ULLI PAVAN KALYAN

21BCE9090

CSE(AI&ML)

VIT-AP

9392429845

```python
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', 'penguins', 'planets',
'seaice', 'taxis', 'tips', 'titanic']
```

sns.scatterplot(x="alcohol",y="total",data=df)

Premise: This code creates a scatterplot using Seaborn to visualize the relationship between two variables, "alcohol" and "total," from a DataFrame df. It helps explore how these variables are distributed and if there is any observable pattern or correlation between them.

sns.lineplot(x="alcohol",y="total",data=df)

Premise: This code uses Seaborn to create a line plot, displaying the relationship between "alcohol" and "total" variables from the DataFrame df. It allows for observing trends or patterns in the data.

sns.distplot(df['alcohol'])

Premise: This code generates a distribution plot (histogram) using Seaborn for the "alcohol" variable within the DataFrame df. It helps visualize the distribution of values and assess the data's central tendency and spread.

sns.relplot(x="alcohol",y="total",data=df,hue="abbrev")

Premise: This code produces a relational plot (relplot) using Seaborn, showing the relationship between "alcohol" and "total" variables from the DataFrame df. The hue parameter is used to differentiate data points based on the "abbrev" column, allowing for the exploration of the relationship across different categories.

sns.barplot(data=df,x="alcohol",y="total")

Premise: This code creates a bar plot (barplot) using Seaborn to visualize the relationship between "alcohol" and "total" variables from the DataFrame df. It displays the values as bars, providing insights into how these variables compare.

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

Premise: This code uses Seaborn to generate a count plot (countplot) that counts the occurrences of unique values in the "alcohol" column of the DataFrame df. It helps visualize the distribution and frequency of each unique value in the "alcohol" variable.

sns.jointplot(x="alcohol",y="total",data=df):

Premise: This code creates a joint plot (jointplot) using Seaborn, combining a scatterplot of "alcohol" and "total" variables with marginal histograms of each variable. It facilitates the examination of the relationship between the two variables and their individual distributions.

corr=df.corr() and corr

Premise: These lines of code calculate and store the correlation matrix (corr) for numeric columns in the DataFrame df. The matrix contains correlation coefficients, providing insights into the linear relationships between different variables in the dataset.

sns.heatmap(corr,annot=True,cmap="YlGnBu")

Premise: This code generates a heatmap using Seaborn to visually represent the correlation matrix (corr) calculated earlier. The heatmap uses colors to show the strength and direction of correlations between pairs of numeric variables, with annotations for clarity. It helps identify patterns and dependencies within the dataset.

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

    total  speeding  alcohol  not_distracted  no_previous  ins_premium
\
0    18.8     7.332    5.640          18.048       15.040       784.55

1    18.1     7.421    4.525          16.290       17.014      1053.48

2    18.6     6.510    5.208          15.624       17.856       899.47

3    22.4     4.032    5.824          21.056       21.280       827.34

4    12.0     4.200    3.360          10.920       10.680       878.41

5    13.6     5.032    3.808          10.744       12.920       835.50

6    10.8     4.968    3.888           9.396        8.856      1068.73

7    16.2     6.156    4.860          14.094       16.038      1137.87

8     5.9     2.006    1.593           5.900        5.900      1273.89

9    17.9     3.759    5.191          16.468       16.826      1160.13

10   15.6     2.964    3.900          14.820       14.508       913.15

11   17.5     9.450    7.175          14.350       15.225       861.18
```

12	15.3	5.508	4.437	13.005	14.994	641.96
13	12.8	4.608	4.352	12.032	12.288	803.11
14	14.5	3.625	4.205	13.775	13.775	710.46
15	15.7	2.669	3.925	15.229	13.659	649.06
16	17.8	4.806	4.272	13.706	15.130	780.45
17	21.4	4.066	4.922	16.692	16.264	872.51
18	20.5	7.175	6.765	14.965	20.090	1281.55
19	15.1	5.738	4.530	13.137	12.684	661.88
20	12.5	4.250	4.000	8.875	12.375	1048.78
21	8.2	1.886	2.870	7.134	6.560	1011.14
22	14.1	3.384	3.948	13.395	10.857	1110.61
23	9.6	2.208	2.784	8.448	8.448	777.18
24	17.6	2.640	5.456	1.760	17.600	896.07
25	16.1	6.923	5.474	14.812	13.524	790.32
26	21.4	8.346	9.416	17.976	18.190	816.21
27	14.9	1.937	5.215	13.857	13.410	732.28
28	14.7	5.439	4.704	13.965	14.553	1029.87
29	11.6	4.060	3.480	10.092	9.628	746.54
30	11.2	1.792	3.136	9.632	8.736	1301.52
31	18.4	3.496	4.968	12.328	18.032	869.85
32	12.3	3.936	3.567	10.824	9.840	1234.31
33	16.8	6.552	5.208	15.792	13.608	708.24
34	23.9	5.497	10.038	23.661	20.554	688.75
35	14.1	3.948	4.794	13.959	11.562	697.73
36	19.9	6.368	5.771	18.308	18.706	881.51
37	12.8	4.224	3.328	8.576	11.520	804.71

```
38   18.2     9.100    5.642          17.472       16.016      905.99
39   11.1     3.774    4.218          10.212        8.769     1148.99
40   23.9     9.082    9.799          22.944       19.359      858.97
41   19.4     6.014    6.402          19.012       16.684      669.31
42   19.5     4.095    5.655          15.990       15.795      767.91
43   19.4     7.760    7.372          17.654       16.878     1004.75
44   11.3     4.859    1.808           9.944       10.848      809.38
45   13.6     4.080    4.080          13.056       12.920      716.20
46   12.7     2.413    3.429          11.049       11.176      768.95
47   10.6     4.452    3.498           8.692        9.116      890.03
48   23.8     8.092    6.664          23.086       20.706      992.61
49   13.8     4.968    4.554           5.382       11.592      670.31
50   17.4     7.308    5.568          14.094       15.660      791.14

    ins_losses abbrev
0       145.08     AL
1       133.93     AK
2       110.35     AZ
3       142.39     AR
4       165.63     CA
5       139.91     CO
6       167.02     CT
7       151.48     DE
8       136.05     DC
9       144.18     FL
10      142.80     GA
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
```

```
<bound method DataFrame.info of      total  speeding  alcohol
not_distracted  no_previous  ins_premium  \
0    18.8     7.332    5.640          18.048       15.040       784.55

1    18.1     7.421    4.525          16.290       17.014      1053.48

2    18.6     6.510    5.208          15.624       17.856       899.47

3    22.4     4.032    5.824          21.056       21.280       827.34

4    12.0     4.200    3.360          10.920       10.680       878.41

5    13.6     5.032    3.808          10.744       12.920       835.50

6    10.8     4.968    3.888           9.396        8.856      1068.73

7    16.2     6.156    4.860          14.094       16.038      1137.87

8     5.9     2.006    1.593           5.900        5.900      1273.89
```

```
9    17.9    3.759   5.191          16.468      16.826    1160.13
10   15.6    2.964   3.900          14.820      14.508     913.15
11   17.5    9.450   7.175          14.350      15.225     861.18
12   15.3    5.508   4.437          13.005      14.994     641.96
13   12.8    4.608   4.352          12.032      12.288     803.11
14   14.5    3.625   4.205          13.775      13.775     710.46
15   15.7    2.669   3.925          15.229      13.659     649.06
16   17.8    4.806   4.272          13.706      15.130     780.45
17   21.4    4.066   4.922          16.692      16.264     872.51
18   20.5    7.175   6.765          14.965      20.090    1281.55
19   15.1    5.738   4.530          13.137      12.684     661.88
20   12.5    4.250   4.000           8.875      12.375    1048.78
21    8.2    1.886   2.870           7.134       6.560    1011.14
22   14.1    3.384   3.948          13.395      10.857    1110.61
23    9.6    2.208   2.784           8.448       8.448     777.18
24   17.6    2.640   5.456           1.760      17.600     896.07
25   16.1    6.923   5.474          14.812      13.524     790.32
26   21.4    8.346   9.416          17.976      18.190     816.21
27   14.9    1.937   5.215          13.857      13.410     732.28
28   14.7    5.439   4.704          13.965      14.553    1029.87
29   11.6    4.060   3.480          10.092       9.628     746.54
30   11.2    1.792   3.136           9.632       8.736    1301.52
31   18.4    3.496   4.968          12.328      18.032     869.85
32   12.3    3.936   3.567          10.824       9.840    1234.31
33   16.8    6.552   5.208          15.792      13.608     708.24
34   23.9    5.497  10.038          23.661      20.554     688.75
```

```
35   14.1    3.948   4.794         13.959      11.562     697.73

36   19.9    6.368   5.771         18.308      18.706     881.51

37   12.8    4.224   3.328          8.576      11.520     804.71

38   18.2    9.100   5.642         17.472      16.016     905.99

39   11.1    3.774   4.218         10.212       8.769    1148.99

40   23.9    9.082   9.799         22.944      19.359     858.97

41   19.4    6.014   6.402         19.012      16.684     669.31

42   19.5    4.095   5.655         15.990      15.795     767.91

43   19.4    7.760   7.372         17.654      16.878    1004.75

44   11.3    4.859   1.808          9.944      10.848     809.38

45   13.6    4.080   4.080         13.056      12.920     716.20

46   12.7    2.413   3.429         11.049      11.176     768.95

47   10.6    4.452   3.498          8.692       9.116     890.03

48   23.8    8.092   6.664         23.086      20.706     992.61

49   13.8    4.968   4.554          5.382      11.592     670.31

50   17.4    7.308   5.568         14.094      15.660     791.14
```

```
    ins_losses abbrev
0       145.08     AL
1       133.93     AK
2       110.35     AZ
3       142.39     AR
4       165.63     CA
5       139.91     CO
6       167.02     CT
7       151.48     DE
8       136.05     DC
9       144.18     FL
10      142.80     GA
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.head()
```

```
   total  speeding  alcohol  not_distracted  no_previous  ins_premium
\
0   18.8     7.332    5.640          18.048       15.040       784.55

1   18.1     7.421    4.525          16.290       17.014      1053.48

2   18.6     6.510    5.208          15.624       17.856       899.47

3   22.4     4.032    5.824          21.056       21.280       827.34

4   12.0     4.200    3.360          10.920       10.680       878.41
```

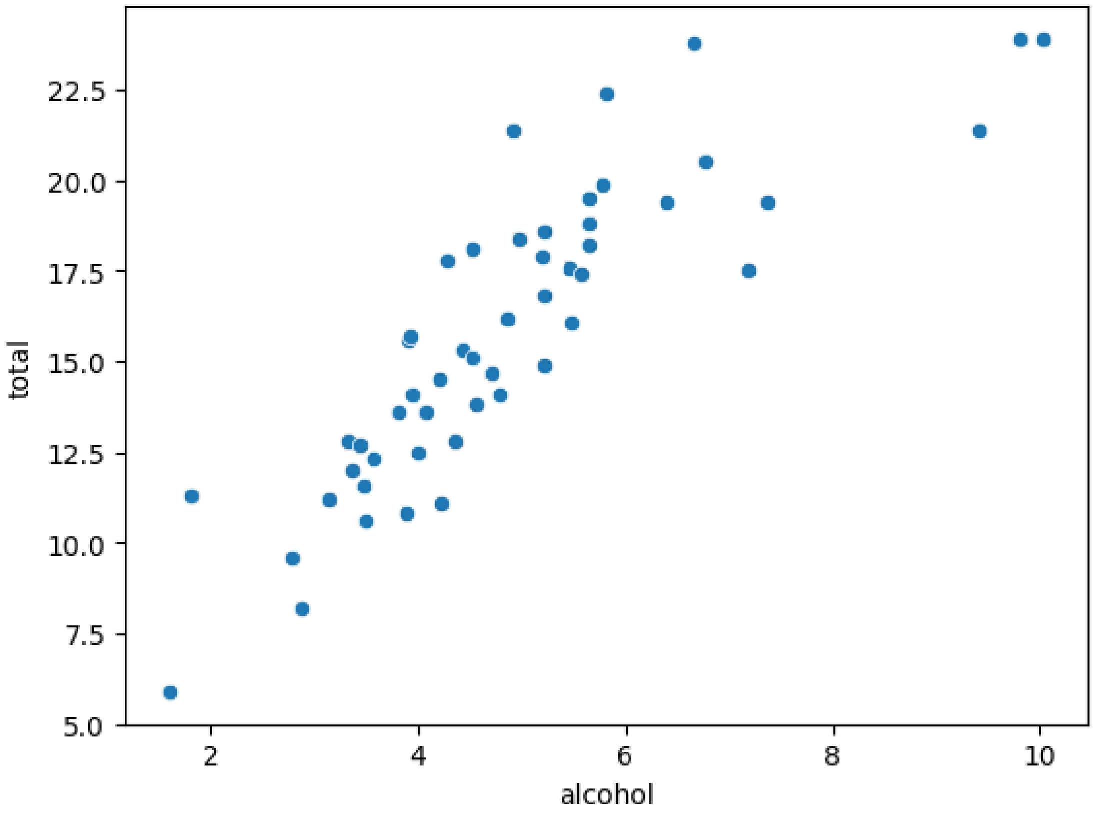
```
   ins_losses abbrev
0      145.08     AL
1      133.93     AK
2      110.35     AZ
3      142.39     AR
4      165.63     CA
```

```
df.tail()
```

```
    total  speeding  alcohol  not_distracted  no_previous  ins_premium 
\
46   12.7     2.413    3.429          11.049       11.176       768.95   

47   10.6     4.452    3.498           8.692        9.116       890.03   

48   23.8     8.092    6.664          23.086       20.706       992.61   

49   13.8     4.968    4.554           5.382       11.592       670.31   

50   17.4     7.308    5.568          14.094       15.660       791.14   


    ins_losses abbrev
46      153.72     VA
47      111.62     WA
48      152.56     WV
49      106.62     WI
50      122.04     WY
```

```
sns.scatterplot(x="alcohol",y="total",data=df)
```

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

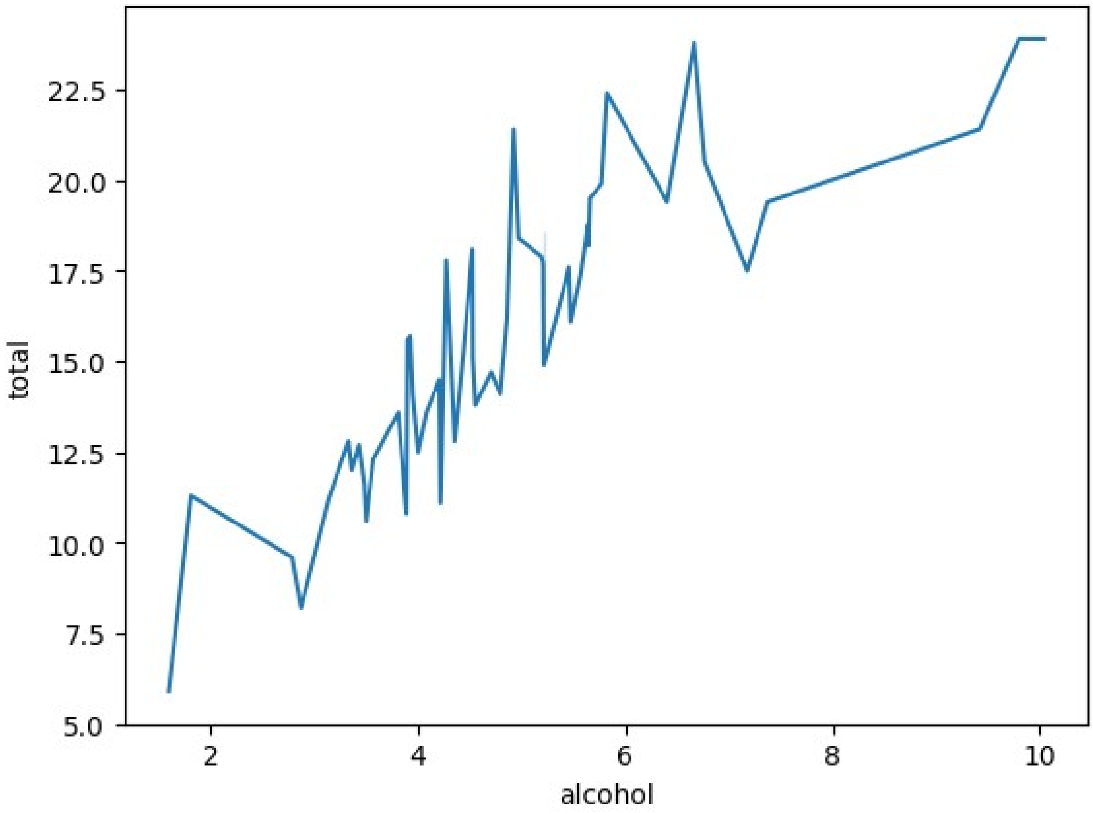


INFERENCE

This code uses Seaborn to create a scatterplot, visualizing the relationship between two variables, "alcohol" on the x-axis and "total" on the y-axis, using data from a DataFrame called df.

```
sns.lineplot(x="alcohol",y="total",data=df)
```

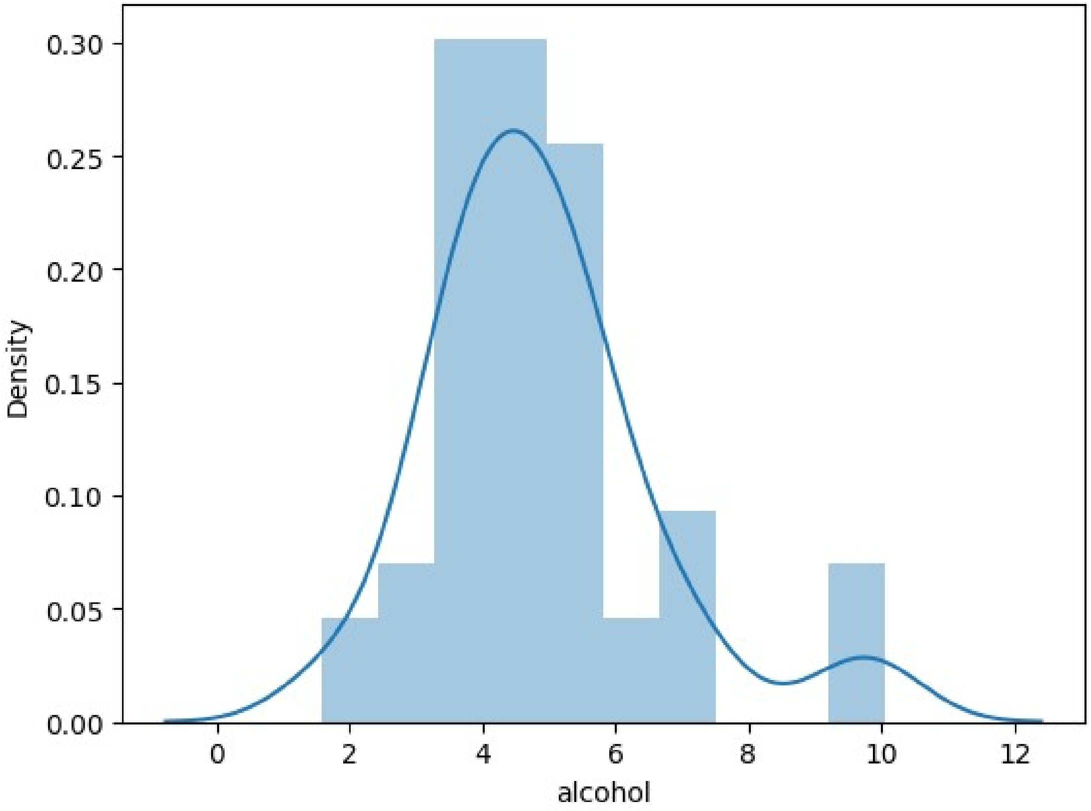
```
<Axes: xlabel='alcohol', ylabel='total'>
```



INFERENCE

This code uses Seaborn to create a line plot, displaying the relationship between two variables, "alcohol" on the x-axis and "total" on the y-axis, based on data from a DataFrame called df

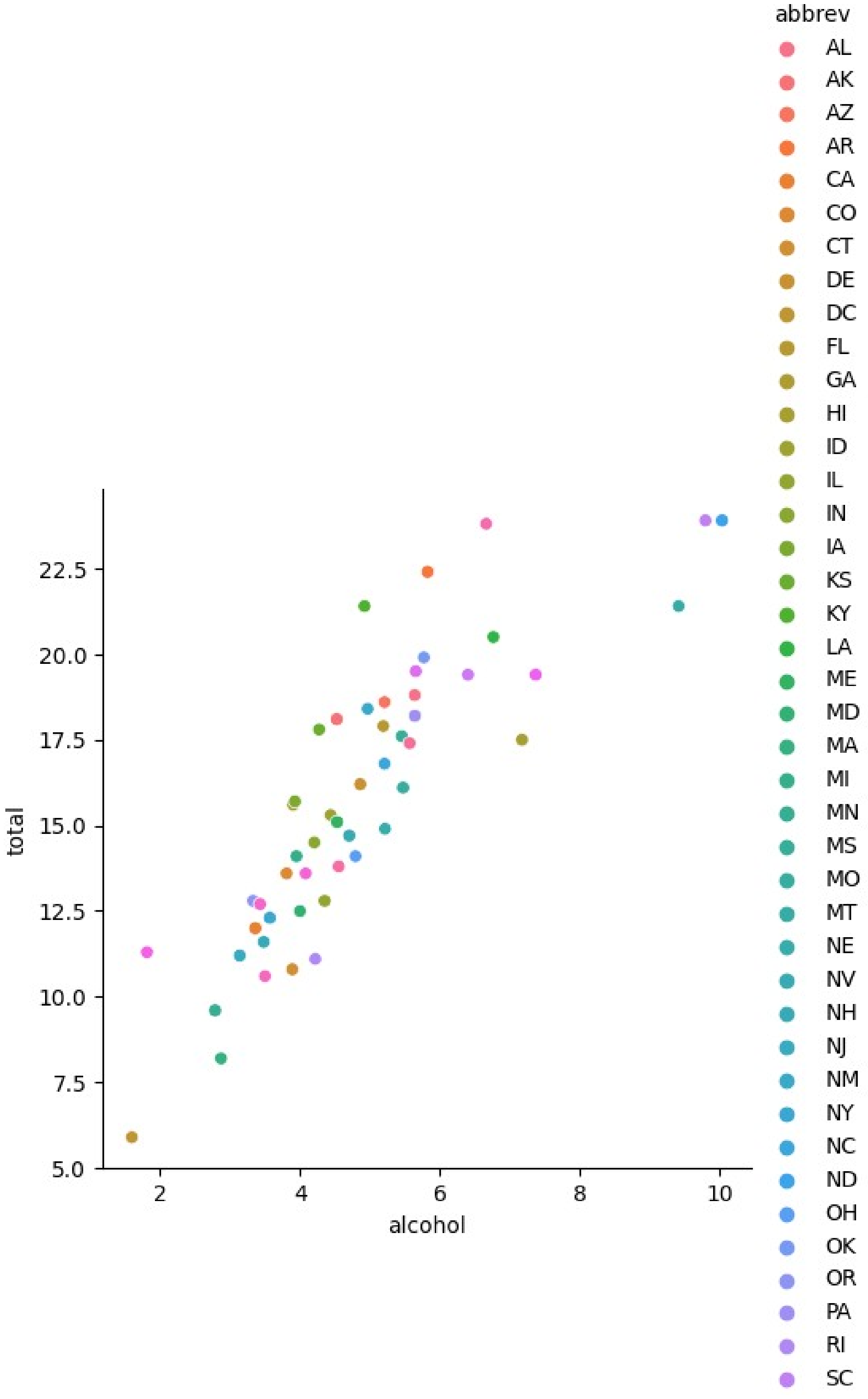
```
sns.distplot(df['alcohol'])

<ipython-input-9-570de8ff0310>: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['alcohol'])

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



INFERENCE

This code utilizes Seaborn to create a distribution plot (histogram) for the "alcohol" variable within the DataFrame df.

```
sns.relplot(x="alcohol",y="total",data=df,hue="abbrev")

<seaborn.axisgrid.FacetGrid at 0x7dfa5343d000>
```

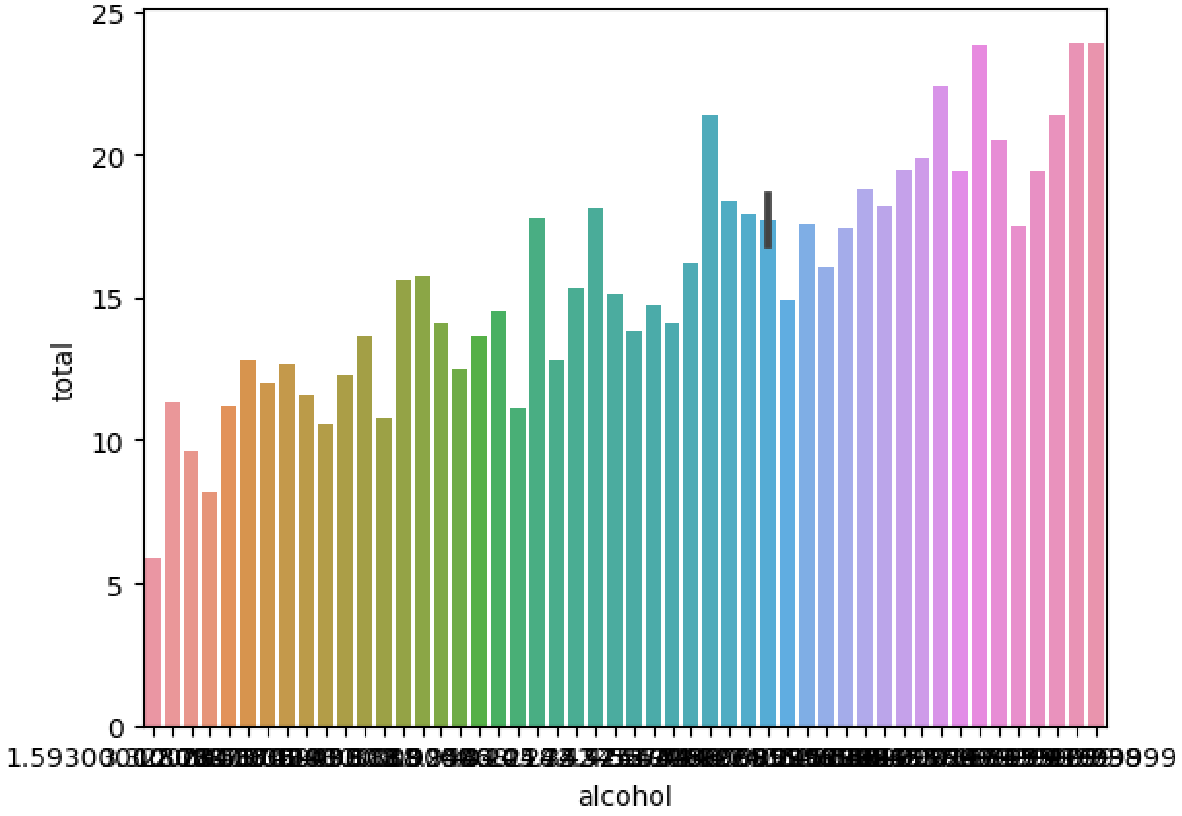

abbrev
AL
AK
AZ
AR
CA
CO
CT
DE
DC
FL
GA
HI
ID
IL
IN
IA
KS
KY
LA
ME
MD
MA
MI
MN
MS
MO
MT
NE
NV
NH
NJ
NM
NY
NC
ND
OH
OK
OR
PA
RI
SC
total
22.5
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2
4
6
8
10
alcohol

INFERENCE

This code generates a relational plot (relplot) using Seaborn, displaying the relationship between two variables, "alcohol" on the x-axis and "total" on the y-axis, from the DataFrame df. The hue parameter is used to color-code the data points based on the "abbrev" column in the DataFrame, allowing for the visualization of how the relationship varies across different categories represented by "abbrev."

```
sns.barplot(data=df,x="alcohol",y="total")
```

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

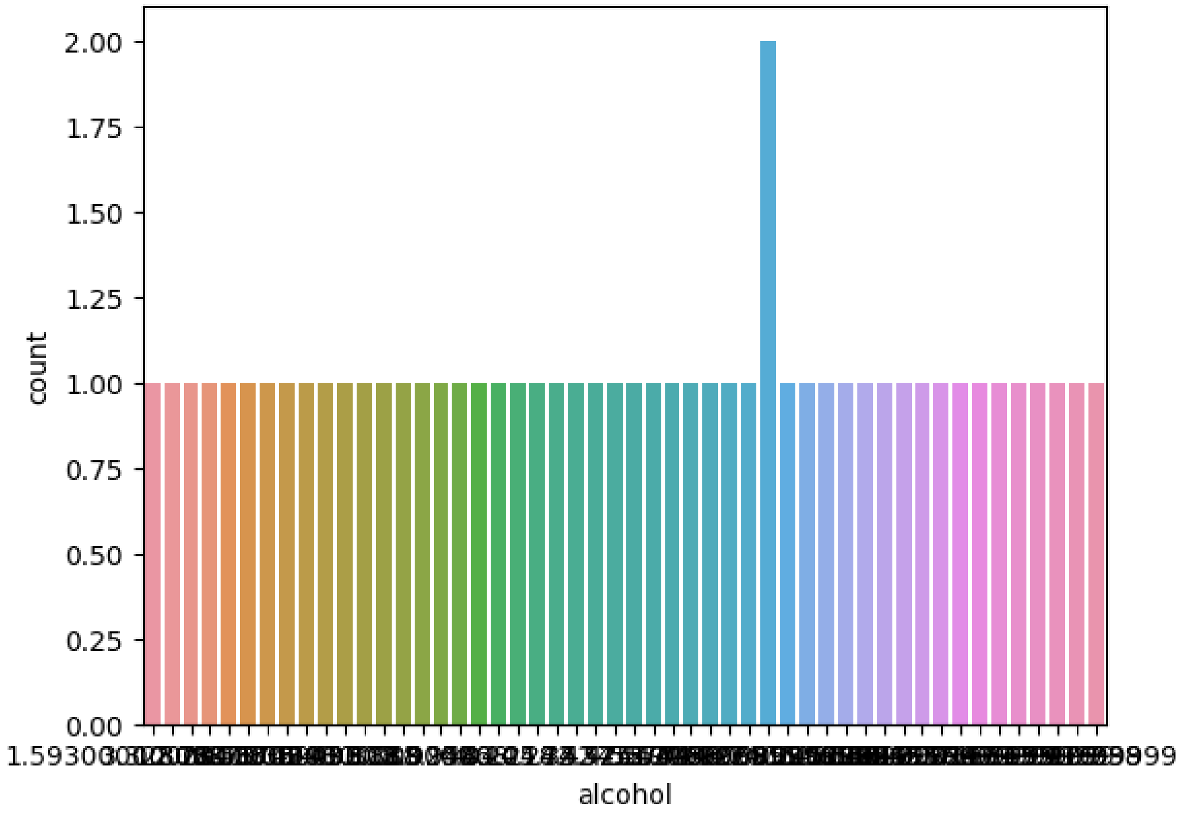


INFERENCE

This code utilizes Seaborn to create a bar plot (barplot), with "alcohol" on the x-axis and "total" on the y-axis, using data from the DataFrame df. It visualizes the relationship between these two variables through the height of the bars.

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

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

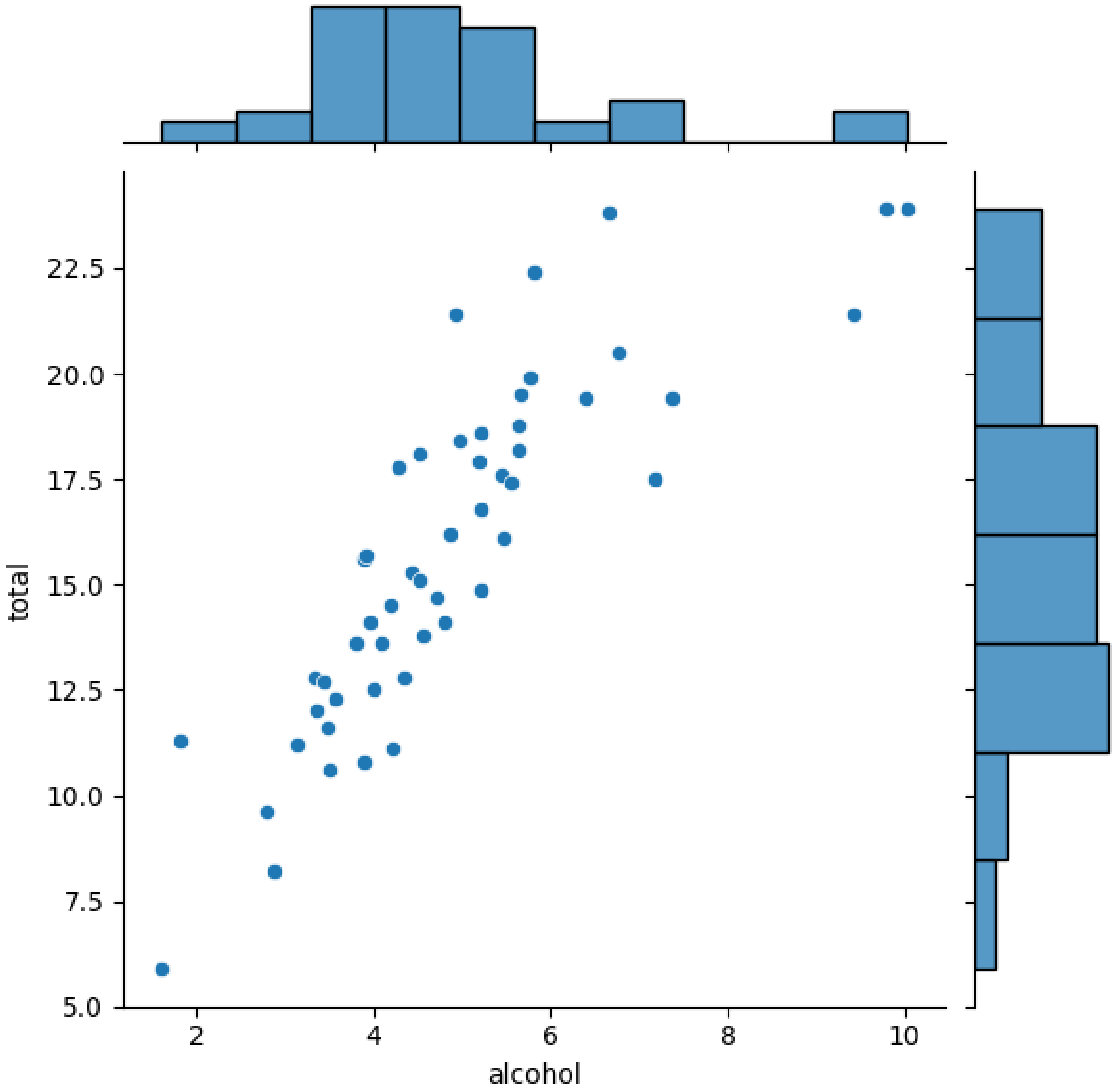


INFERENCE

This code uses Seaborn to create a count plot (countplot), where it counts the occurrences of unique values in the "alcohol" column of the DataFrame df and displays the counts as bars along the x-axis. It helps visualize the distribution of different values in the "alcohol" column.

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

```
<seaborn.axisgrid.JointGrid at 0x7dfa4c36a590>
```

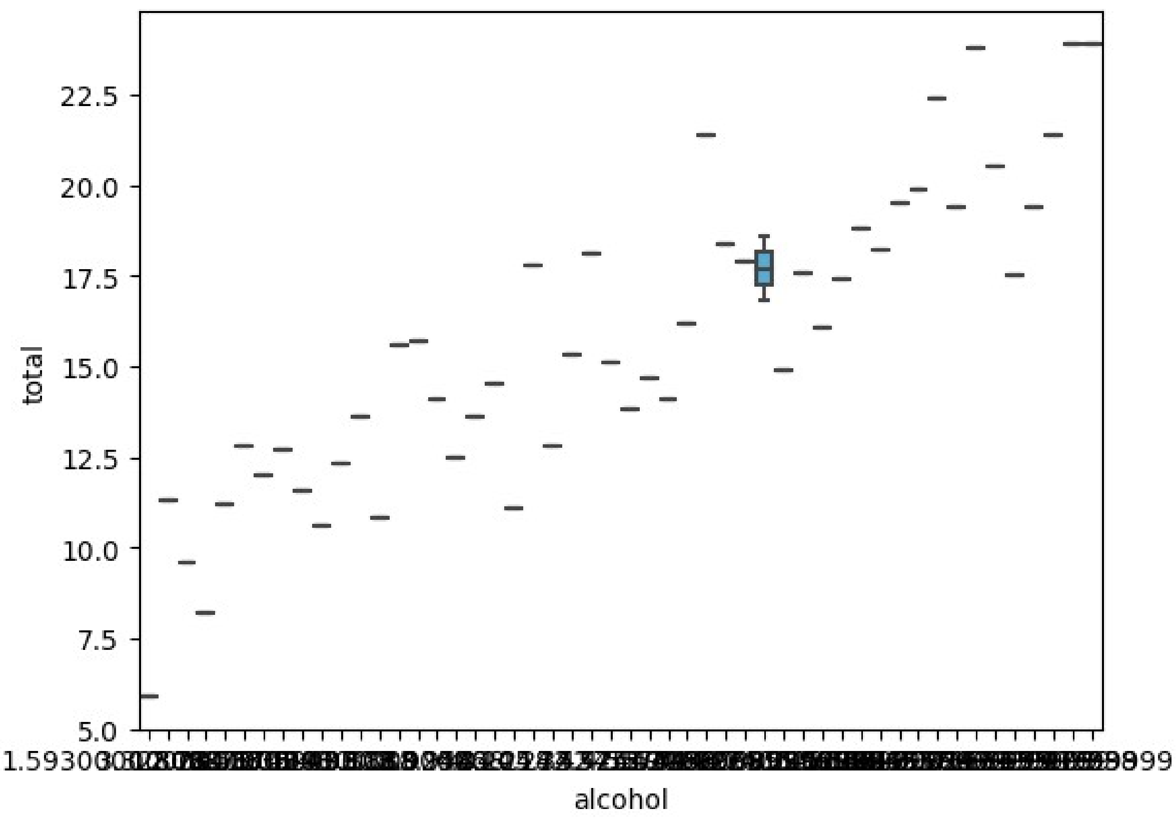


INFERENCE

This code generates a joint plot (jointplot) using Seaborn, which combines a scatterplot of "alcohol" on the x-axis and "total" on the y-axis with marginal histograms of each variable. This visualization helps explore the relationship between the two variables and their individual distributions.

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

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



INFERENCE

This code employs Seaborn to create a box plot (boxplot) that visualizes the distribution of the "total" variable grouped by different values of the "alcohol" variable from the DataFrame df. It provides information about the central tendency, spread, and potential outliers of the "total" variable for each category of the "alcohol" variable.

```
corr=df.corr()
corr

<ipython-input-15-7d5195e2bf4d>: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=df.corr()

                   total  speeding   alcohol  not_distracted
no_previous  \
total           1.000000  0.611548  0.852613        0.827560
0.956179
speeding        0.611548  1.000000  0.669719        0.588010
0.571976
alcohol         0.852613  0.669719  1.000000        0.732816
0.783520
not_distracted  0.827560  0.588010  0.732816        1.000000
```

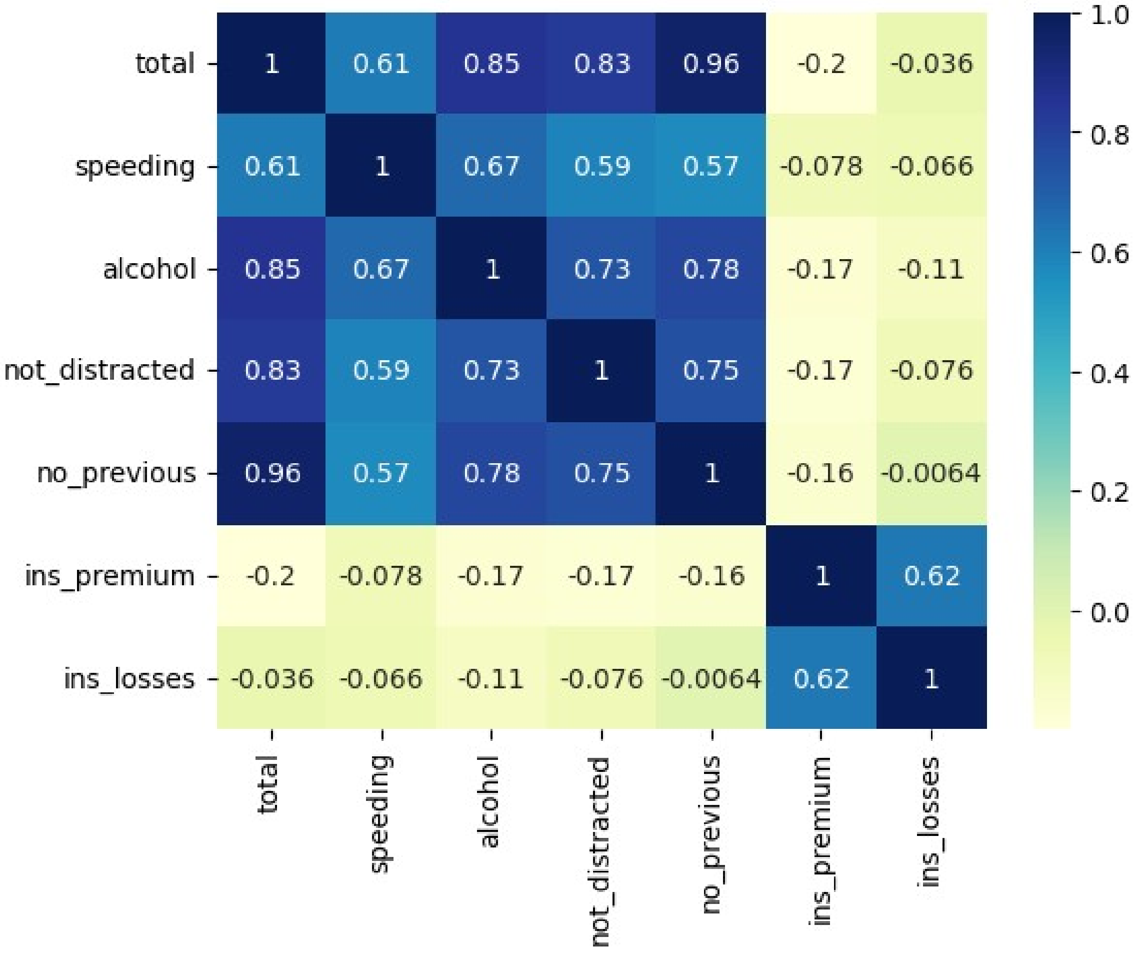
```
0.747307
no_previous      0.956179  0.571976  0.783520        0.747307
1.000000
ins_premium     -0.199702 -0.077675 -0.170612       -0.174856     -
0.156895
ins_losses      -0.036011 -0.065928 -0.112547       -0.075970     -
0.006359

                ins_premium  ins_losses
total             -0.199702   -0.036011
speeding          -0.077675   -0.065928
alcohol           -0.170612   -0.112547
not_distracted    -0.174856   -0.075970
no_previous       -0.156895   -0.006359
ins_premium        1.000000    0.623116
ins_losses         0.623116    1.000000
```

INFERENCE

This code calculates the correlation matrix for the columns in the DataFrame df using the .corr() method. The resulting corr DataFrame contains correlation coefficients between all pairs of numeric columns in df. Each value in the matrix represents the degree and direction of linear correlation between two variables, with values ranging from -1 (perfect negative correlation) to 1 (perfect positive correlation), and 0 indicating no correlation. This matrix can be used to explore relationships and dependencies between variables in the dataset.

```
 sns.heatmap(corr,annot=True,cmap="YlGnBu")
```

```
<Axes: >
```



INFERENCE

This code uses Seaborn to create a heatmap visualization of the correlation matrix (corr) previously calculated. The sns.heatmap function displays the correlations between pairs of numeric variables in a color-coded format. The annot=True argument adds numerical values to each cell of the heatmap, making it easier to interpret the correlation coefficients. The cmap="YlGnBu" argument sets the color map to use for the heatmap, with "YlGnBu" representing shades of yellow, green, and blue. This heatmap provides a visual representation of how strongly variables are correlated in the dataset, with warmer colors indicating stronger positive correlations, cooler colors indicating stronger negative correlations, and shades of green indicating weaker correlations.