In [90]:
```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [91]:
```
p=sns.load_dataset('car_crashes')
p
```

Out[91]:

	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

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses	abbrev
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 [92]:

```
p.shape

#Inference:From above code gives number of columns and rows present in datas
```

Out[92]:

```
(51, 8)
```

In [93]:

```
p.info()

#Inference : it shows number of rows and provides basic description of rows
```

```
<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 [94]:

```
p.describe()

#Inference:It describes as percentile how many accidents took per quantile a
```

Out[94]:

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_los
count	51.000000	51.000000	51.000000	51.000000	51.000000	51.000000	51.000
mean	15.790196	4.998196	4.886784	13.573176	14.004882	886.957647	134.493
std	4.122002	2.017747	1.729133	4.508977	3.764672	178.296285	24.835
min	5.900000	1.792000	1.593000	1.760000	5.900000	641.960000	82.750
25%	12.750000	3.766500	3.894000	10.478000	11.348000	768.430000	114.645
50%	15.600000	4.608000	4.554000	13.857000	13.775000	858.970000	136.050
75%	18.500000	6.439000	5.604000	16.140000	16.755000	1007.945000	151.870
max	23.900000	9.450000	10.038000	23.661000	21.280000	1301.520000	194.780

In [95]:

```
p.head()

#Inference:It gives first 5 cases from the dataset carcrashes
```

Out[95]:

	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 [96]:

```
p.tail()

#Inference:It gives last 5 cases from the dataset carcrashes
```

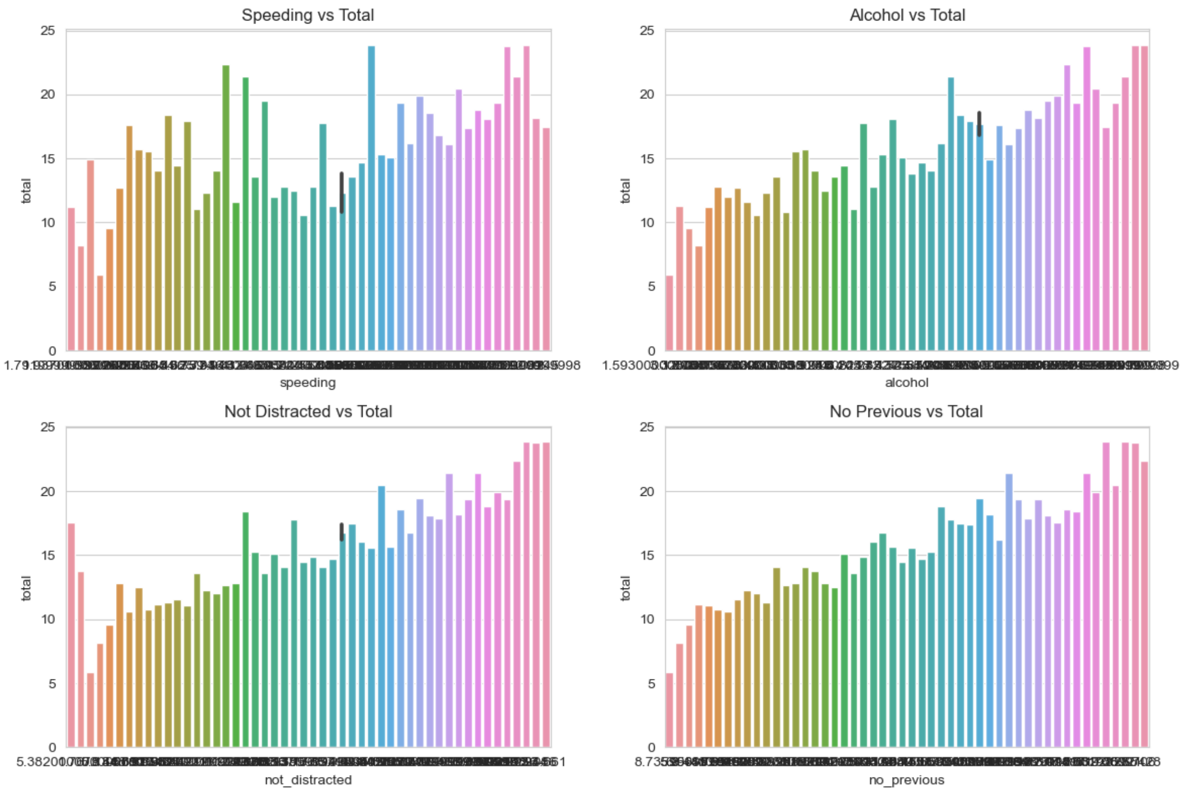
Out[96]:

	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 [97]:

```
fig, axes = plt.subplots(nrows=2, ncols=2, figsize=(12, 8))

# Create barplots for each column
sns.barplot(data=p, x='speeding', y='total', ax=axes[0, 0])
axes[0, 0].set_title('Speeding vs Total')

sns.barplot(data=p, x='alcohol', y='total', ax=axes[0, 1])
axes[0, 1].set_title('Alcohol vs Total')

sns.barplot(data=p, x='not_distracted', y='total', ax=axes[1, 0])
axes[1, 0].set_title('Not Distracted vs Total')

sns.barplot(data=p, x='no_previous', y='total', ax=axes[1, 1])
axes[1, 1].set_title('No Previous vs Total')

# Adjust layout
```

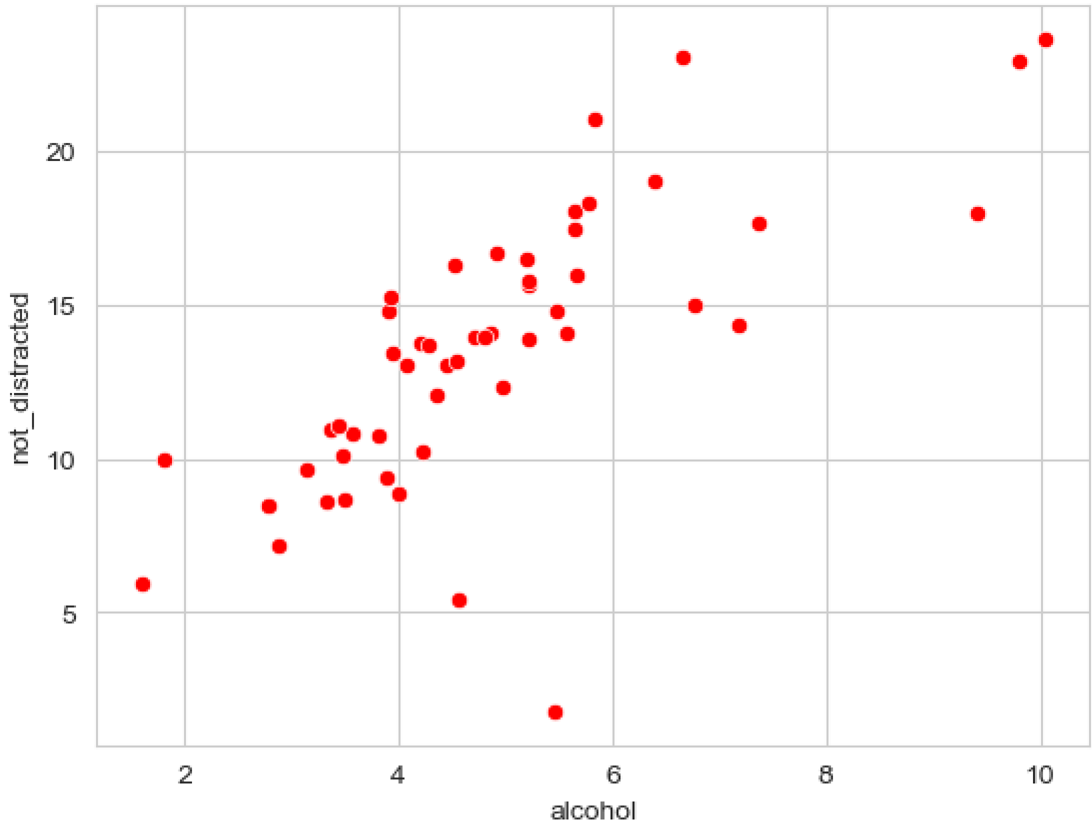
```
plt.tight_layout()

# Show the plots
plt.show()

#Inference:below 4 graphs relate accidents more alcohol intake causes more a
#who have records of previous accidents were increasing.Here due to being a
#cannot be drawn in one single graph.Here we imported matplotlib.pyplot and
#got graphs of all dependent variables
```



In [98]:

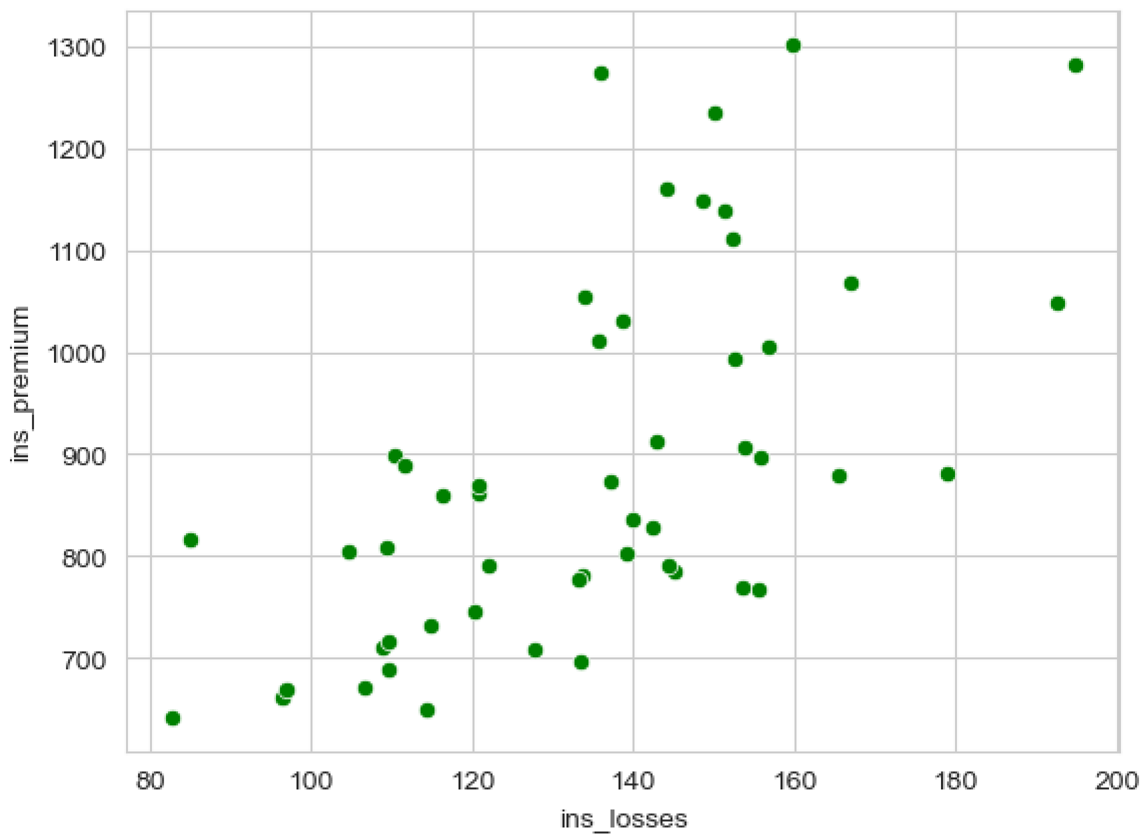
```
sns.scatterplot(x="alcohol",y="not_distracted",data=p,color="red")

#Inference: Above scatterplot describes as speed increases number of acciden
#and on average accidents were causing at position between region of 4-6
```

Out[98]: <Axes: xlabel='alcohol', ylabel='not_distracted'>



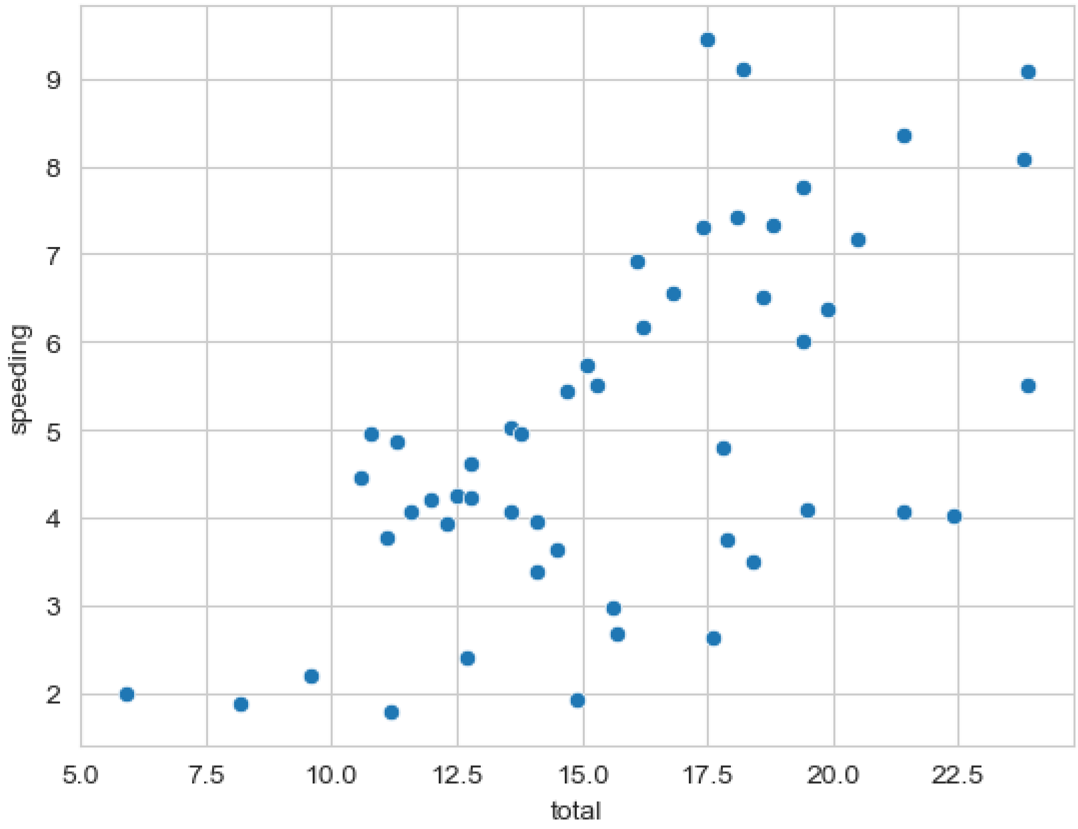
In [99]:
```
sns.scatterplot(x="ins_losses",y="ins_premium",data=p,color="green")
```

Out[99]: <Axes: xlabel='ins_losses', ylabel='ins_premium'>



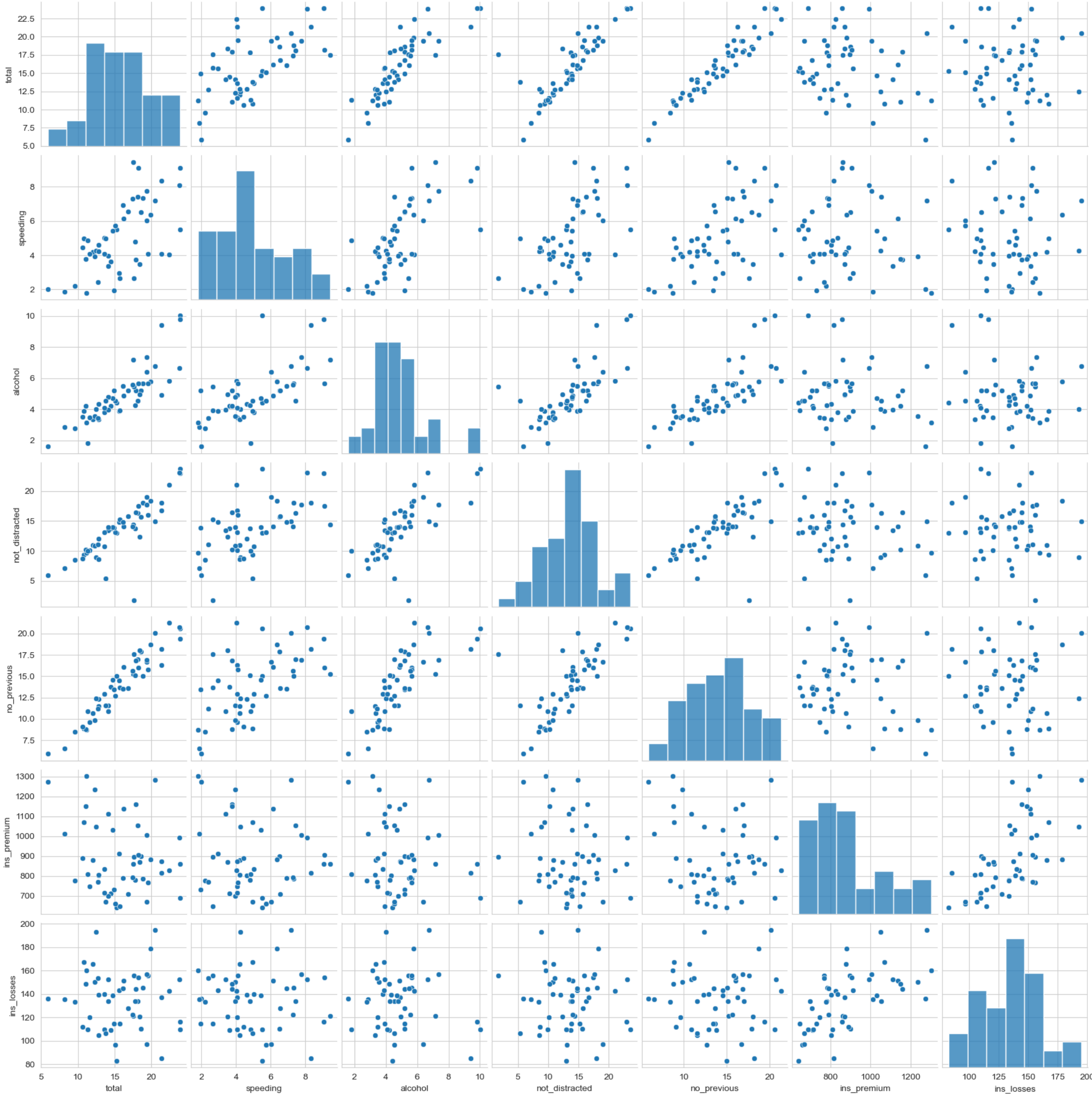
In [100…
```
sns.scatterplot(x='total',y='speeding',data=p)
```

Out[100]: <Axes: xlabel='total', ylabel='speeding'>



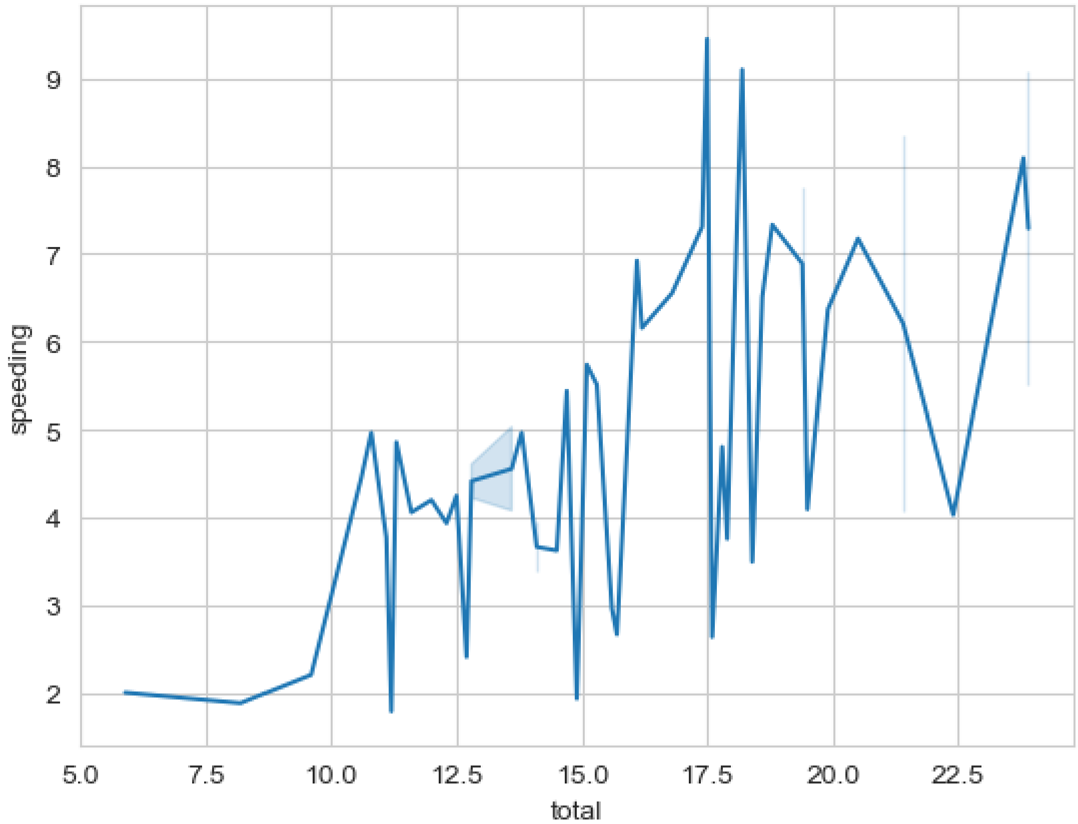
In [101…

```
sns.pairplot(p)
plt.show()

#Inference:Gives basic representation of all graph tells relation on how the
```



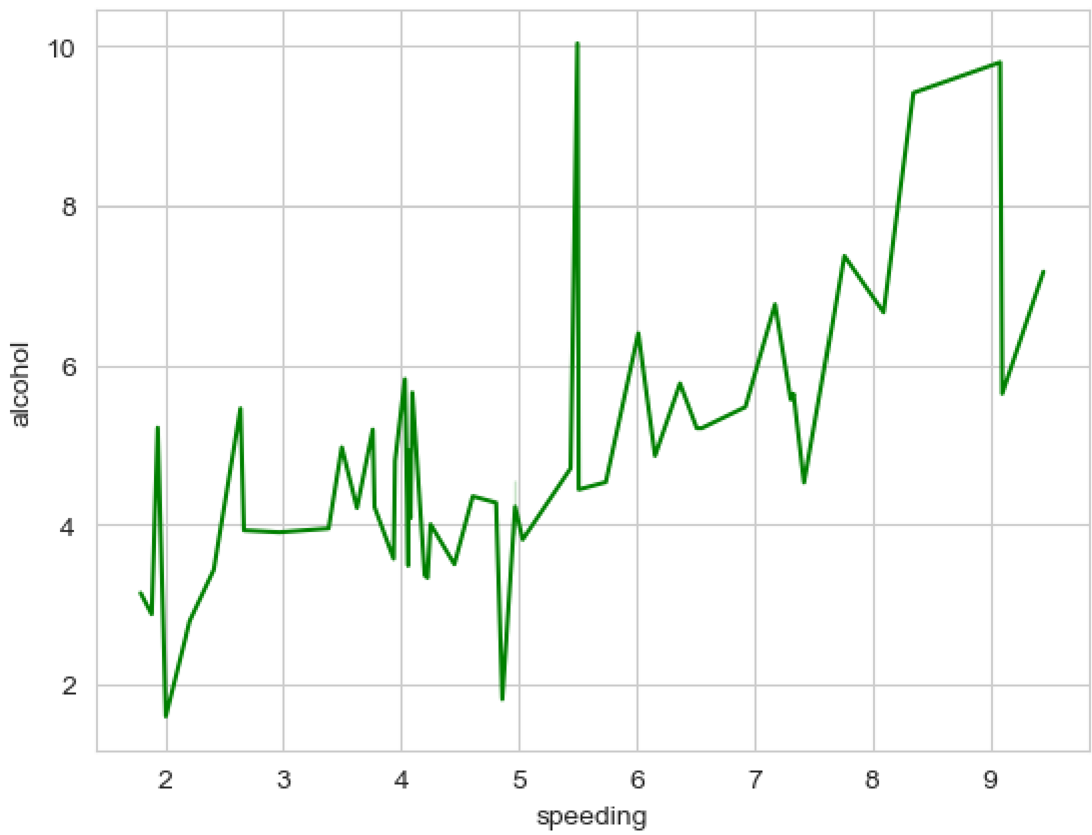
In [102…
```
sns.lineplot(x='total',y='speeding',data=p)


#Infernce:On average speeding id directly proportional to car crashes but th
#middle than at end where total is 17.5 where speeding is greater than 9

#Inference: From below lineplot and scatterplot concludes that speeding is m
```

Out[102]: <Axes: xlabel='total', ylabel='speeding'>



In [103…

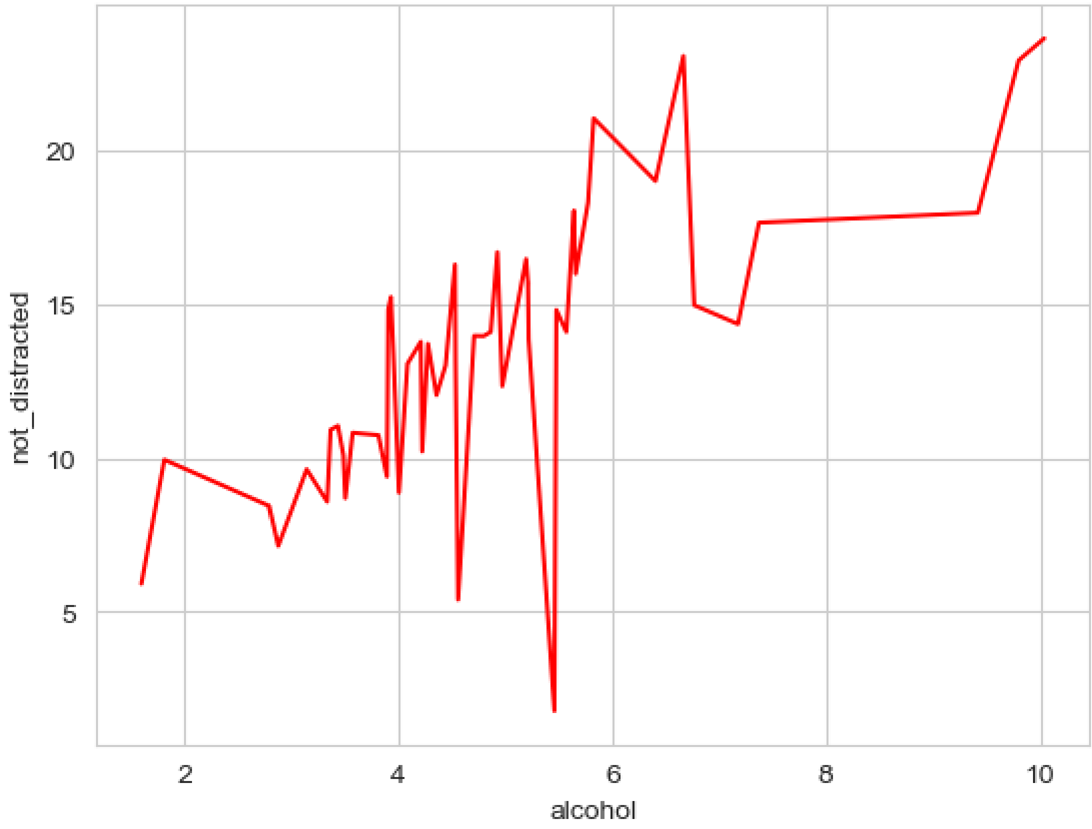
```
#lineplot
a=sns.lineplot(x="speeding",y="alcohol",data=p,color="green")

#inference is that when alcohol increases speed also increases
```



In [104…

```
sns.lineplot(x="alcohol",y="not_distracted",data=p,color="red")

#Inference: On average,there is less levels of distraction the levels of not
```

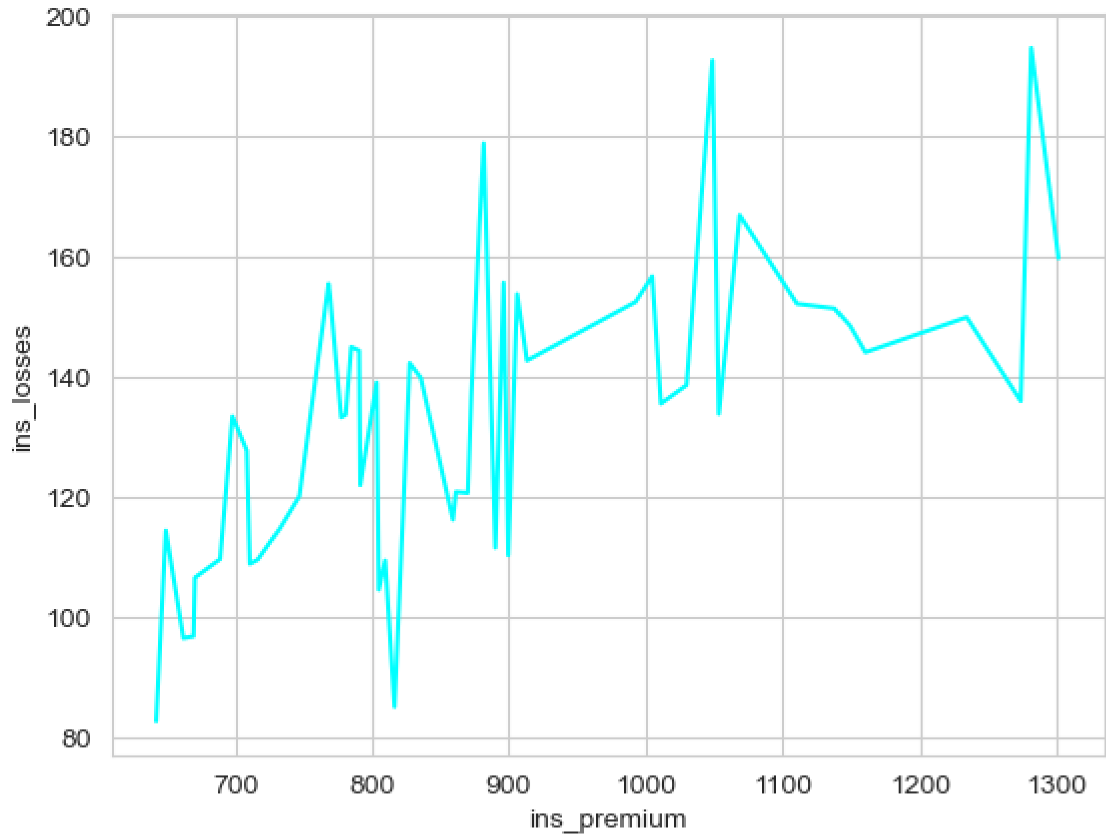
Out[104]: <Axes: xlabel='alcohol', ylabel='not_distracted'>



In [105…

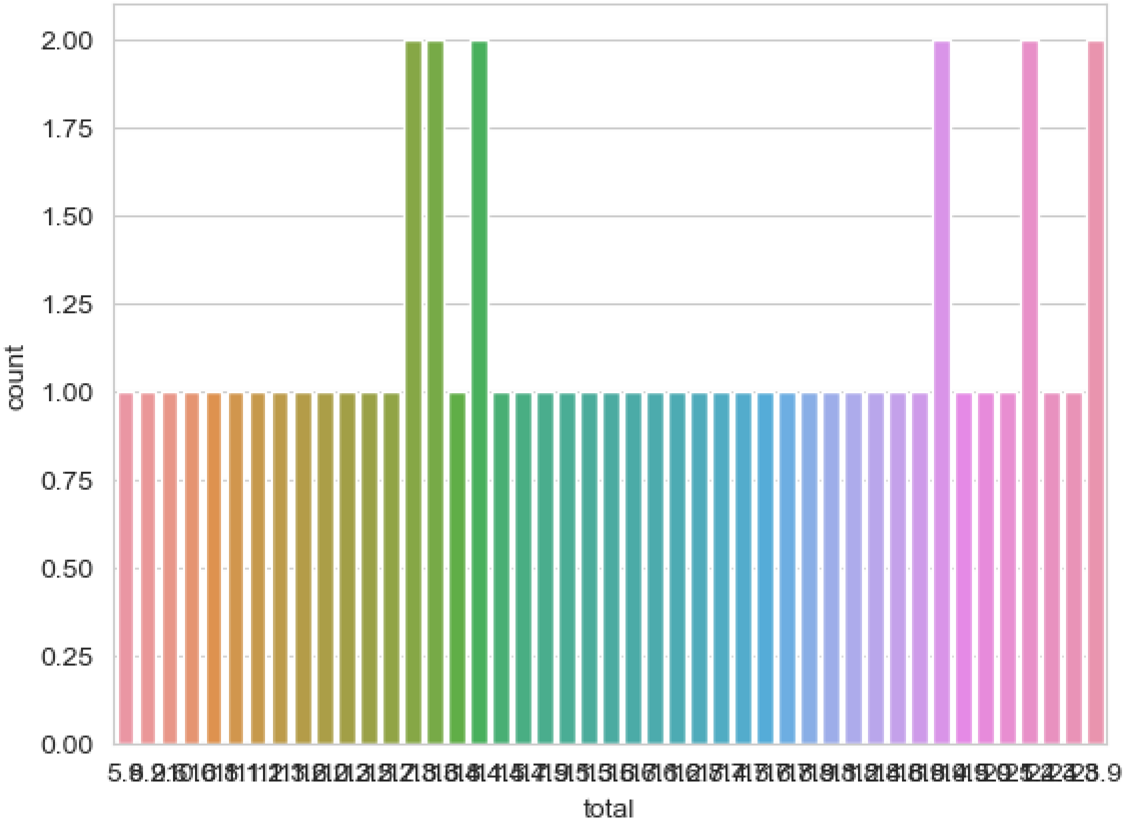
```
sns.lineplot(x="ins_premium",y="ins_losses",data=p,color="cyan")

#Inference: On average, Premium increases ideally losses decreases
```

Out[105]: <Axes: xlabel='ins_premium', ylabel='ins_losses'>



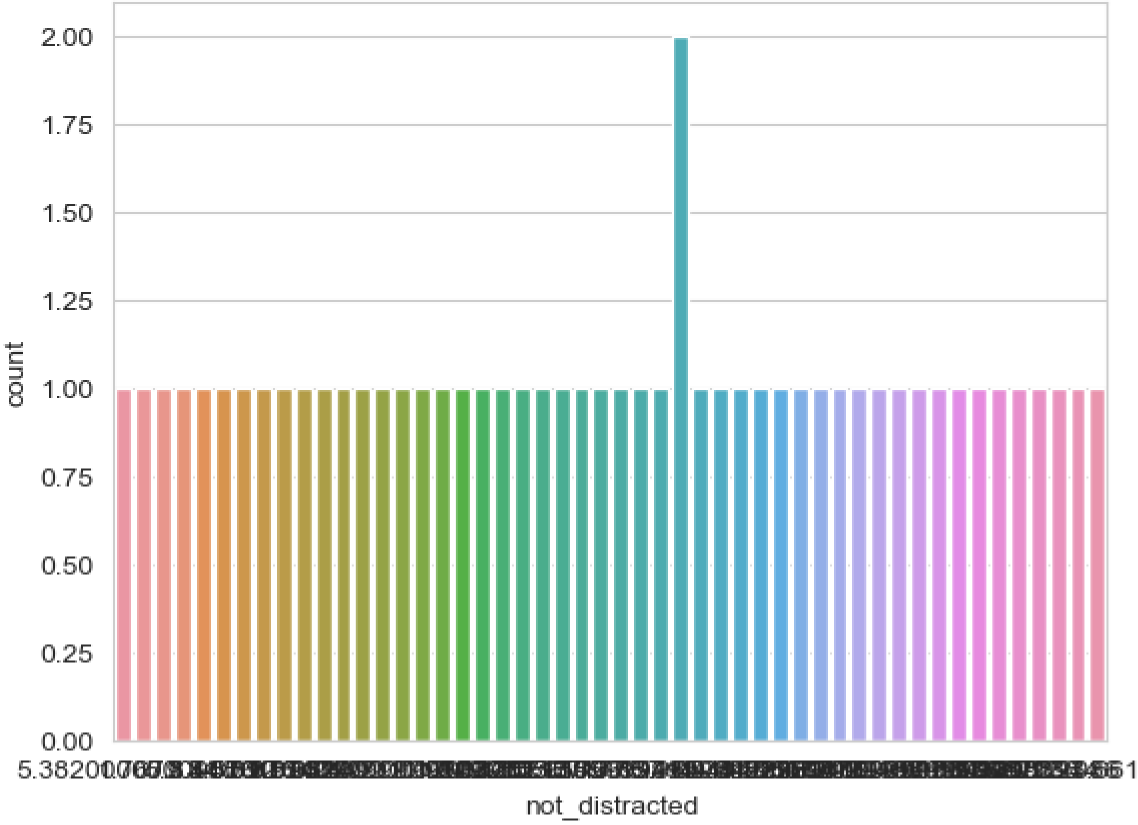
In [106…

```
sns.set_style("whitegrid")
sns.countplot(x="total", data=p)
plt.show()

#Inference is this gives the total count of car crashes
```



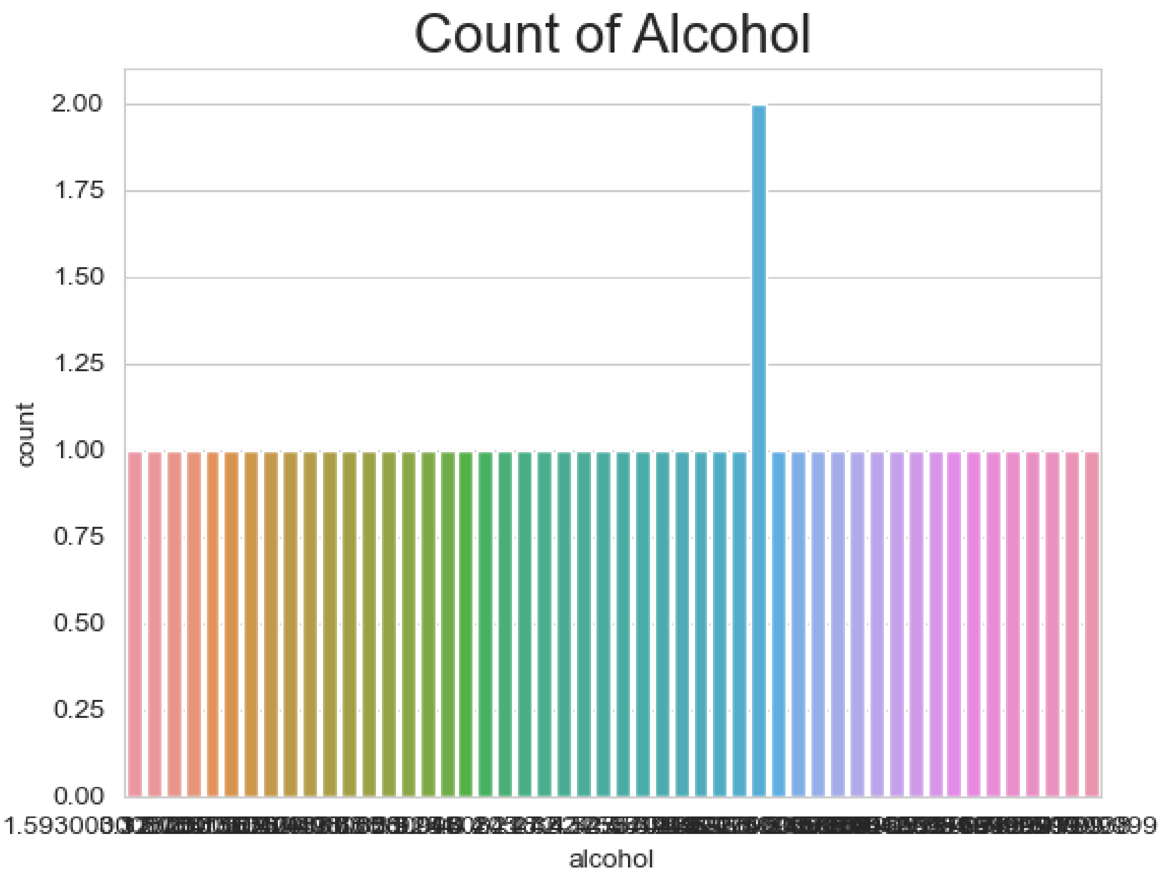
In [107…

```
sns.set_style("whitegrid")
sns.countplot(x="not_distracted", data=p)
plt.show()

#Inference is this gives the total count of not_distracted car crashes
```

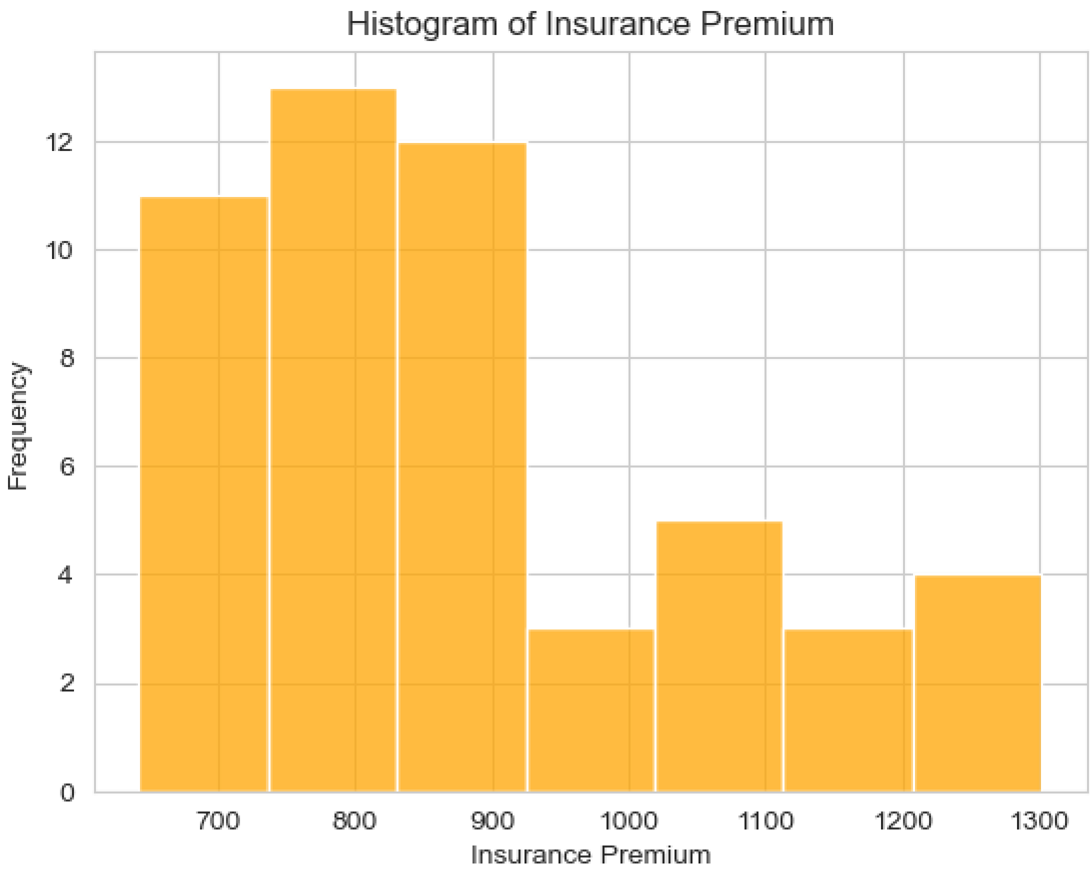


In [108…

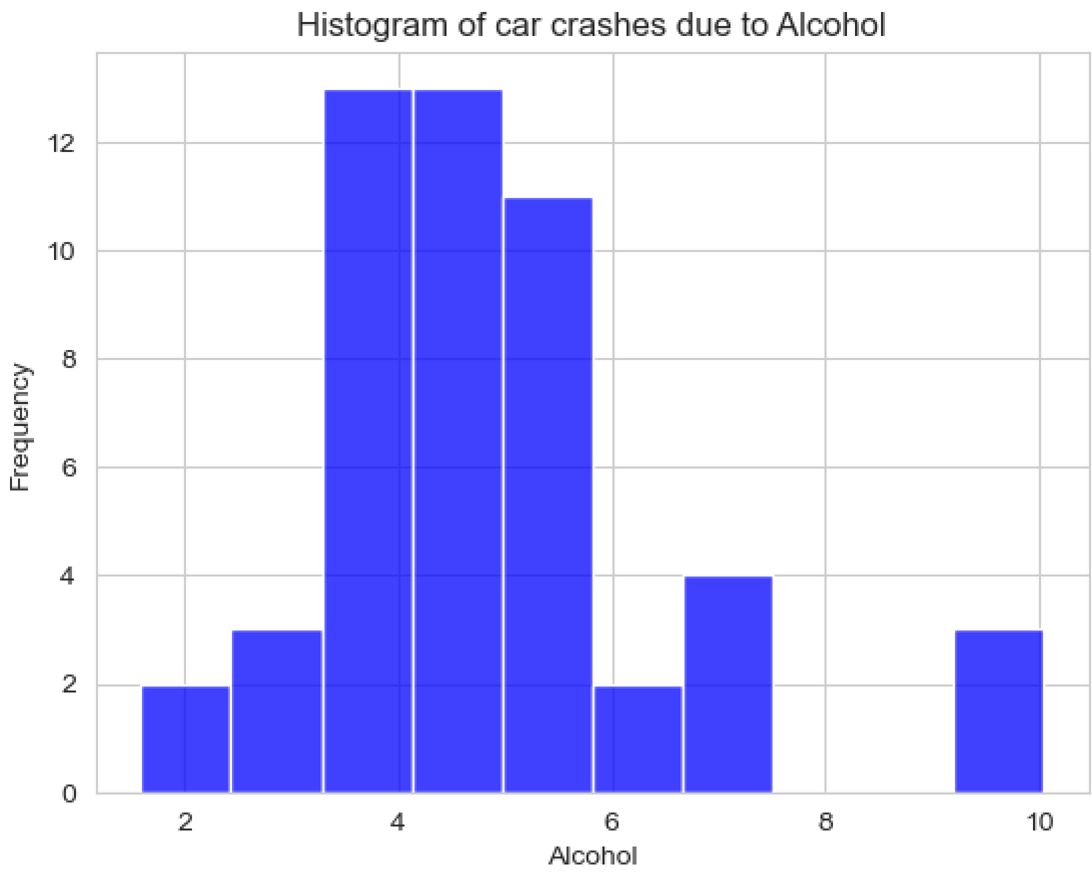
```
sns.set_style("whitegrid")
sns.countplot(x="alcohol", data=p)
plt.title("Count of Alcohol",fontsize=20)
plt.show()

#Inference is the count of alcohol is constant
```



In [109…
```
sns.histplot(p['ins_premium'], color='orange')
plt.xlabel('Insurance Premium')
plt.ylabel('Frequency')
plt.title('Histogram of Insurance Premium')
plt.show()

#Inference is for premium of 800 it has high frequency
```

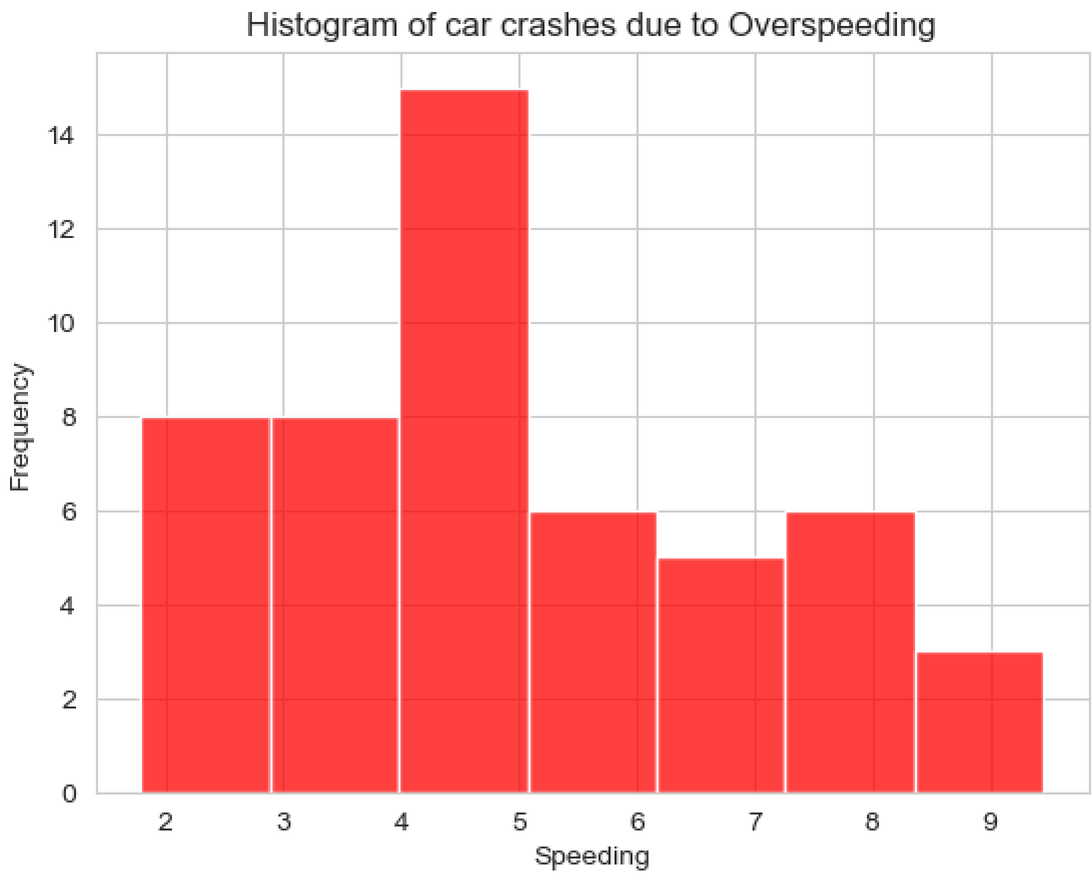


In [110…
```
sns.histplot(p['alcohol'], color='blue')
plt.xlabel('Alcohol')
plt.ylabel('Frequency')
plt.title('Histogram of car crashes due to Alcohol')
plt.show()

#Inference is for alcohol of 4 has highest frequency
```



In [111…

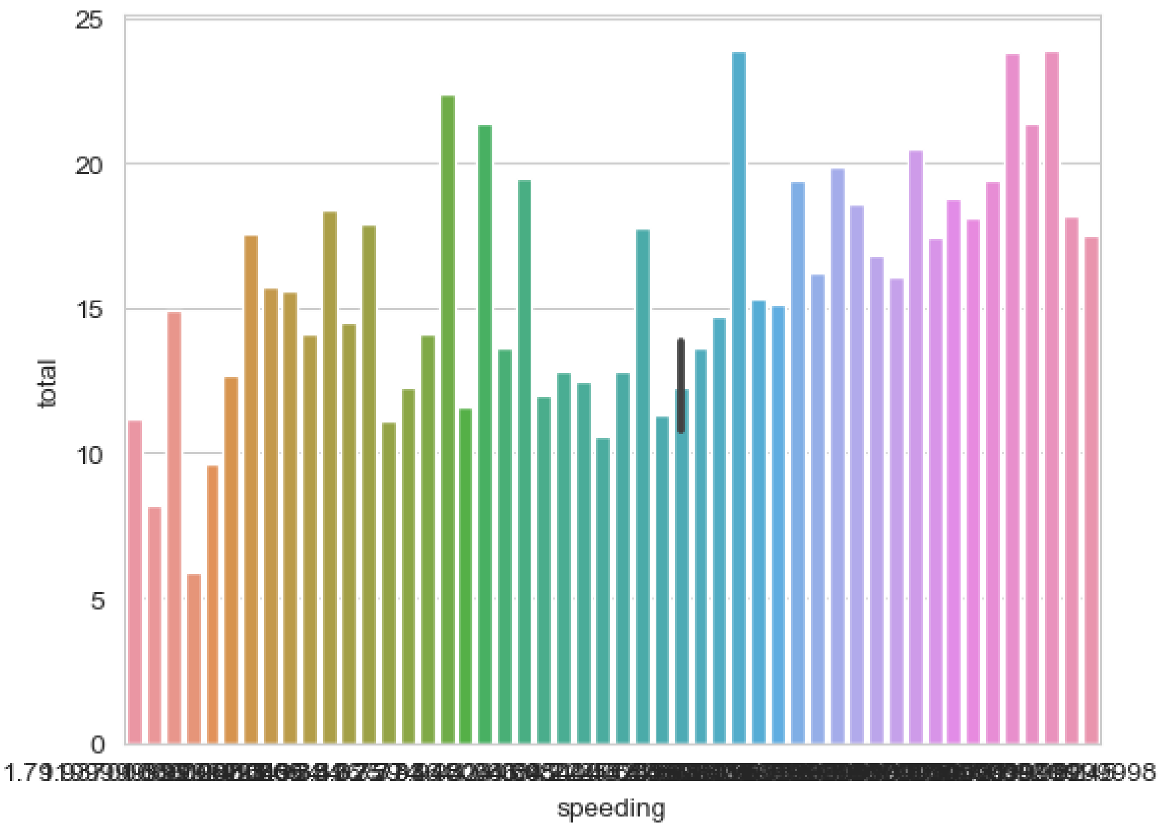
```
sns.histplot(p['speeding'], color='red')
plt.xlabel('Speeding')
plt.ylabel('Frequency')
plt.title('Histogram of car crashes due to Overspeeding')
plt.show()

#Inference is for speeding of 4-5 has highest frequency
```



In [112…

```
sns.barplot(y=p['total'],x=p['speeding'],data=p)
```

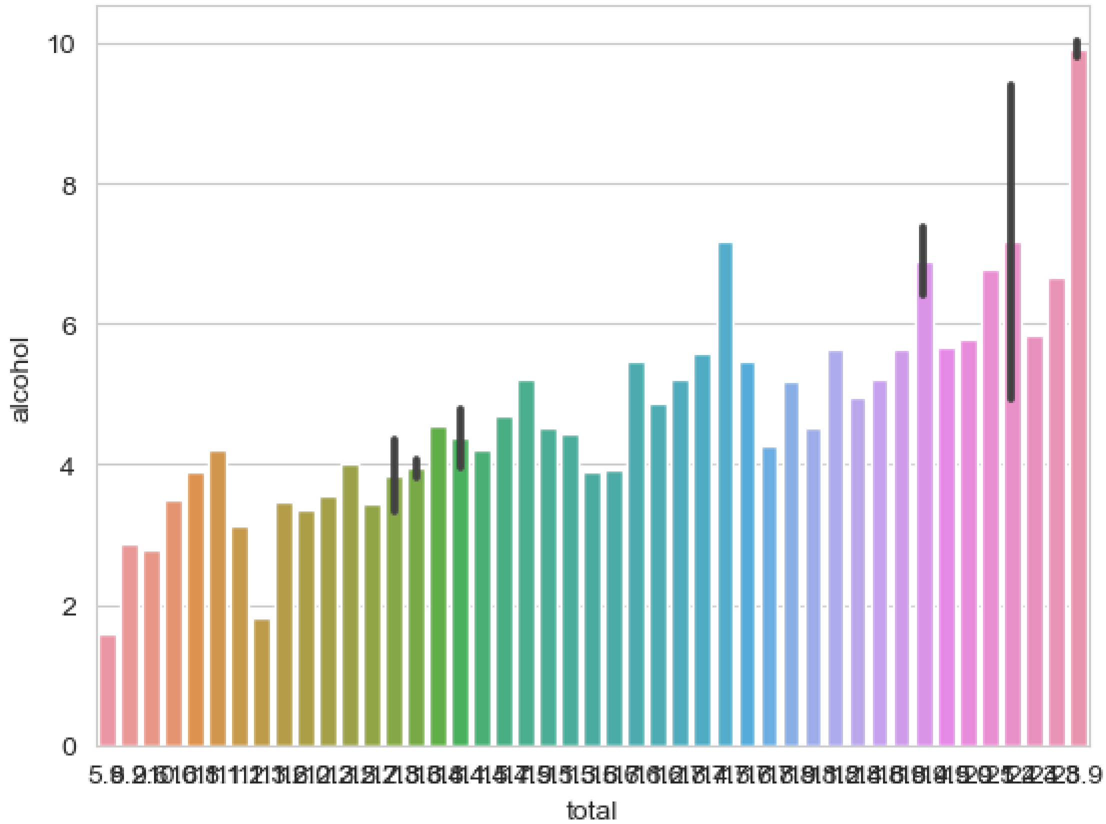
Out[112]: <Axes: xlabel='speeding', ylabel='total'>



In [113…

```
sns.barplot(y=p['alcohol'],x=p['total'],data=p)
```

Out[113]: <Axes: xlabel='total', ylabel='alcohol'>

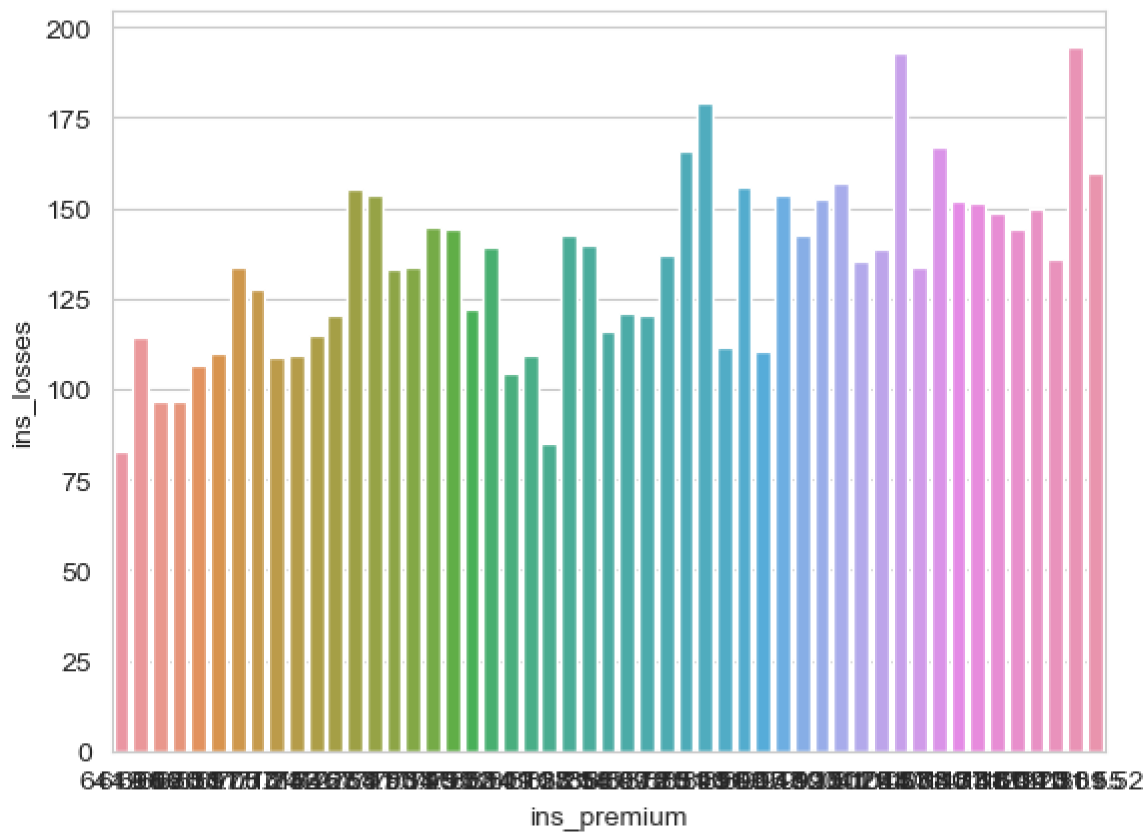


In [114…

```
sns.barplot(y=p['ins_losses'],x=p['ins_premium'],data=p)
```

Out[114]:

```
<Axes: xlabel='ins_premium', ylabel='ins_losses'>
```



In [115…

```
a=p.corr()
a

#Inference: It gives data from the region of -1 to 1 where greater than 0
#can be considered as positively correlated and less than 0 are considered
```

```
#as neagtively corelated.From above premium insurance and intial loses are
#independent variables so they were negatively correlated.Speeding and alcoh
#are high positively correlated and not_distracted attribute is positively c
```

```
/tmp/ipykernel_60381/3939962587.py:1: FutureWarning: The default value of nu
meric_only in DataFrame.corr is deprecated. In a future version, it will def
ault to False. Select only valid columns or specify the value of numeric_onl
y to silence this warning.
  a=p.corr()
```

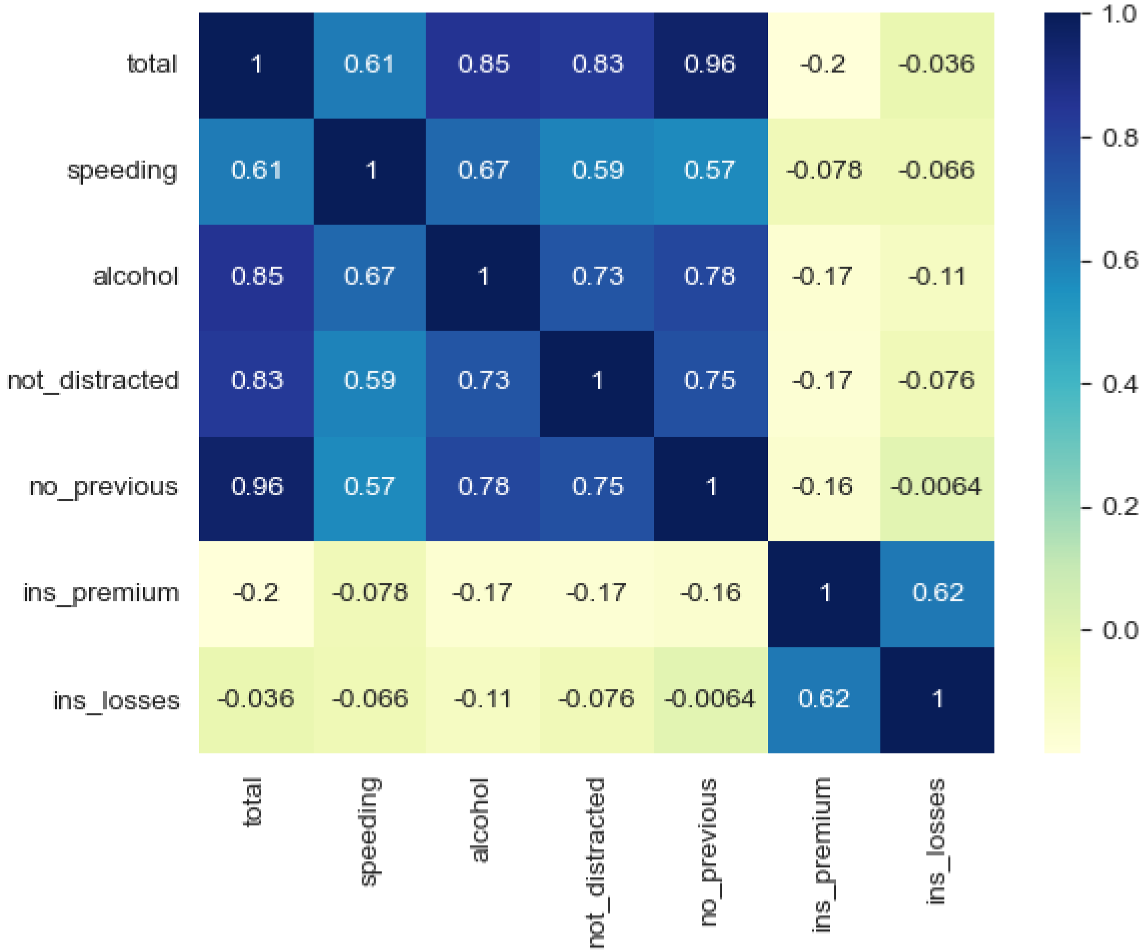
Out[115]:

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

In [117…

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

#Inference: In below heatmap blue indicates extreme values which are positiv
#correlated and green represents negatively correlated.We can get carcrashes
#more precisely like higher the speeding there is a chance of more likely to
#have accident.It also tells alcohol intake and carcrashes are directly prop
#It also tells where values are drivers are not distracted but had carcrash,
#also tells insurance premium are not involved ,similarily losses weren't in
#in the similar way.In this extreme values can be seen in dark blue and mini
#are seen in light green
```

Out[117]: <Axes: >


total
speeding
alcohol
not_distracted
no_previous
ins_premium
ins_losses
1 0.61 0.85 0.83 0.96 -0.2 -0.036
0.61 1 0.67 0.59 0.57 -0.078 -0.066
0.85 0.67 1 0.73 0.78 -0.17 -0.11
0.83 0.59 0.73 1 0.75 -0.17 -0.076
0.96 0.57 0.78 0.75 1 -0.16 -0.0064
-0.2 -0.078 -0.17 -0.17 -0.16 1 0.62
-0.036 -0.066 -0.11 -0.076 -0.0064 0.62 1
1.0
0.8
0.6
0.4
0.2
0.0

In [ ]: