

KRISHNA LATHVIK MORNING SESSION

ASSIGNMENT-2

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

```
In [49]: # For Data Preprocessing---
        """
        #Steps:
        #1. Import the necessary libraries
        #2. Import the dataset
        #3. handling null values
        #4. dependent and independent variable seperation
        #5. Encoding
        #6. Split the data into training and testing sets
        #7. Feature scaling
        """
```

```
Out[49]: '\n#Steps:\n#1. Import the necessary libraries\n#2. Import the dataset\n#3. handling
null values\n#4. dependent and independent variable seperation\n#5. Encoding\n#6. Spl
it the data into training and testing sets\n#7. Feature scaling\n'
```

```
In [4]: ak = sns.load_dataset('car_crashes')
```

```
In [5]: ak
```

Out[5]:

	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

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses	abbrev
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

In [6]: `ak.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
```

In []: `total` - Number of drivers involved in fatal collisions per billion miles
`speeding` - Number of drivers Involved In Fatal Collisions Who Were Speeding
`alcohol` - Number of Drivers Involved In Fatal Collisions Who Were Alcohol-Impaired
`not_distracted` - Number of Drivers Involved In Fatal Collisions Who Were Not Distracted
`no_previous` - Number Of Drivers Involved In Fatal Collisions Who Had Not Been Involved
`ins_premium` - Car Insurance Premiums(\$)
`ins_loses` - Losses incurred by insurance companies for collisions per insured driver (

abbrev - represents the states

In [7]: `ak.head()`

Out[7]:

	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

In [8]: `ak.head(2)`

Out[8]:

	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

In [9]: `ak.tail(8)`

Out[9]:

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses	abbrev
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

In [10]: `ak.tail()`

Out[10]:

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses	abbrev
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

In [11]: `ak.shape`

Out[11]: (51, 8)

In [12]: `ak.describe()`

Out[12]:

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses
count	51.000000	51.000000	51.000000	51.000000	51.000000	51.000000	51.000000
mean	15.790196	4.998196	4.886784	13.573176	14.004882	886.957647	134.493137
std	4.122002	2.017747	1.729133	4.508977	3.764672	178.296285	24.835922
min	5.900000	1.792000	1.593000	1.760000	5.900000	641.960000	82.750000
25%	12.750000	3.766500	3.894000	10.478000	11.348000	768.430000	114.645000
50%	15.600000	4.608000	4.554000	13.857000	13.775000	858.970000	136.050000
75%	18.500000	6.439000	5.604000	16.140000	16.755000	1007.945000	151.870000
max	23.900000	9.450000	10.038000	23.661000	21.280000	1301.520000	194.780000

In [13]: `corr = ak.corr()`
`corr`

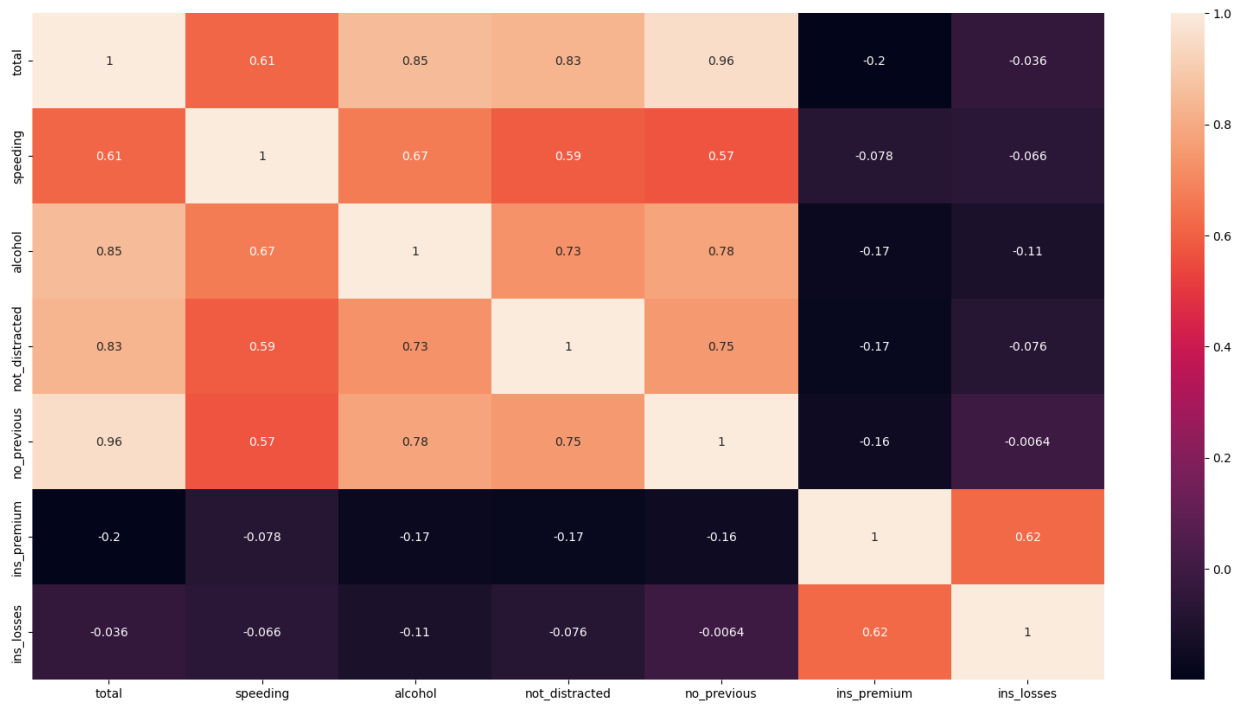
C:\Users\nagka\AppData\Local\Temp\ipykernel_14956\3288373526.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
`corr = ak.corr()`

Out[13]:

	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

In [14]: `plt.subplots(figsize = (20,10))`
`sns.heatmap(corr,annot=True)`

Out[14]: <Axes: >



```
In [15]: ak["total"].value_counts()
```

```
Out[15]: 14.1    2
          12.8    2
          13.6    2
          21.4    2
          19.4    2
          23.9    2
          14.9    1
          14.7    1
          11.6    1
          11.2    1
          18.4    1
          12.3    1
          16.8    1
          19.9    1
          17.6    1
          18.2    1
          11.1    1
          19.5    1
          11.3    1
          12.7    1
          10.6    1
          23.8    1
          13.8    1
          16.1    1
          18.8    1
           9.6    1
          18.1    1
          18.6    1
          22.4    1
          12.0    1
          10.8    1
          16.2    1
           5.9    1
          17.9    1
          15.6    1
          17.5    1
          15.3    1
          14.5    1
          15.7    1
          17.8    1
          20.5    1
          15.1    1
          12.5    1
           8.2    1
          17.4    1
          Name: total, dtype: int64
```

```
In [16]: ak.alcohol.value_counts()
```

```
Out[16]: 5.208      2
          5.640      1
          4.218      1
          4.704      1
          3.480      1
          3.136      1
          4.968      1
          3.567      1
          10.038     1
          4.794      1
          5.771      1
          3.328      1
          5.642      1
          9.799      1
          9.416      1
          6.402      1
          5.655      1
          7.372      1
          1.808      1
          4.080      1
          3.429      1
          3.498      1
          6.664      1
          4.554      1
          5.215      1
          5.474      1
          4.525      1
          5.456      1
          5.824      1
          3.360      1
          3.808      1
          3.888      1
          4.860      1
          1.593      1
          5.191      1
          3.900      1
          7.175      1
          4.437      1
          4.352      1
          4.205      1
          3.925      1
          4.272      1
          4.922      1
          6.765      1
          4.530      1
          4.000      1
          2.870      1
          3.948      1
          2.784      1
          5.568      1
          Name: alcohol, dtype: int64
```

```
In [17]: ak.isnull().any()
```



```
Out[17]: total          False
          speeding      False
          alcohol        False
          not_distracted False
          no_previous     False
          ins_premium     False
          ins_losses      False
          abbrev          False
          dtype: bool
```

```
In [18]: ak.isnull().sum()
```

```
Out[18]: total          0
          speeding      0
          alcohol        0
          not_distracted 0
          no_previous     0
          ins_premium     0
          ins_losses      0
          abbrev          0
          dtype: int64
```

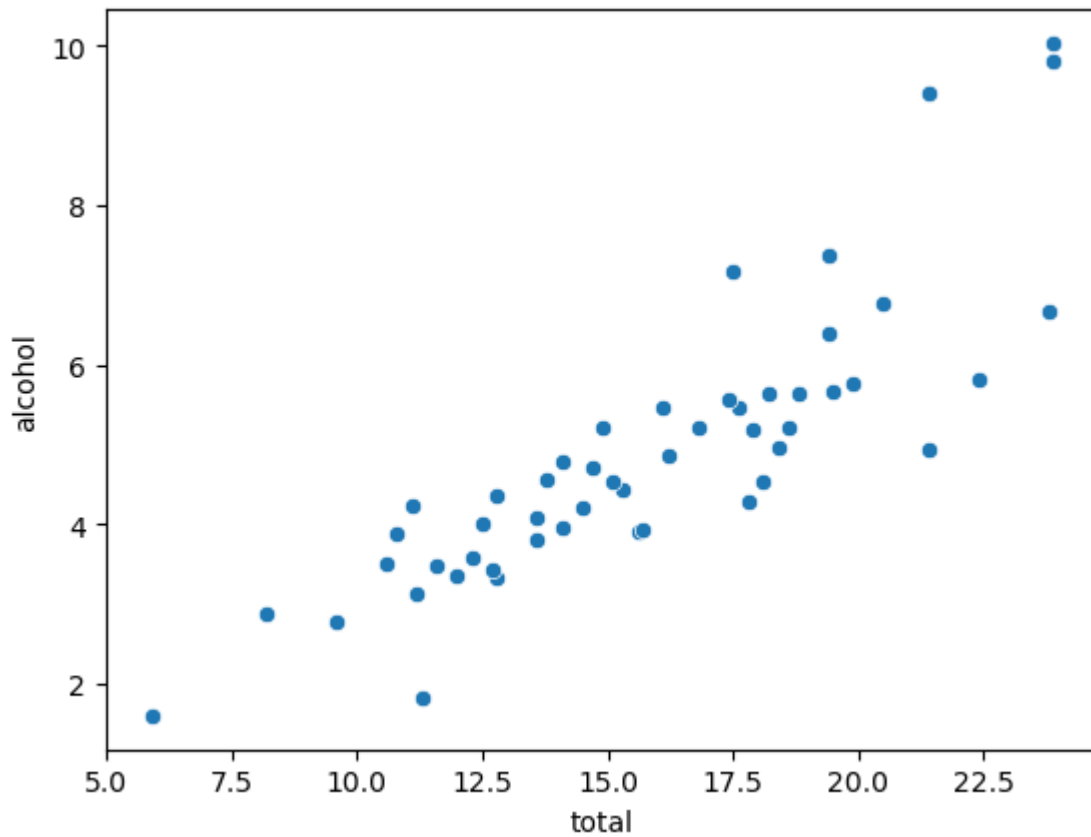
DATA VISUALIZATION

```
In [ ]: total - Number of drivers involved in fatal collisions per billion miles
          speeding - Number of drivers Involved In Fatal Collisions Who Were Speeding
          alcohol - Number of Drivers Involved In Fatal Collisions Who Were Alcohol-Impaired
          not_distracted - Number of Drivers Involved In Fatal Collisions Who Were Not Distracted
          no_previous - Number Of Drivers Involved In Fatal Collisions Who Had Not Been Involved
          ins_premium - Car Insurance Premiums($)

          ins_loses - Losses incurred by insurance companies for collisions per insured driver (
          abbrev - represents the states
```

```
In [19]: sns.scatterplot(x="total",y="alcohol", data=ak)
```

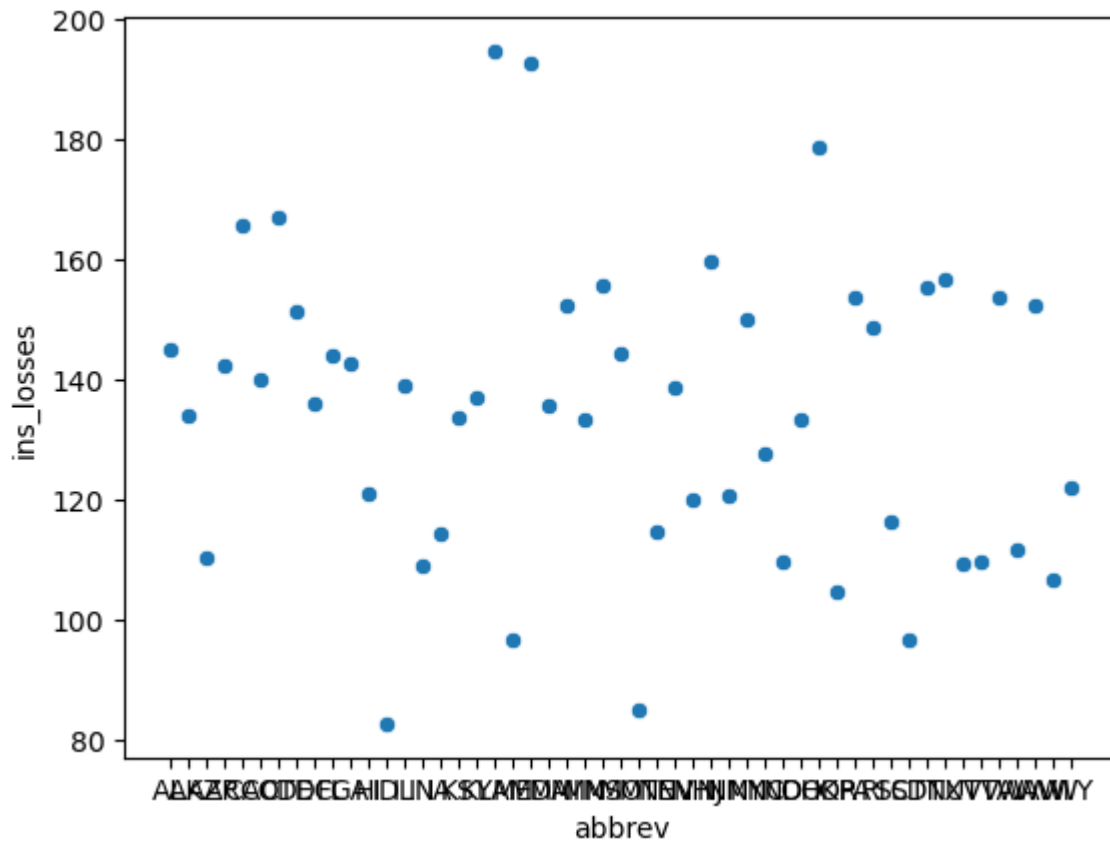
```
Out[19]: <Axes: xlabel='total', ylabel='alcohol'>
```



In []: We can clearly see that Number of drivers involved **in** fatal collisions per billion mil
Number of Drivers Involved In Fatal Collisions Who Were Alcohol-Impaired. By this we c
drinking **and** driving are more prone to getting an accident. That **is** why it **is** recommen
If people drive without impairing **with** alcohol, the numbers of accidents would have be

In [20]: `sns.scatterplot(x="abbrev",y="ins_losses",data=ak)`

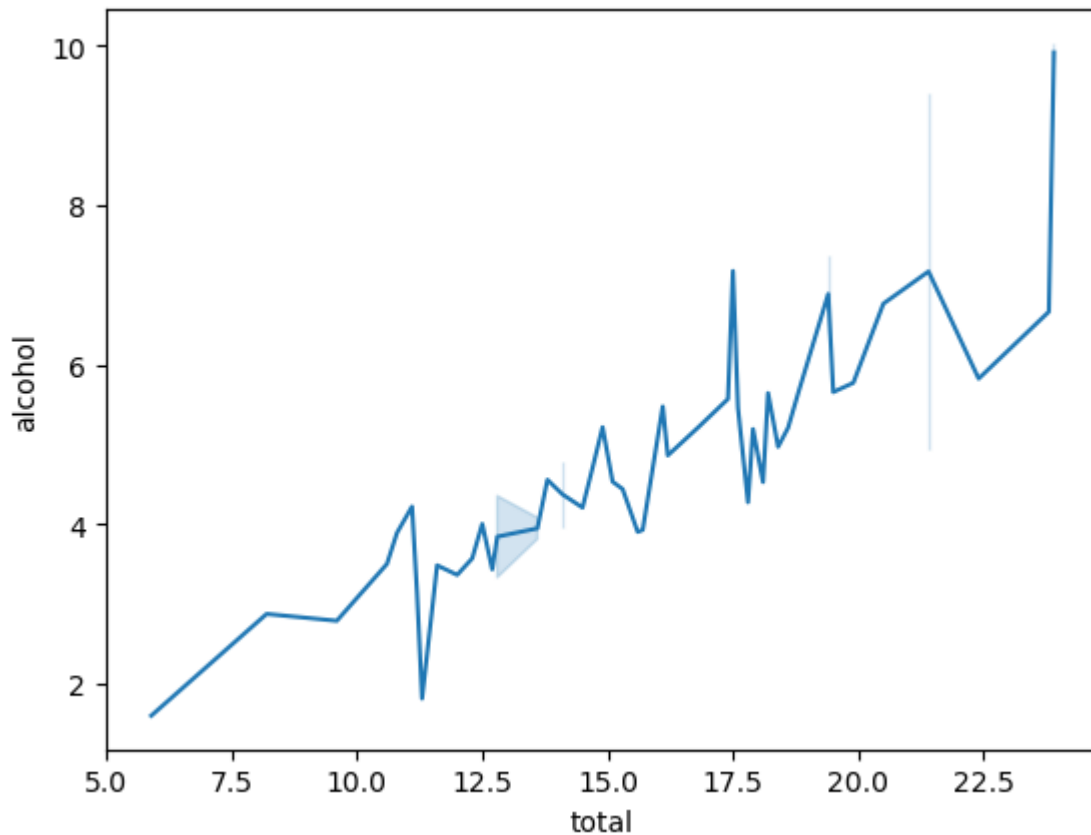
Out[20]: `<Axes: xlabel='abbrev', ylabel='ins_losses'>`



In []: This graph shows the Losses incurred by insurance companies for collisions per insured. For example, In Arizona State, the insurance companies loss is around 110.35 dollars. From the graph, we can see that Idaho state as the lowest insurance losses for the company.

```
In [21]: #Lineplot
sns.lineplot(y="alcohol",x="total",data=ak)
```

Out[21]: <Axes: xlabel='total', ylabel='alcohol'>



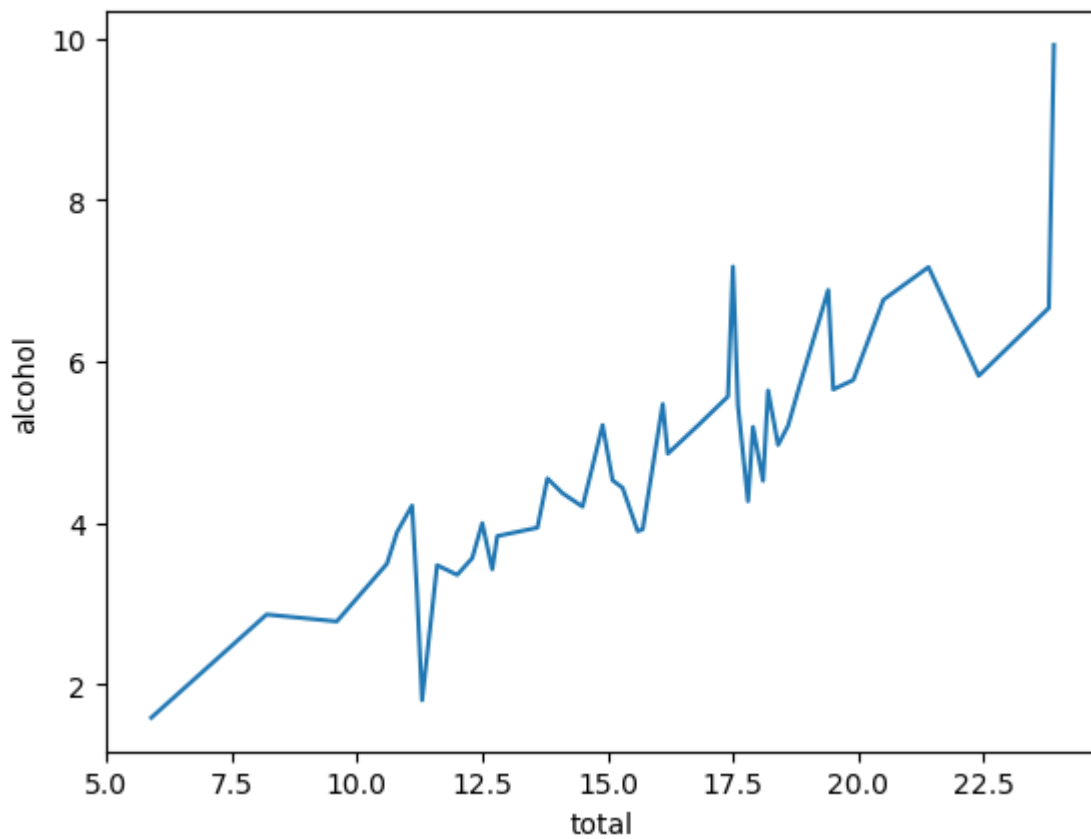
In []: We already discussed that the people who are drinking **and** driving are more prone to an accident. This conclusion is justified. But **for** the same number of total people who faced an accident, the number of people who were drinking alcohol was **not** exactly the same. Some states were affected by this more severely than the others.

```
In [22]: #Lineplot
sns.lineplot(y="alcohol",x="total",data=ak,ci=None)
```

C:\Users\nagka\AppData\Local\Temp\ipykernel_14956\2654429431.py:2: FutureWarning:
The `ci` parameter is deprecated. Use `errorbar=None` for the same effect.

```
sns.lineplot(y="alcohol",x="total",data=ak,ci=None)
```

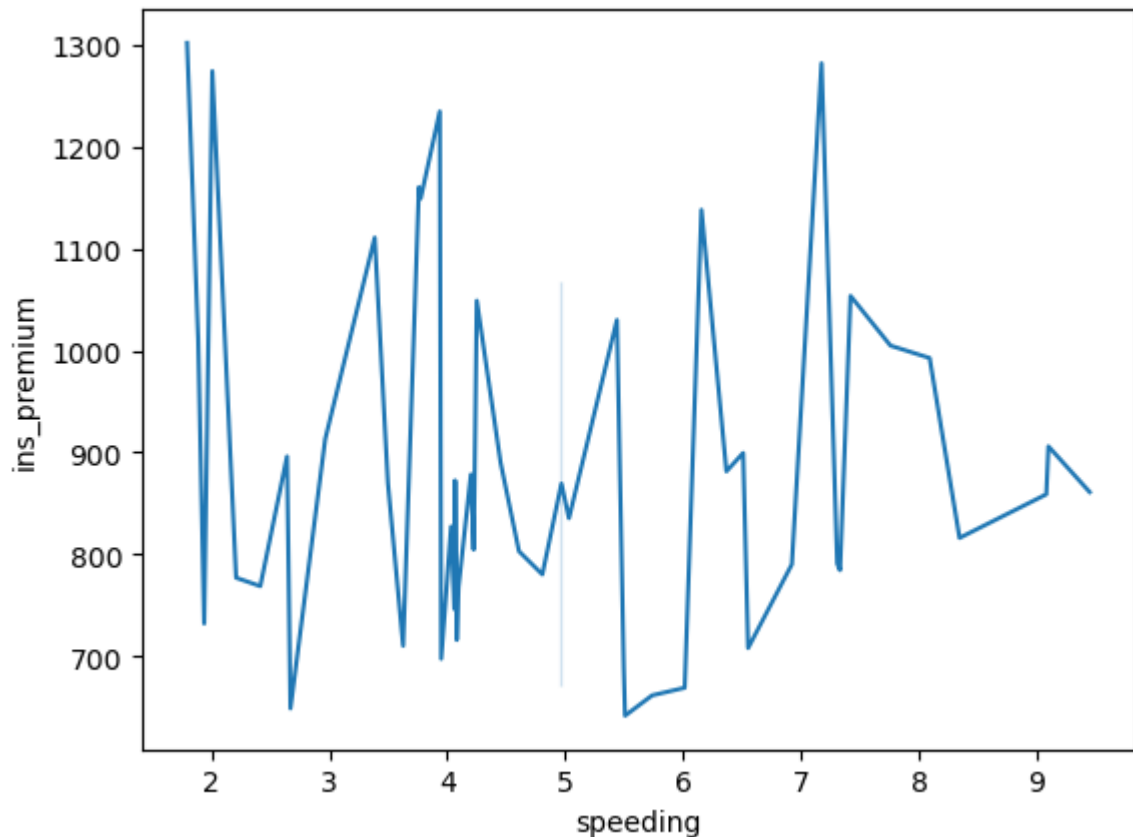
Out[22]: <Axes: xlabel='total', ylabel='alcohol'>



In []: We already discussed that the people who are drinking **and** driving are more prone to accidents. This conclusion is justified. But **for** the same number of total people who faced an accident, the number of people who were drinking alcohol was **not** exactly the same. Some states were affected by this more severely than the others.

In [50]: `sns.lineplot(y="ins_premium",x="speeding",data=ak)`

Out[50]: `<Axes: xlabel='speeding', ylabel='ins_premium'>`



In []: The relation between people who faced the car crash majorly because of speeding and th
in their state are not exactly correlated in any way. The states where the Insurance p
New York, New Jersey and District of Columbia.

In [24]: `#Displot`
`sns.distplot(ak["total"])`

C:\Users\nagka\AppData\Local\Temp\ipykernel_14956\2663308116.py:2: UserWarning:

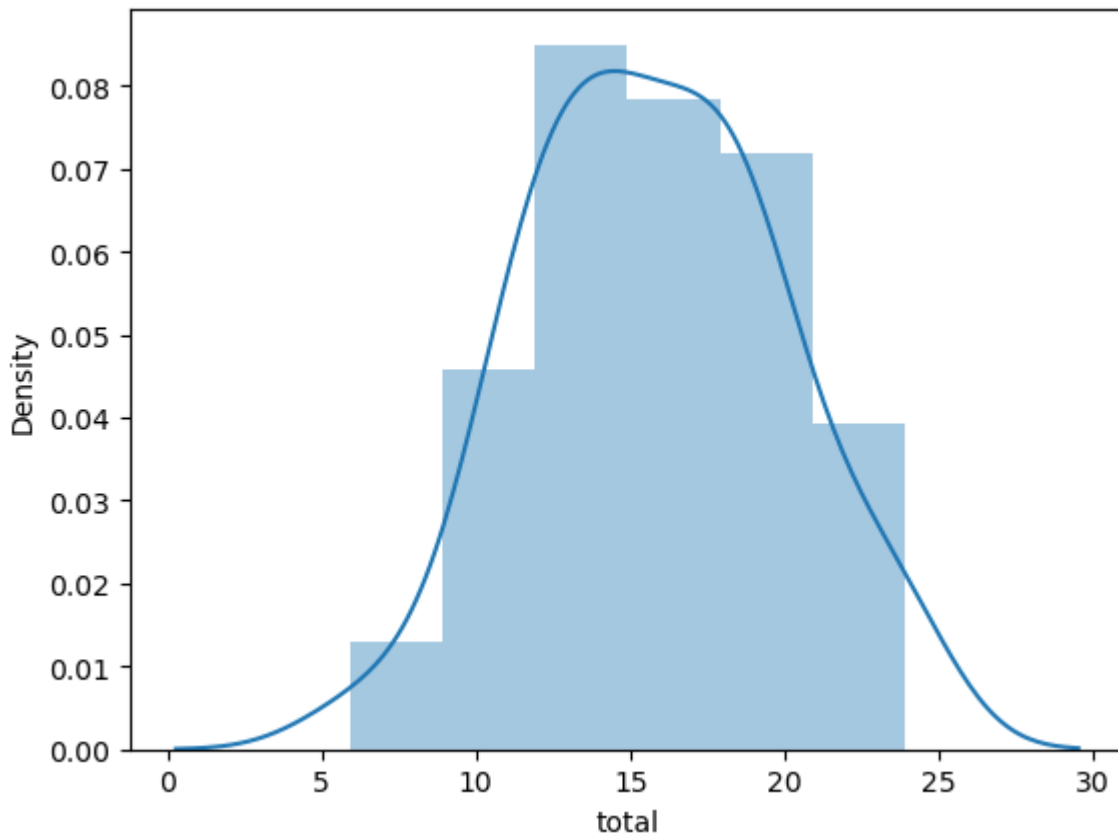
`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

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(ak["total"])`

Out[24]: <Axes: xlabel='total', ylabel='Density'>



In []: This above distribution plot shows that the distribution **is** unimodal. And there are **no** very less outliers **in** our data. We can observe the data **is in** a normal curve distribution. We can see that the shape of the distribution graph **is** quite symmetric. The states where the total is less than compared to other states are Kentucky, Arkansas **and** Louisiana.

In [25]: `sns.distplot(ak["not_distracted"])`

C:\Users\nagka\AppData\Local\Temp\ipykernel_14956\3265424172.py:1: UserWarning:

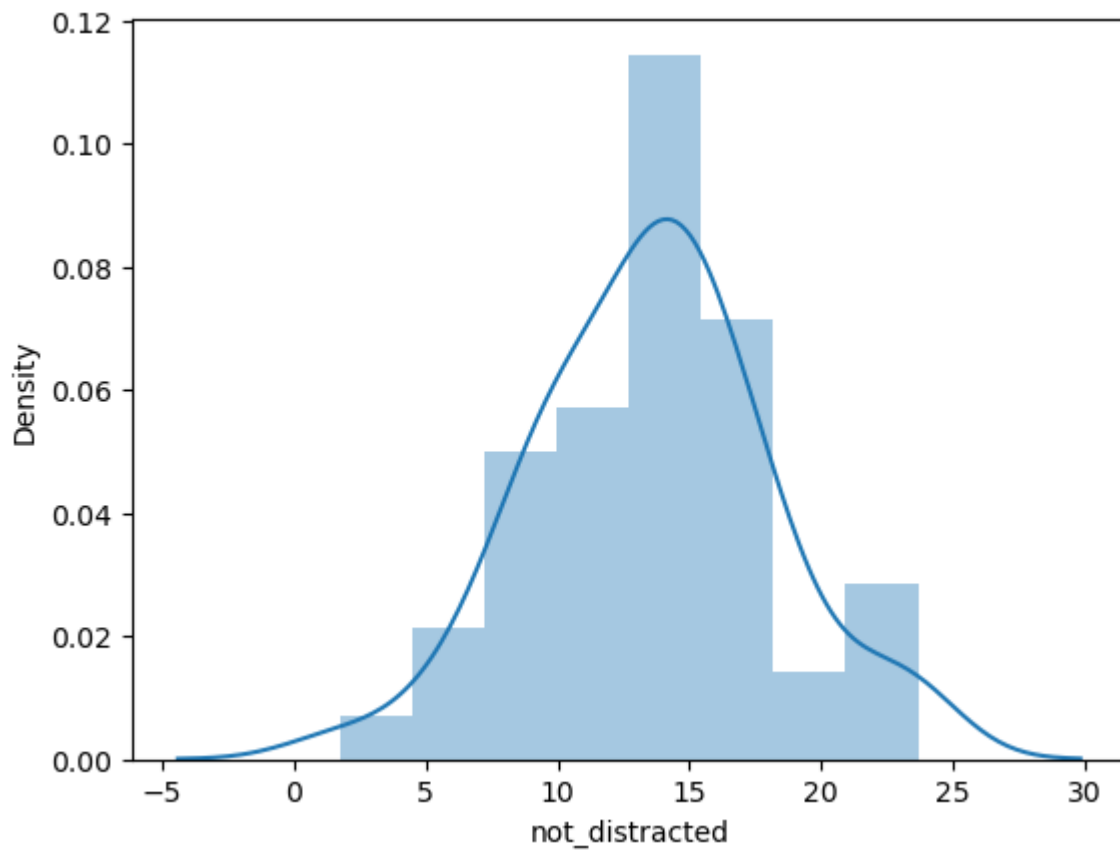
``distplot` is a deprecated function and will be removed in seaborn v0.14.0.`

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(ak["not_distracted"])`

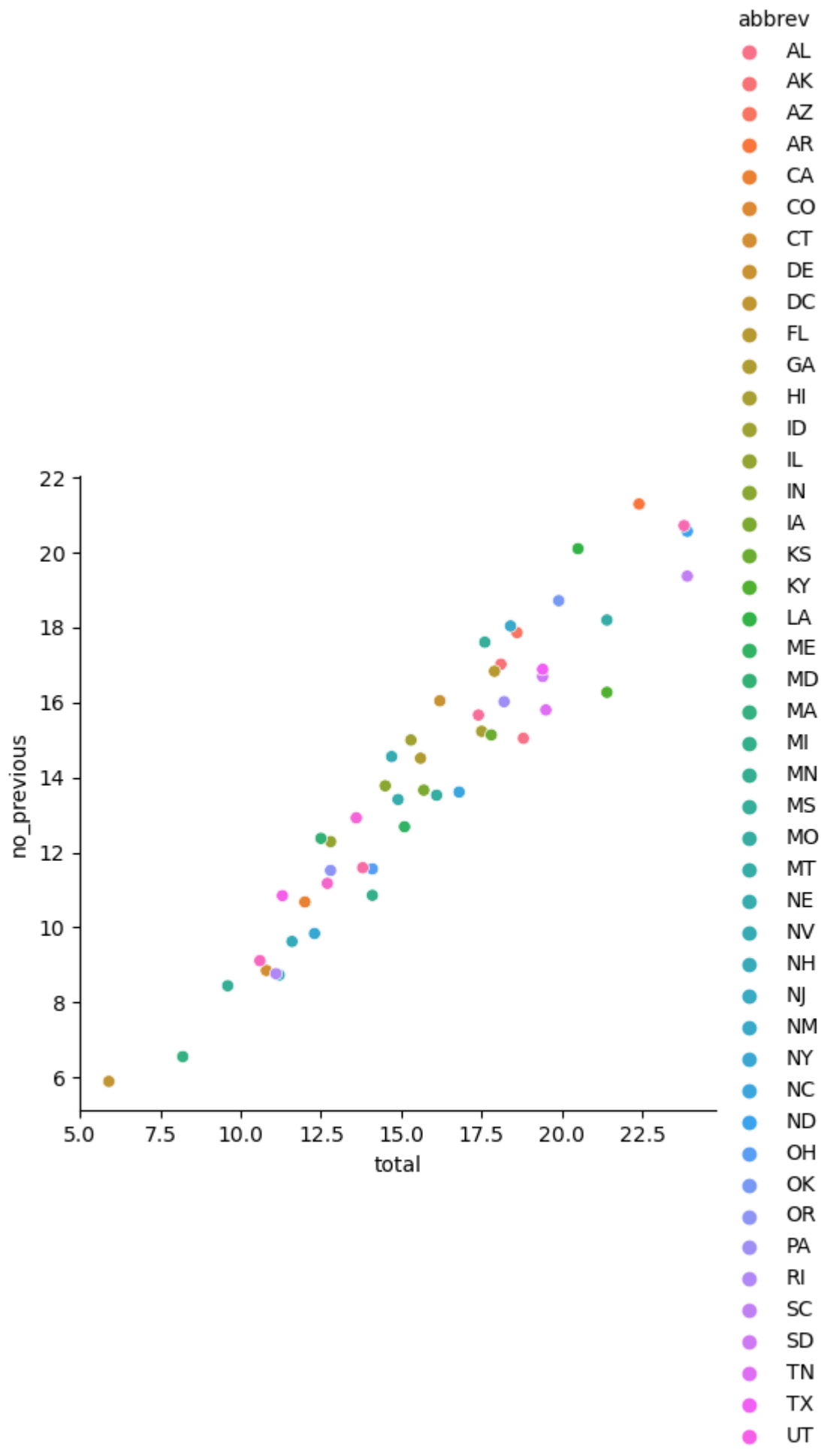
Out[25]: `<Axes: xlabel='not_distracted', ylabel='Density'>`



In []: This above distribution plot shows that the distribution **is** unimodal. We can see that Negatively Skewed **as** the Q2 **is** close to Q3. And there are **not** many infactvery less out data **is in** a normal curve distribution shape. The states which involved people sufferi distracted **from** driving were

In [52]: `#Relationplot`
`sns.relplot(x="total",y="no_previous",data=ak,hue="abbrev")`

Out[52]: <seaborn.axisgrid.FacetGrid at 0x1e4f9abb910>

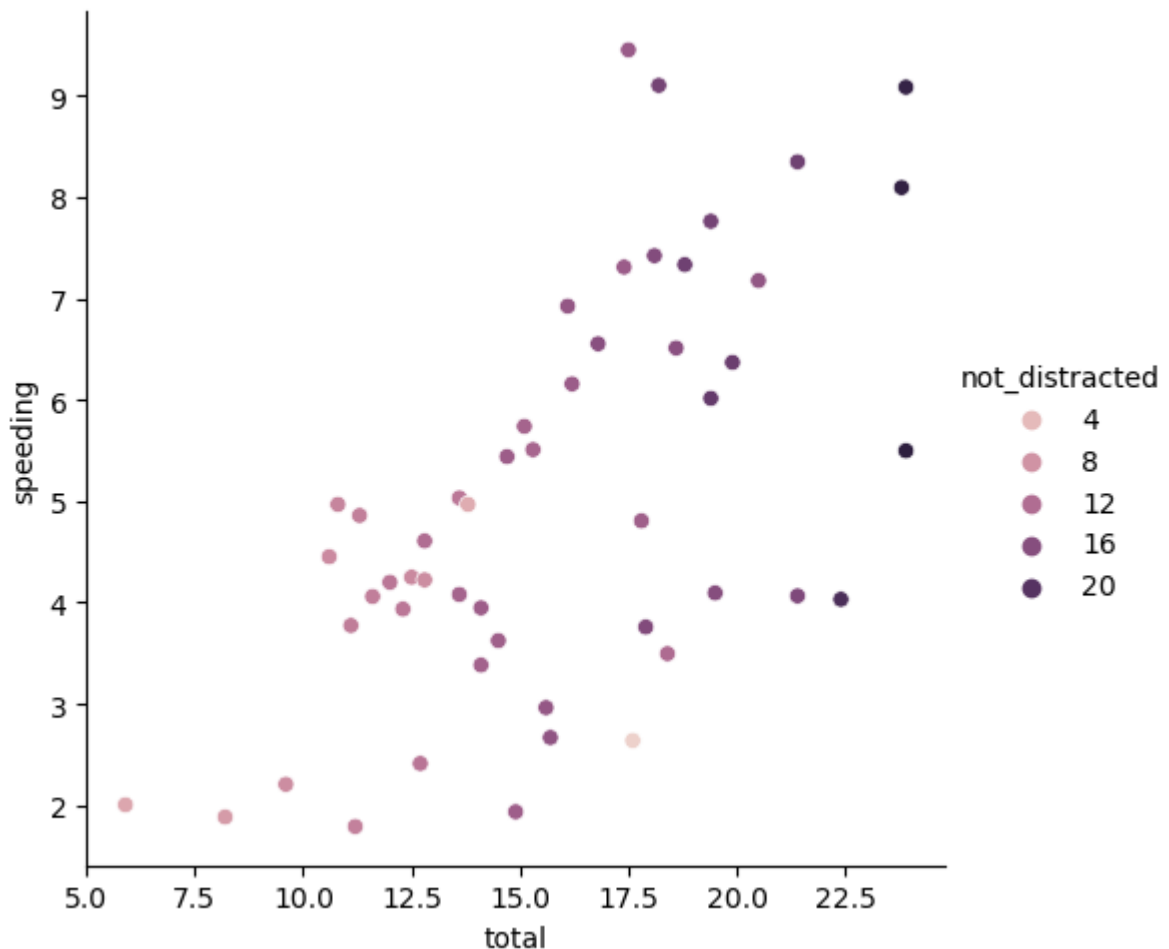


- VT
- VA
- WA
- WV
- WI
- WY

In []: We can see that the total number of people who faced the accident had no previous most related or positively correlated data. This graph clearly shows as the more number of people found out to be having no previous record of any fatal accidents as such. 50 states are no_previous count of data in their respective states.

```
In [51]: #Relationplot
sns.relplot(x="total",y="speeding",data=ak,hue="not_distracted")
```

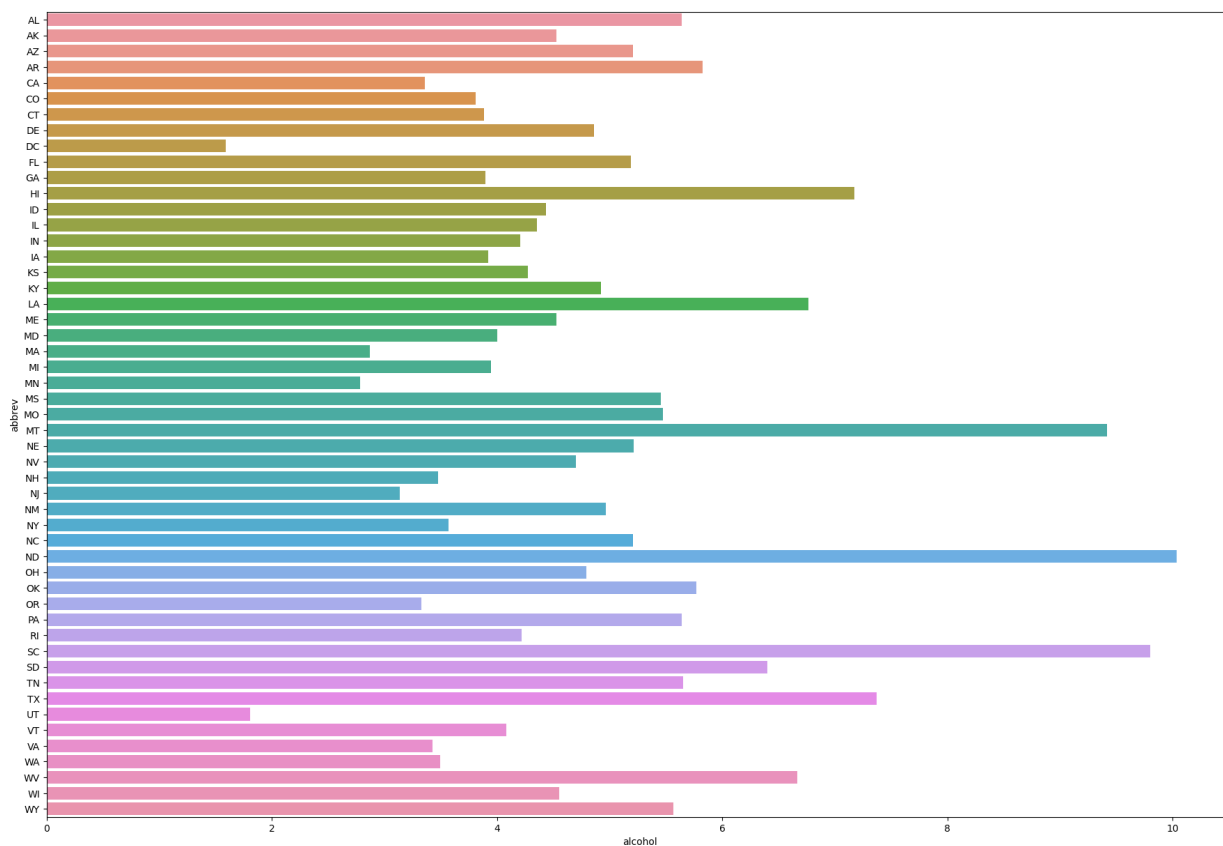
Out[51]: <seaborn.axisgrid.FacetGrid at 0x1e4f887d090>



In []: This graph is almost scattered widely that is they are not very much related. But yes, those who faced the fatal car crash were speeding but most of them are not speeding. So we can see that those who faced the accident were not at all distracted from driving but due to speeding they were not speeding can be clearly seen as not being focussed that is being more distracted.

```
In [53]: #barplot
plt.subplots(figsize=(22,15))
sns.barplot(x="alcohol",y="abbrev",data=ak)
```

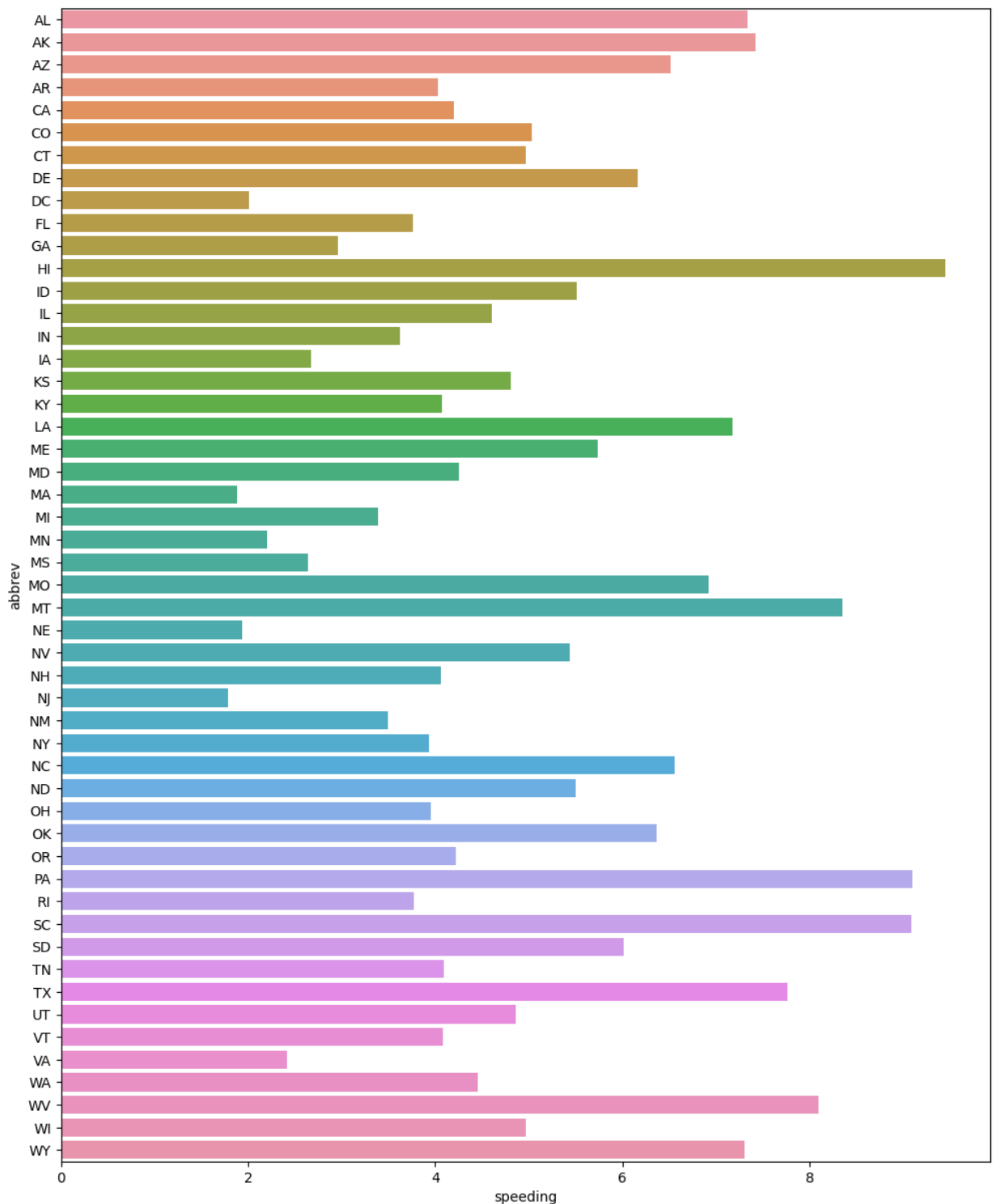
Out[53]: <Axes: xlabel='alcohol', ylabel='abbrev'>



In []: This shows that how much that particular state was affected majorly because of consumption of alcohol. The top states that faced Alcohol Impairment were South Carolina, North Carolina, and Texas.

In [54]: `#barplot`
`plt.subplots(figsize=(12,15))`
`sns.barplot(x="speeding",y="abbrev",data=ak)`

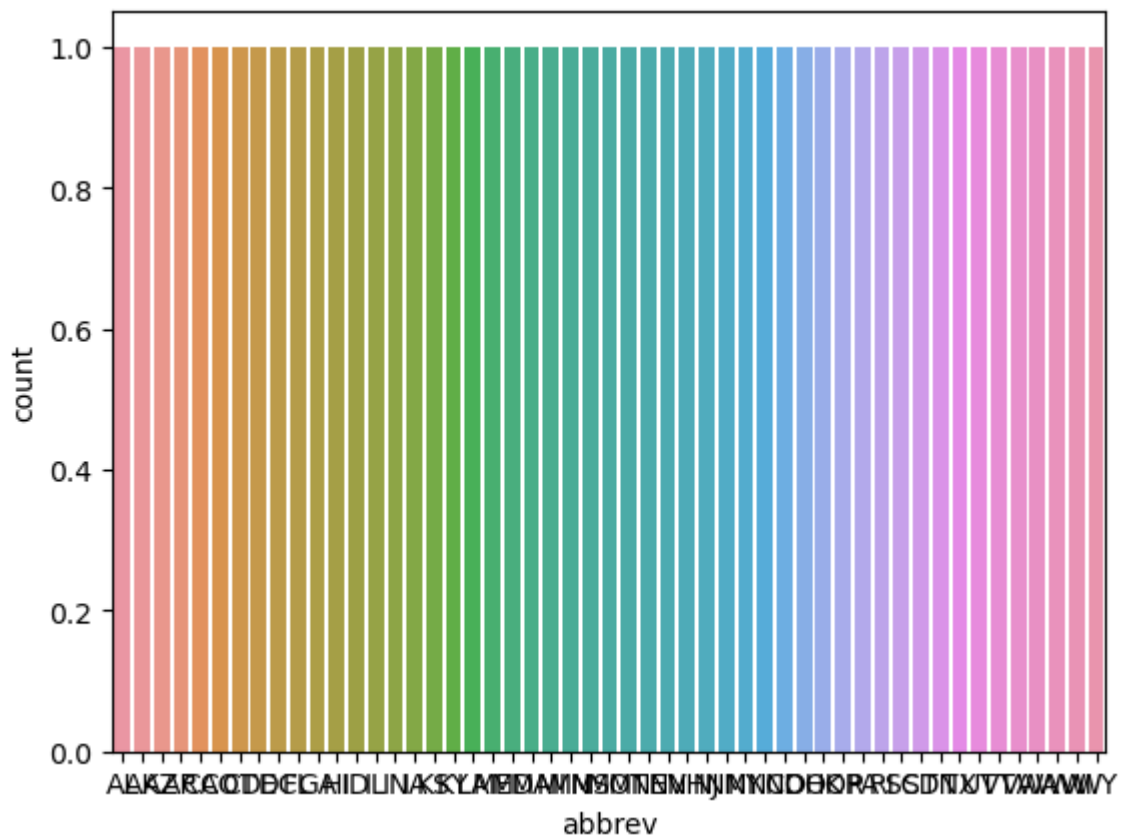
Out[54]: <Axes: xlabel='speeding', ylabel='abbrev'>



In []: This shows that how much that particular state was affected majorly because of speeding. That is how many car crashes took place because the driver was speeding. The top state issues were Hawaii, Pennsylvania and South Carolina.

```
In [30]: #countplot
sns.countplot(x="abbrev", data=ak)
```

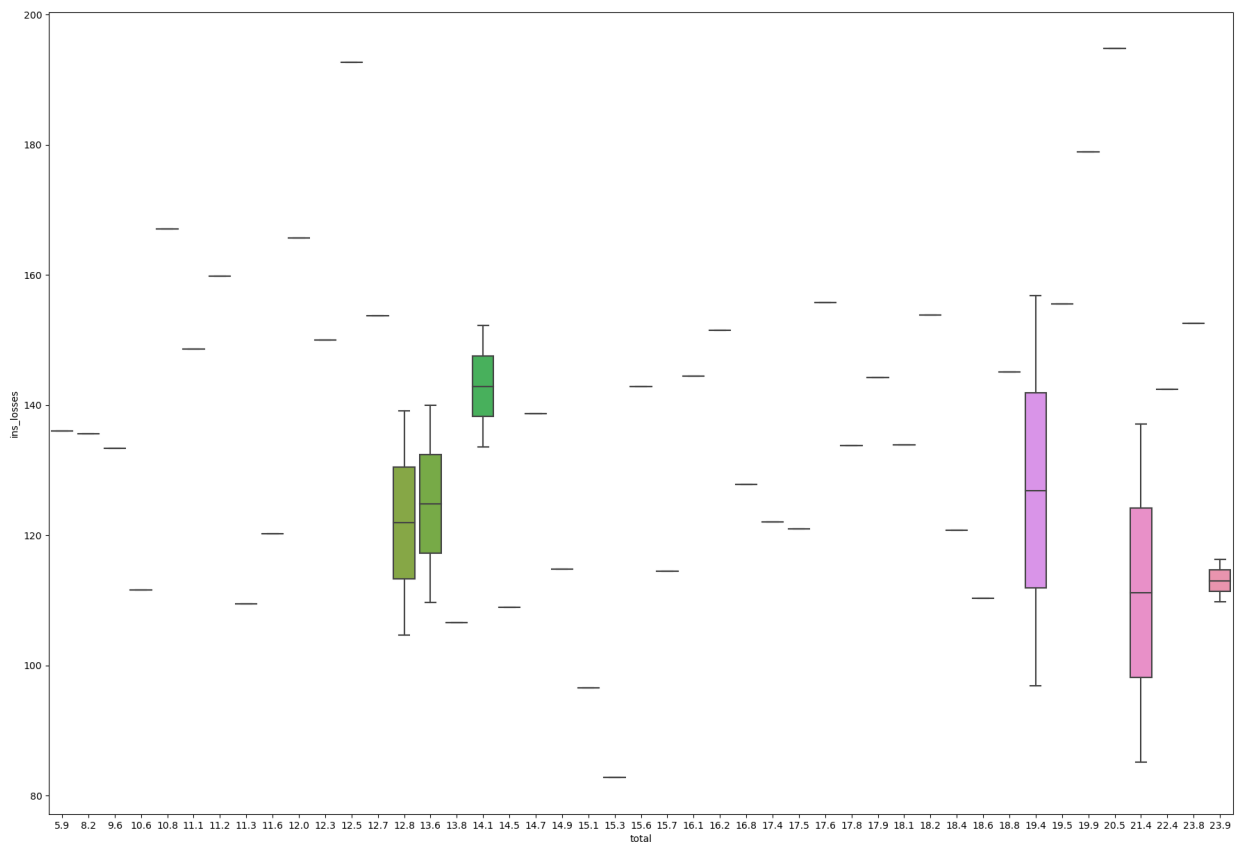
Out[30]: <Axes: xlabel='abbrev', ylabel='count'>



In []: This is a basic graph where we can see how many times a state has been repeated in the data. We can see every state frequency is exactly 1.

```
In [31]: #boxplot
plt.subplots(figsize=(22,15))
sns.boxplot(x="total",y="ins_losses",data=ak)
```

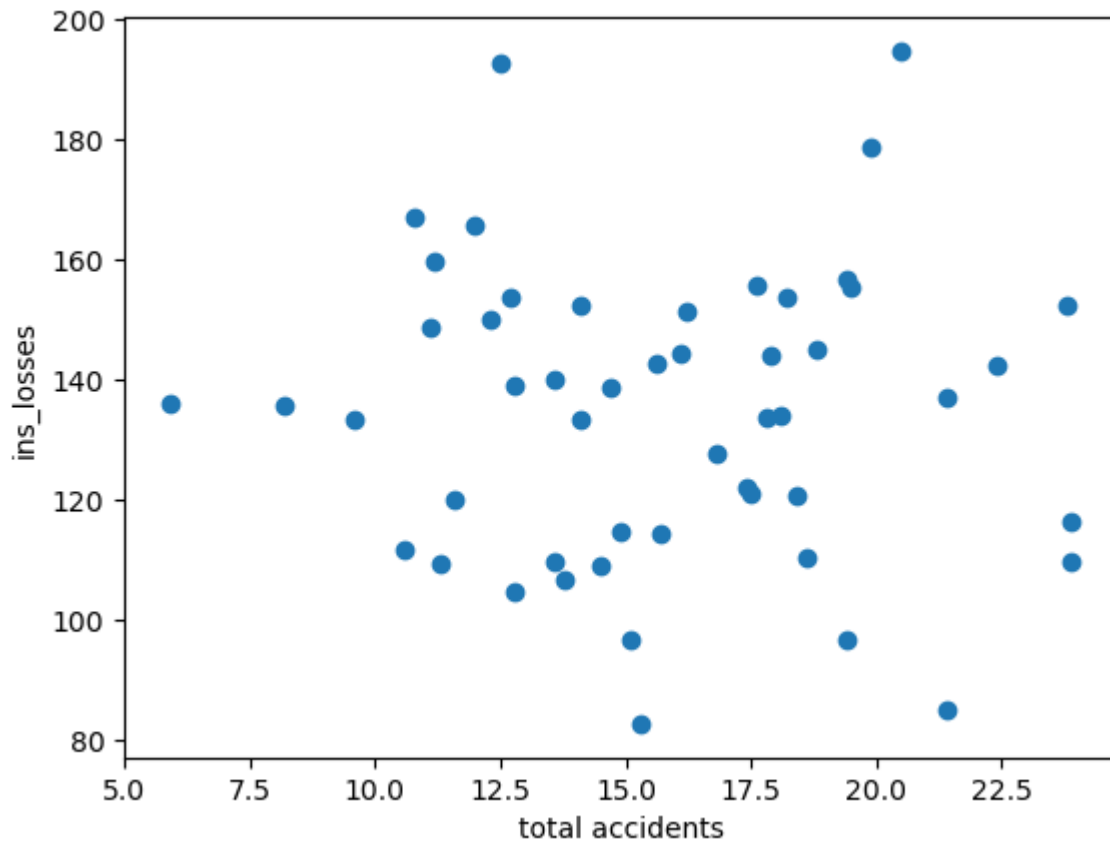
Out[31]: <Axes: xlabel='total', ylabel='ins_losses'>



In []: We can clearly see the data is not skewed positively nor negatively, i.e the data is not skewed. We can also see that there are no outliers in the data of the

```
In [55]: x=ak["total"]
y=ak["abbrev"]
plt.plot(x,y)
plt.ylabel("State")
plt.xlabel("State's Total")
```

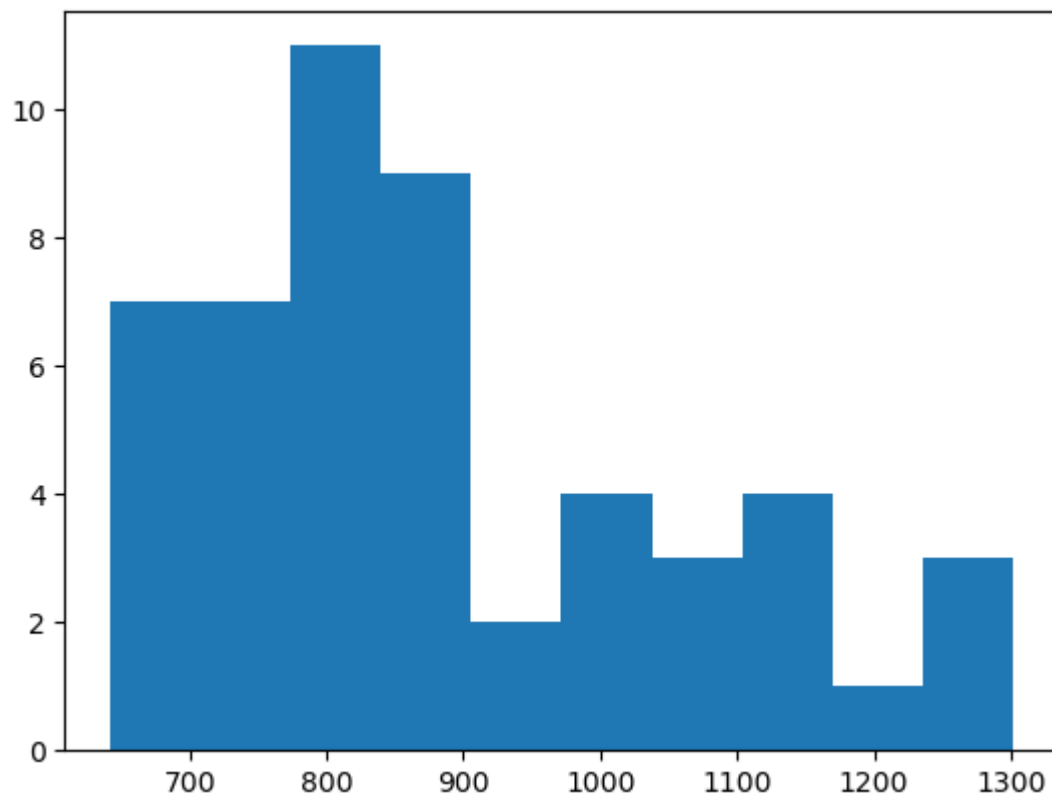
Out[55]: Text(0.5, 0, "State's Total")



In []: This graph **is** widely scattered **as** the insurance losses that the companies are facing **are** varying **from** one state to another.

```
In [48]: r = ak["ins_premium"]
plt.hist(r)
```

```
Out[48]: (array([ 7.,  7., 11.,  9.,  2.,  4.,  3.,  4.,  1.,  3.]),
array([ 641.96 ,  707.916,  773.872,  839.828,  905.784,  971.74 ,
        1037.696, 1103.652, 1169.608, 1235.564, 1301.52 ]),
<BarContainer object of 10 artists>)
```

In []: This is a histogram depicting the Insurance Premium that the company is offering. This