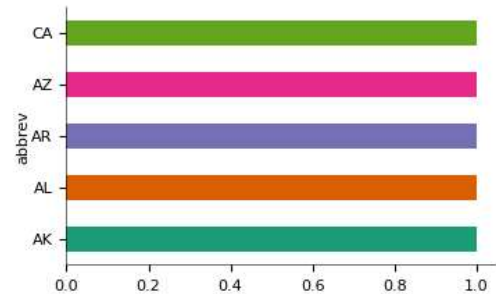
Values



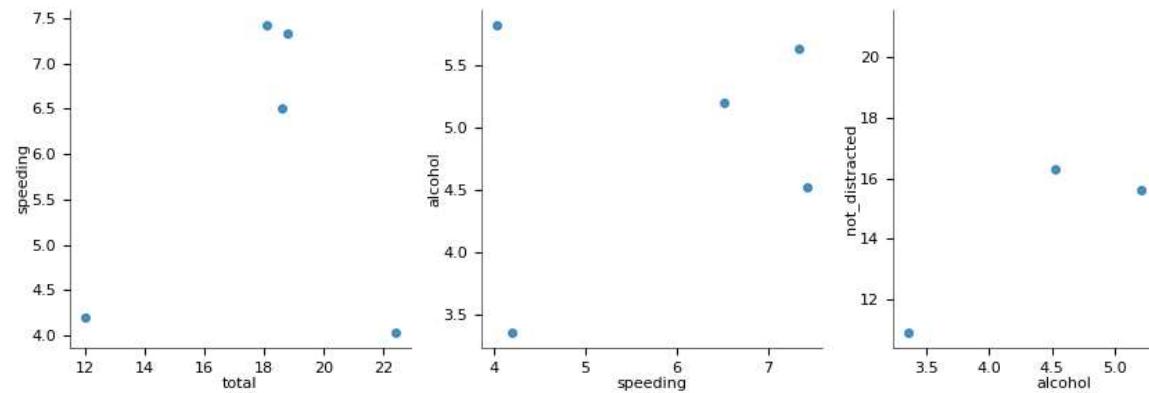
Distributions



Categorical distributions



2-d distributions



Faceted distributions



df.tail

11	120.92	HI
12	82.75	ID
13	139.15	IL
14	108.92	IN
15	114.47	IA
16	133.80	KS
17	137.13	KY
18	194.78	LA
19	96.57	ME
20	192.70	MD
21	135.63	MA
22	152.26	MI
23	133.35	MN
24	155.77	MS
25	144.45	MO
26	85.15	MT
27	114.82	NE
28	138.71	NV
29	120.21	NH
30	159.85	NJ
31	120.75	NM
32	150.01	NY
33	127.82	NC
34	109.72	ND
35	133.52	OH
36	178.86	OK
37	104.61	OR
38	153.86	PA
39	148.58	RI
40	116.29	SC
41	96.87	SD
42	155.57	TN
43	156.83	TX
44	109.48	UT
45	109.61	VT
46	153.72	VA
47	111.62	WA
48	152.56	WV
49	106.62	WI
50	122.04	WY >

```
df.info
```

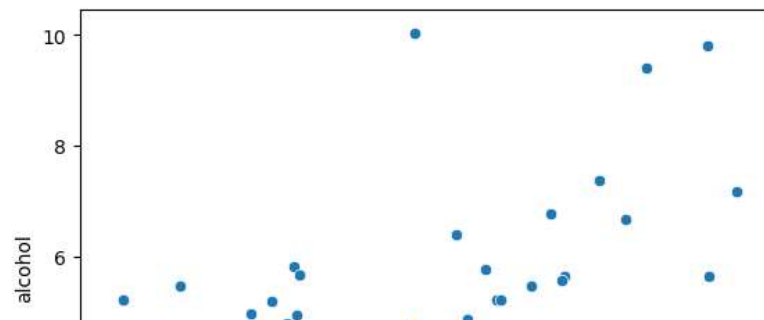
11	120.92	HI
12	82.75	ID
13	139.15	IL
14	108.92	IN
15	114.47	IA
16	133.80	KS
17	137.13	KY
18	194.78	LA
19	96.57	ME
20	192.70	MD
21	135.63	MA
22	152.26	MI
23	133.35	MN
24	155.77	MS
25	144.45	MO
26	85.15	MT
27	114.82	NE
28	138.71	NV
29	120.21	NH
30	159.85	NJ
31	120.75	NM
32	150.01	NY
33	127.82	NC
34	109.72	ND
35	133.52	OH
36	178.86	OK
37	104.61	OR
38	153.86	PA
39	148.58	RI
40	116.29	SC
41	96.87	SD
42	155.57	TN
43	156.83	TX
44	109.48	UT
45	109.61	VT
46	153.72	VA
47	111.62	WA
48	152.56	WV
49	106.62	WI
50	122.04	WY

```
df.shape
```

```
(51, 8)
```

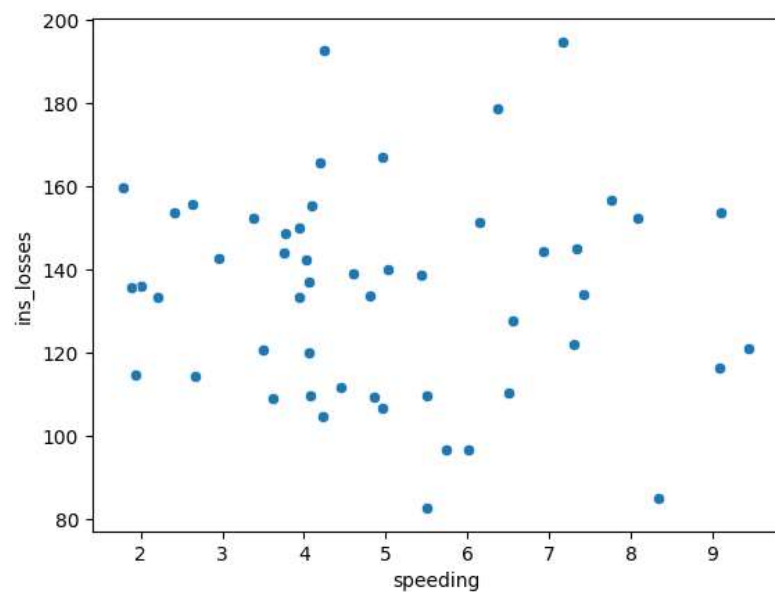
```
sns.scatterplot(x="speeding" , y ="alcohol",data = df)
print("here there no exact linear graph but we can say people with alcohol consumptions has more speed")
```


here there no exact linear graph but we can say people with alcohol consumptions has more speed



```
sns.scatterplot(x="speeding" , y ="ins_losses",data = df)  
print("speeding and ins_losses doesn't seem to have any linear relation")
```

speeding and ins_losses doesn't seem to have any linear relation

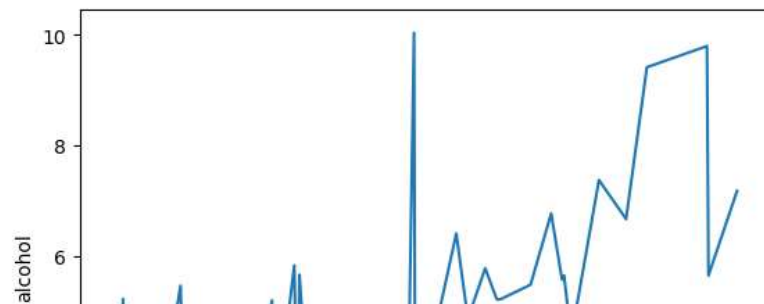


```
sns.lineplot(x="speeding" , y ="alcohol",data = df,ci= None)
```

```
<ipython-input-16-a36f8d625e77>:1: FutureWarning:
```

The `ci` parameter is deprecated. Use `errorbar=None` for the same effect.

```
sns.lineplot(x="speeding", y="alcohol", data=df, ci=None)
<Axes: xlabel='speeding', ylabel='alcohol'>
```



```
sns.distplot(df["speeding"])
print("data distribution of a speeding against the density distribution")
```

```
<ipython-input-17-b279b67b5860>:1: UserWarning:
```

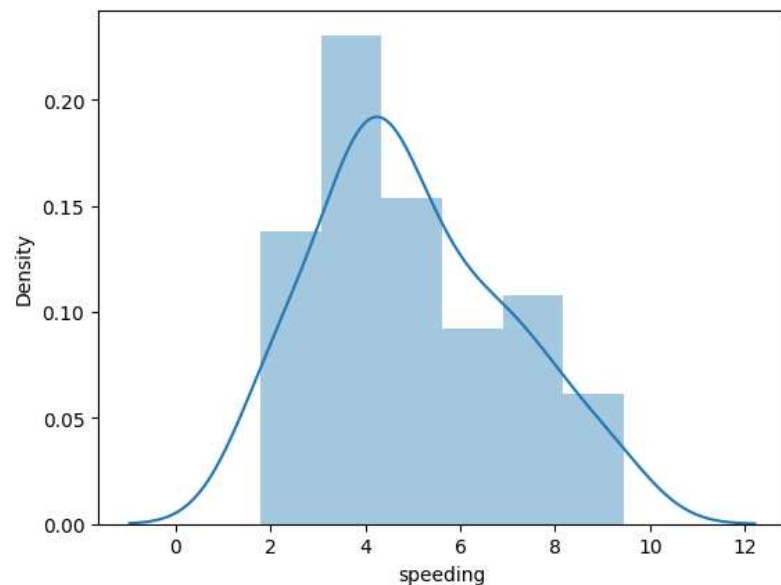
`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see

<https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751>

```
sns.distplot(df["speeding"])
data distribution of a speeding against the density distribution
```



```
sns.relplot(x="speeding" , y ="alcohol",data =df,hue = "abbrev")  
print("using hue we differentiated different categories with colors")
```

using hue we differentiated different categories with colors

abbrev
 ● AL
 ● AK
 ● AZ
 ● AR

```
df["abbrev"].value_counts()
print("here we get count of all categories. has everything repeated only once we got count 1 for all")
```

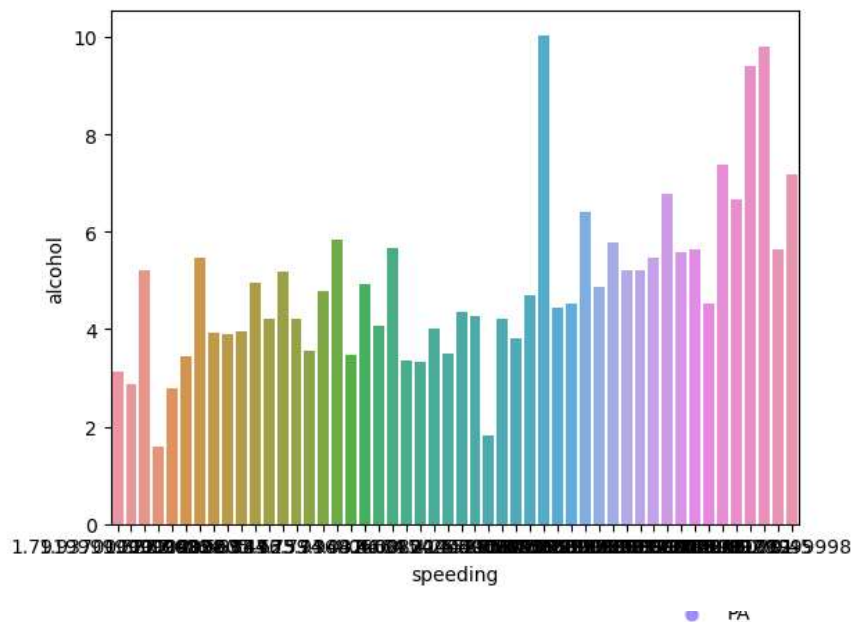
here we get count of all categories. has everything repeated only once we got count 1 for all

```
sns.barplot(data = df,x = "speeding",y = "alcohol",ci = None)
print("bargraph: speeding vs alcohol")
```

<ipython-input-21-81ad8208ff02>:1: FutureWarning:

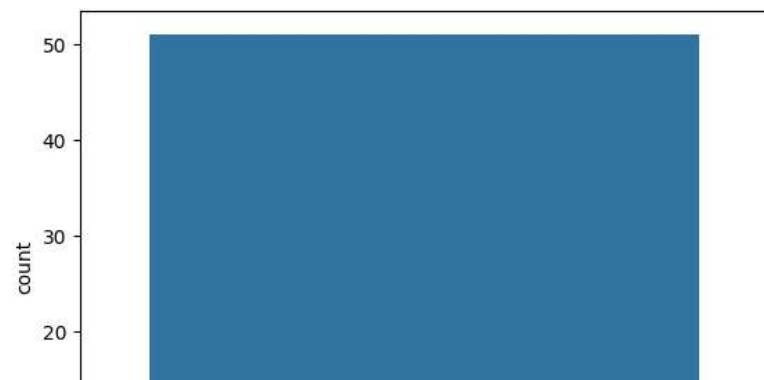
The `ci` parameter is deprecated. Use `errorbar=None` for the same effect.

```
sns.barplot(data = df,x = "speeding",y = "alcohol",ci = None)
bargraph: speeding vs alcohol
```



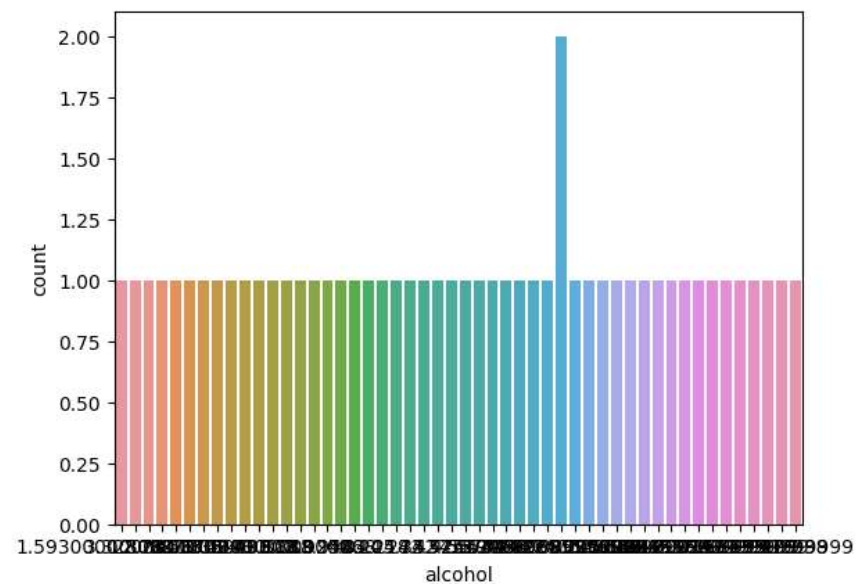
```
sns.countplot(df["speeding"])
```

<Axes: ylabel='count'>



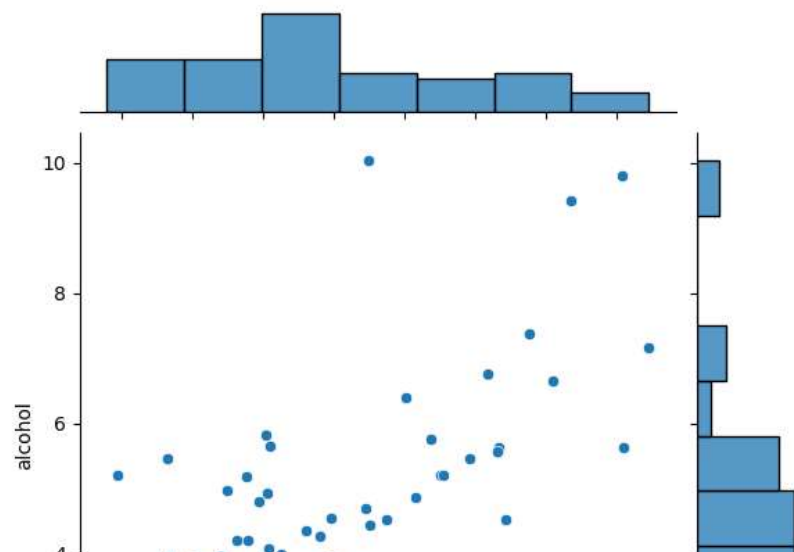
```
sns.countplot(data= df, x = "alcohol")
```

<Axes: xlabel='alcohol', ylabel='count'>



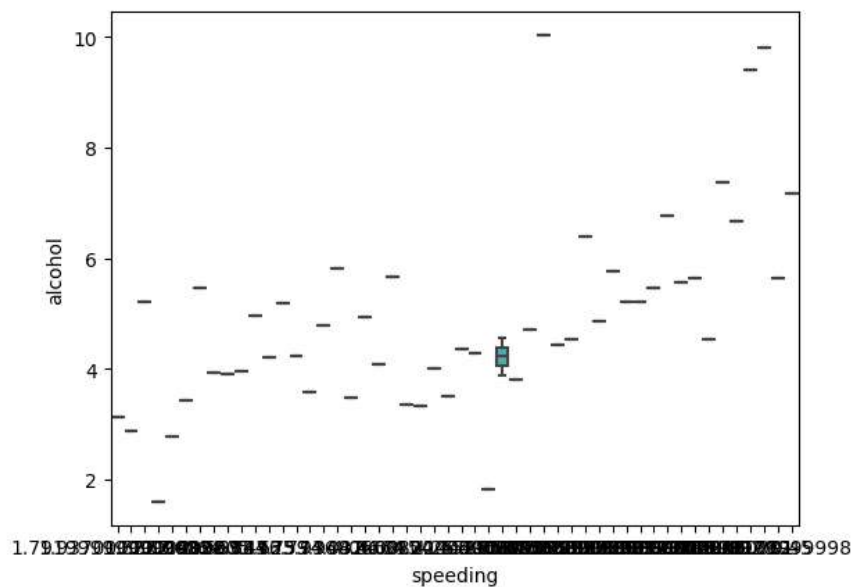
```
sns.jointplot(x="speeding" , y ="alcohol",data =df)
```

```
<seaborn.axisgrid.JointGrid at 0x788e0e3b71c0>
```



```
sns.boxplot(x="speeding" , y ="alcohol",data =df)
```

```
<Axes: xlabel='speeding', ylabel='alcohol'>
```



```
corr= df.corr()  
corr
```

```
<ipython-input-26-a6bc14d9e890>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only
corr= df.corr()
```

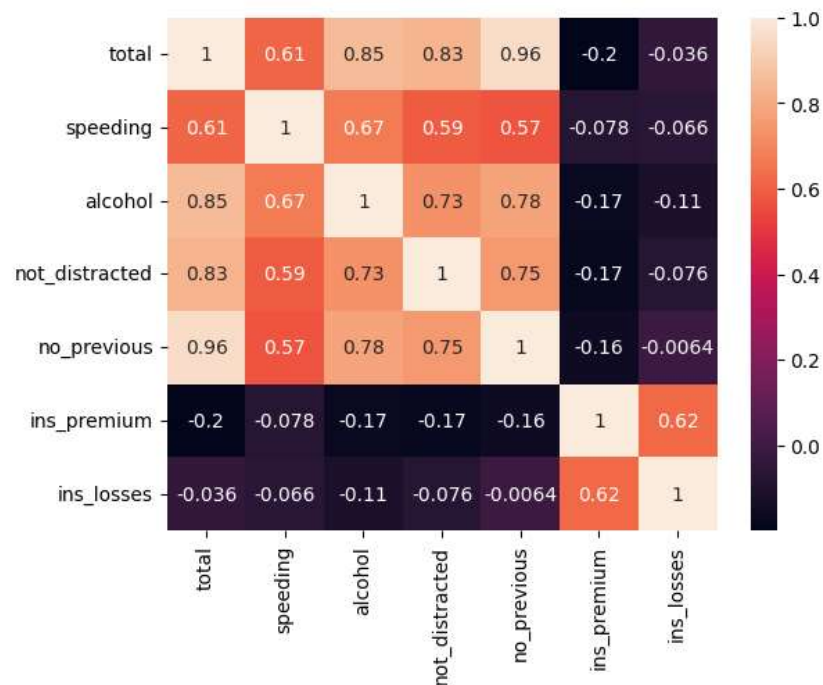
	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses
total	1.000000	0.611548	0.852613	0.827560	0.956179	-0.199702	-0.036011
speeding	0.611548	1.000000	0.669719	0.588010	0.571976	-0.077675	-0.065928
alcohol	0.852613	0.669719	1.000000	0.732816	0.783520	-0.170612	-0.112547
not_distracted	0.827560	0.588010	0.732816	1.000000	0.747307	-0.174856	-0.075970
no_previous	0.956179	0.571976	0.783520	0.747307	1.000000	-0.156895	-0.006359
ins_premium	-0.199702	-0.077675	-0.170612	-0.174856	-0.156895	1.000000	0.623116
ins_losses	-0.036011	-0.065928	-0.112547	-0.075970	-0.006359	0.623116	1.000000

```
sns.heatmap(corr,annot= True)
```

```
print("from this heatmap we could infer that ins_premium and ins_losses are not affecting or leasteffecting our ther columns")
```

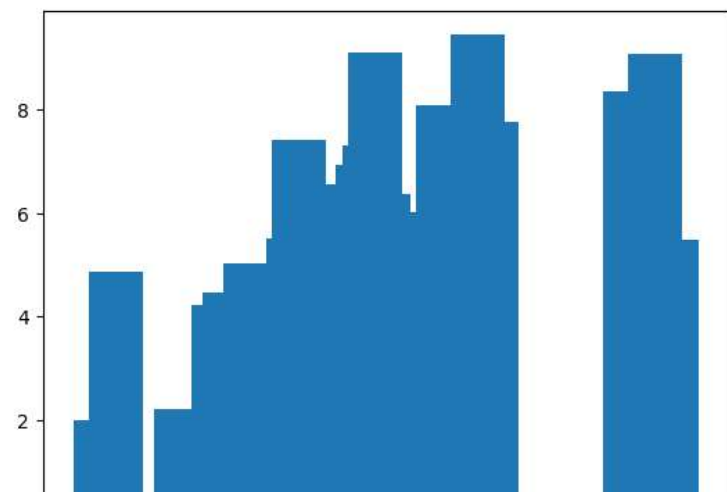
```
print("alcohol is effecting out total")
```

from this heatmap we could infer that ins_premium and ins_losses are not affecting or leasteffecting our ther columns
alcohol is effecting out total



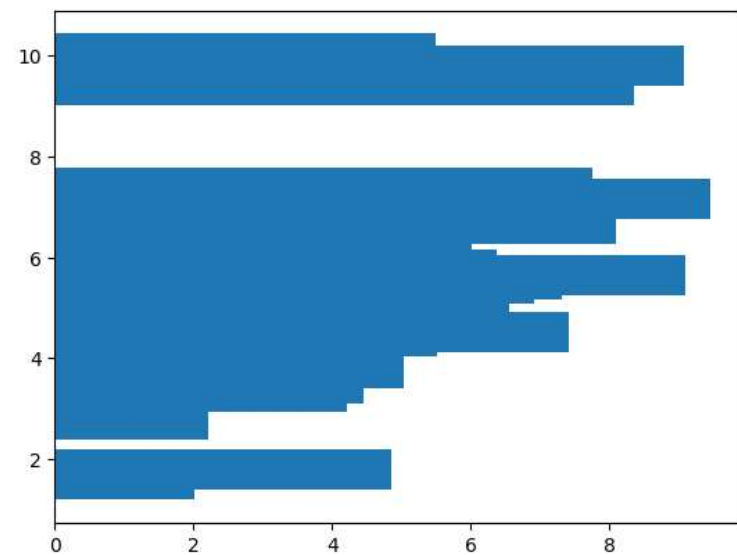
```
x=df["alcohol"]
y = df["speeding"]
plt.bar(x,y)
```

<BarContainer object of 51 artists>



```
plt.barh(x,y)
```

<BarContainer object of 51 artists>



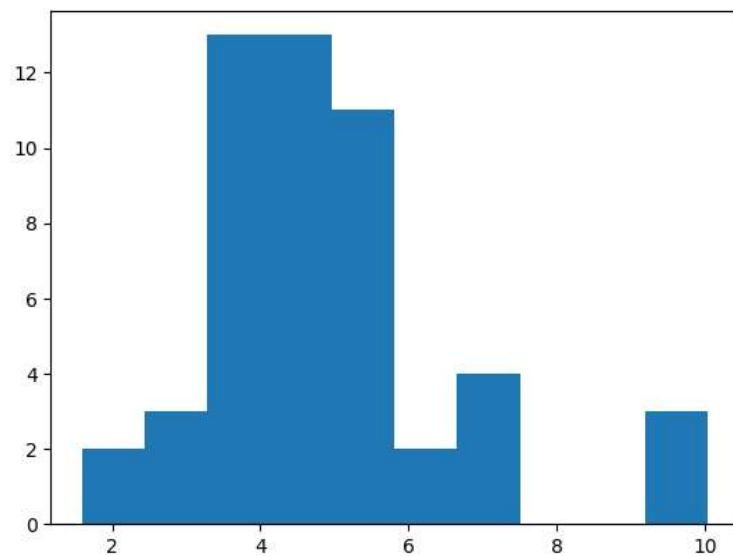
```
plt.barh(x,y,color = 'purple')
```


<BarContainer object of 51 artists>



plt.hist(x)

```
(array([ 2.,  3., 13., 13., 11.,  2.,  4.,  0.,  0.,  3.]),
 array([ 1.593,  2.4375,  3.282,  4.1265,  4.971,  5.8155,  6.66,
        7.5045,  8.349,  9.1935, 10.038 ]),
 <BarContainer object of 10 artists>)
```



```
x1=(df["alcohol"]<5.3)
fig = plt.figure()
axes1 = fig.add_axes([0.1,0.1,0.8,0.8])
axes1.pie(x1,y,autopct="%0.2f%%",colors=["red","green"])
```

```
[(<matplotlib.patches.Wedge at 0x788e09d35060>,
  <matplotlib.patches.Wedge at 0x788e09d352d0>,
  <matplotlib.patches.Wedge at 0x788e09d352a0>,
  <matplotlib.patches.Wedge at 0x788e105f28f0>,
  <matplotlib.patches.Wedge at 0x788e105f0d90>,
  <matplotlib.patches.Wedge at 0x788e105f2e30>,
  <matplotlib.patches.Wedge at 0x788e105f02e0>,
  <matplotlib.patches.Wedge at 0x788e105f3e80>,
  <matplotlib.patches.Wedge at 0x788e105f0490>,
  <matplotlib.patches.Wedge at 0x788e0a534e80>,
  <matplotlib.patches.Wedge at 0x788e0a534490>,
  <matplotlib.patches.Wedge at 0x788e0a5369e0>,
  <matplotlib.patches.Wedge at 0x788e09e64700>,
  <matplotlib.patches.Wedge at 0x788e09e64ca0>,
  <matplotlib.patches.Wedge at 0x788e0a1519f0>,
  <matplotlib.patches.Wedge at 0x788e0a153700>,
  <matplotlib.patches.Wedge at 0x788e09f7d3f0>,
  <matplotlib.patches.Wedge at 0x788e09f7fbb0>,
  <matplotlib.patches.Wedge at 0x788e09f7d480>,
  <matplotlib.patches.Wedge at 0x788e0a130b80>,
  <matplotlib.patches.Wedge at 0x788e09fe60e0>,
  <matplotlib.patches.Wedge at 0x788e09fe70a0>,
  <matplotlib.patches.Wedge at 0x788e114f7940>,
  <matplotlib.patches.Wedge at 0x788e114f5e70>,
  <matplotlib.patches.Wedge at 0x788e114f64d0>,
  <matplotlib.patches.Wedge at 0x788e114f6200>,
  <matplotlib.patches.Wedge at 0x788e09d37370>,
  <matplotlib.patches.Wedge at 0x788e0a1bb9a0>,
  <matplotlib.patches.Wedge at 0x788e0a28ba90>,
  <matplotlib.patches.Wedge at 0x788e09bc5510>,
  <matplotlib.patches.Wedge at 0x788e09bc77f0>,
  <matplotlib.patches.Wedge at 0x788e09bc5060>,
  <matplotlib.patches.Wedge at 0x788e09bc4340>,
  <matplotlib.patches.Wedge at 0x788e09bc7460>,
  <matplotlib.patches.Wedge at 0x788e09bc43a0>,
  <matplotlib.patches.Wedge at 0x788e09bc4580>,
  <matplotlib.patches.Wedge at 0x788e09bc65f0>,
  <matplotlib.patches.Wedge at 0x788e09bc4430>,
  <matplotlib.patches.Wedge at 0x788e09bc5d20>,
  <matplotlib.patches.Wedge at 0x788e09f98400>,
  <matplotlib.patches.Wedge at 0x788e0a0c9510>,
  <matplotlib.patches.Wedge at 0x788e0e4b0400>,
  <matplotlib.patches.Wedge at 0x788e0e4b3490>,
  <matplotlib.patches.Wedge at 0x788e0e4b0760>,
  <matplotlib.patches.Wedge at 0x788e09a252a0>,
  <matplotlib.patches.Wedge at 0x788e09a261d0>,
  <matplotlib.patches.Wedge at 0x788e09a27280>,
  <matplotlib.patches.Wedge at 0x788e09a25ed0>,
  <matplotlib.patches.Wedge at 0x788e09a26650>,
  <matplotlib.patches.Wedge at 0x788e09a266e0>,
  <matplotlib.patches.Wedge at 0x788e09a26860>],
[Text(8.432, 0.0, ''),
 Text(8.486696958750324, 0.7638165554228457, ''),
 Text(7.335757366849085, 2.024540405800627, ''),
 Text(4.804757352505282, 1.803255606825176, ''),
 Text(4.775134994115788, 2.2995838292984265, ''),
 Text(4.9608921968524715, 3.604299184475155, ''),
 Text(4.19336813425361, 4.385919252634093, ''),
 Text(3.9973084640055987, 6.055663551056088, ''),
 Text(1.2207357296943775, 2.8560532695045335, ''),
 Text(1.0812291824532254, 4.737174733426194, ''),
```

```

Text(0.18233063592678578, 4.059907823978586, ''),
Text(-0.473324055350528, 10.539376847737561, ''),
Text(-0.8870134829265006, 6.548196017309395, ''),
Text(-1.7638690610720211, 5.428630576433887, ''),
Text(-2.2390294781847073, 4.160813862193299, ''),
Text(-2.3499331001554014, 2.9467228279554947, ''),
Text(-4.447640130840358, 3.885786055168036, ''),
Text(-4.434746506287771, 2.649637602572927, ''),
Text(-7.455517435917649, 3.590387829013451, ''),
Text(-6.401974082793471, 2.4027009475256595, ''),
Text(-5.264023314814331, 0.9552793000432604, ''),
Text(-2.9859999999999998, -5.241928038327032e-08, ''),
Text(-4.41194026041768, -0.800649198154583, ''),
Text(-3.0970649293392087, -1.1623479786437119, ''),
Text(-3.3696235333985123, -1.6227252519040085, ''),
Text(-7.228473157341247, -3.4810493839641343, ''),
Text(-8.510551843979234, -4.098465970450605, ''),
Text(-2.6071089541361383, -1.5576751590954947, ''),
Text(-4.924334223408194, -4.302261435125813, ''),
Text(-3.2172072865350616, -4.0342505221497715, ''),
Text(-1.3704281118367303, -2.546682310435994, ''),
Text(-1.420241998719717, -4.371055783797848, ''),
Text(-0.6759985990018583, -4.990423017555479, ''),
Text(0.34330588505281284, -7.6442949360479355, ''),
Text(0.8855370316609619, -6.5372955543983995, ''),
Text(1.1232858103764487, -4.921436069706374, ''),
Text(2.3077391134870378, -7.102489998872371, ''),
Text(2.092465408487753, -4.895565801241179, ''),
Text(4.833460618523648, -8.982074284327702, ''),
Text(2.6850720079722414, -4.067709959178988, ''),
Text(6.348373402094667, -7.960607963439539, ''),
Text(4.435506617806075, -5.561949032794036, ''),
Text(3.239029642887625, -4.06161445394504, ''),
Text(5.524119852932503, -6.927026768422148, ''),
Text(4.118042457093326, -4.3071344675522685, ''),
Text(4.190708127072667, -3.0447274744526975, ''),
Text(3.165103682634525, -1.5242334723307212, ''),
Text(5.351921852219658, -1.4770363867331437, ''),
Text(9.044280834378066, -1.6412946685167822, ''),
Text(6.043572034787734, -0.5439311172673039, ''),
Text(8.4079999999999994, 2.952051637391405e-07, '')],
[Text(7.932, 0.0, '0.00%'),
Text(7.988709811775184, 0.7189969007213526, '2.86%'),
Text(6.853775936701314, 1.8915219822920444, '2.86%'),
Text(4.336639917537893, 1.6275681938453264, '0.00%'),
Text(4.324650560708639, 2.0826419586098956, '2.86%'),
Text(4.5563837006968555, 3.3104065569086876, '2.86%'),
Text(3.847836811391579, 4.024521819160618, '2.86%'),
Text(3.721859975581839, 5.638376922675707, '2.86%'),
Text(1.024223216865276, 2.3962893819474607, '2.86%'),
Text(0.9699687191425415, 4.249710776498205, '2.86%'),
Text(0.15989822501059656, 3.5604112905166536, '2.86%'),
Text(-0.45089163566566887, 10.039880314669428, '0.00%'),
Text(-0.8198968452958635, 6.052721137065039, '2.86%'),
Text(-1.6093605588758033, 4.953102319913749, '2.86%'),
Text(-2.0020951418688653, 3.720516098998241, '2.86%'),
Text(-2.038188194324226, 2.5558070906305415, '2.86%'),
Text(-4.0711043933835045, 3.556816697297393, '2.86%'),
Text(-4.005522105756628, 2.393187970113294, '2.86%'),
Text(-7.005032998702081, 3.3734459662331817, '0.00%'),
Text(-5.933856644741886, 2.2270135427636197, '2.86%'),
Text(-4.772058519037291, 0.8660008607868809, '2.86%'),

```

```
Text(-2.4859999999999998, -4.364177194668788e-08, '2.86%'),
Text(-3.9199754677752083, -0.711370741625303, '2.86%'),
Text(-2.6289474974560147, -0.986660557446053, '2.86%'),
Text(-2.9191391037997807, -1.4057833733072158, '0.00%'),
Text(-6.777988727742515, -3.2641075053673414, '0.00%'),
Text(-8.060067414380503, -3.8815240918538123, '0.00%'),
Text(-2.1778845626089502, -1.3012255115657787, '2.86%'),
Text(-4.547798497501465, -3.973292064034988, '2.86%'),
Text(-2.905462394428951, -3.6433347738794444, '2.86%'),
Text(-1.1334937909797576, -2.106384538922164, '2.86%'),
Text(-1.2657335132193126, -3.895527521852913, '2.86%'),
Text(-0.6088819787673609, -4.494948134954656, '2.86%'),
Text(0.3208734565992835, -7.144798403373606, '2.86%'),
Text(0.8184203853322547, -6.0418206753322785, '0.00%'),
Text(1.0120253299508892, -4.433972116684744, '2.86%'),
Text(2.1532306029429136, -6.626961745064633, '0.00%'),
Text(1.8959528795163263, -4.435801920583668, '2.86%'),
Text(4.5965262744783715, -8.541776525292029, '0.00%'),
Text(2.4096235048975347, -3.650423340469612, '2.86%'),
Text(6.036628489400959, -7.569692231587274, '0.00%'),
Text(4.123761705112368, -5.171033300941771, '0.00%'),
Text(2.9272847301939167, -3.6706987220927747, '0.00%'),
Text(5.212374940238795, -6.536111036569881, '0.00%'),
Text(3.772511121542619, -3.9457370462104104, '2.86%'),
Text(3.786199620598471, -2.7508348610885376, '2.86%'),
Text(2.7146192416105395, -1.3072916174587141, '2.86%'),
Text(4.869940417401605, -1.3440179801469456, '2.86%'),
Text(8.55231603703374, -1.5520162378968527, '0.00%'),
Text(5.5455848862389745, -0.4991114800501562, '2.86%'),
Text(7.9079999999999995, 2.776501468659756e-07, '0.00%'))]
```

