In [1]:

```python
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
import numpy as np
import datetime as dt
sns.set_palette('colorblind')
```

Outcomes Analysis - Elliot Linsey 210764484

This notebook performs a small amount of exploratory data analysis, then calculates probabilities for each crime outcome.

In [3]:

```python
df = pd.read_csv('D:\\EOY Datasets\\Full Datasets'+'\\'+'Everything_cleaned.csv',index_col=0)
#df = pd.read_csv('Everything_cleaned.csv')
```

```
C:\Users\ellio\anaconda3\lib\site-packages\numpy\lib\arraysetops.py:583: FutureWarning: elementwise comparison failed; returning scalar instead, but in the future will perform elementwise comparison
  mask |= (ar1 == a)
```

In [4]:

```python
df
```

Out[4]:

	Crime_ID	Month	Reported_by	Crime_type	Outcome_type	Year	Level_of_Lockdown	Reg
0	e9a50727f2189e0c50f704e3661bc1a8ae3a39aece4866...	2019-01-01	Avon and Somerset	Violence and sexual offences	No suspect identified	2019	No Lockdown	So W
1	fb06f54e9d633a961109ef74171e4beb40b2fe44e57a10...	2019-01-01	Avon and Somerset	Violence and sexual offences	Unable to prosecute suspect	2019	No Lockdown	So W
2	4655ba031d5c8d00b247577a22472fbb8f9130d98b7d95...	2019-01-01	Avon and Somerset	Violence and sexual offences	No suspect identified	2019	No Lockdown	So W
3	2830f28a4a93138717182d1c25b5d677cfaf3c27398934...	2019-01-01	Avon and Somerset	Theft offences	No suspect identified	2019	No Lockdown	So W
4	ca01d599d45f6e1d1bc5b05ddc6a1b5559870fc831baa8e...	2019-01-01	Avon and Somerset	Theft offences	No suspect identified	2019	No Lockdown	So W
...	...	...	...	...	...	...	...	...
10297558	b81d4240a922e62a8ab5c7563a890f53503bbd9a72d30f...	2021-12-01	Wiltshire	Theft offences	No suspect identified	2021	Medium	So W
10297559	8936ed88bc946d71ea9e1ba7bf545d335aca64a2aad0c2...	2021-12-01	Wiltshire	Theft offences	No suspect identified	2021	Medium	So W
10297560	ec0357d8210b3df52277002d3a0f5967bf78d26d572583...	2021-12-01	Wiltshire	Theft offences	No suspect identified	2021	Medium	So W
10297561	8af4abefb399d3a2c1c71eec2045f3d4f9898c2ba97001...	2021-12-01	Wiltshire	Theft offences	No suspect identified	2021	Medium	So W
10297562	c841195e1663055c4167d3ed9297003aed51d6...	2021-12-01	Wiltshire	Theft offences	No suspect identified	2021	Medium	So W

10175912 rows × 8 columns

Exploratory Data Analysis

In [5]:

```python
len(df.Reported_by.value_counts())
```

Out[5]:

```
42
```

In [6]:

```python
df.dtypes
```

Out[6]:

```
Crime_ID             object
Month                object
Reported_by          object
Crime_type           object
Outcome_type         object
Year                  int64
Level_of_Lockdown    object
Region               object
dtype: object
```

In [7]:

```python
grouped = df.groupby(['Year','Outcome_type'])['Crime_ID'].count().reset_index()
```

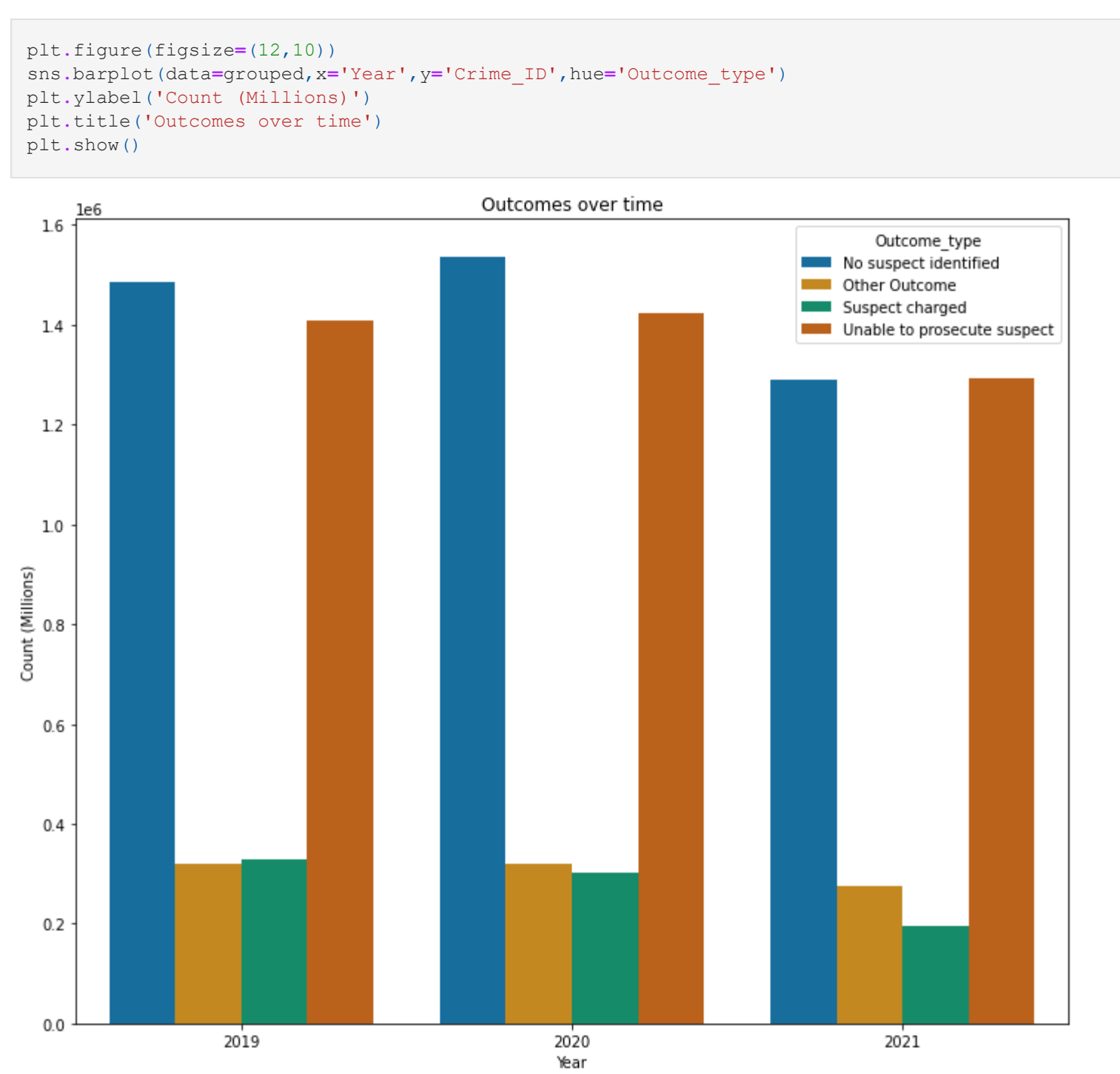
In [8]:

```python
plt.rcParams['font.size'] = '16'
```

These next graphs are investigating the raw counts for outcomes by year, outcomes by crime type and crime type by year.

In [9]:

```python
plt.figure(figsize=(12,10))
sns.barplot(data=grouped,x='Year',y='Crime_ID',hue='Outcome_type')
plt.ylabel('Count (Millions)')
plt.title('Outcomes over time')
plt.show()
```

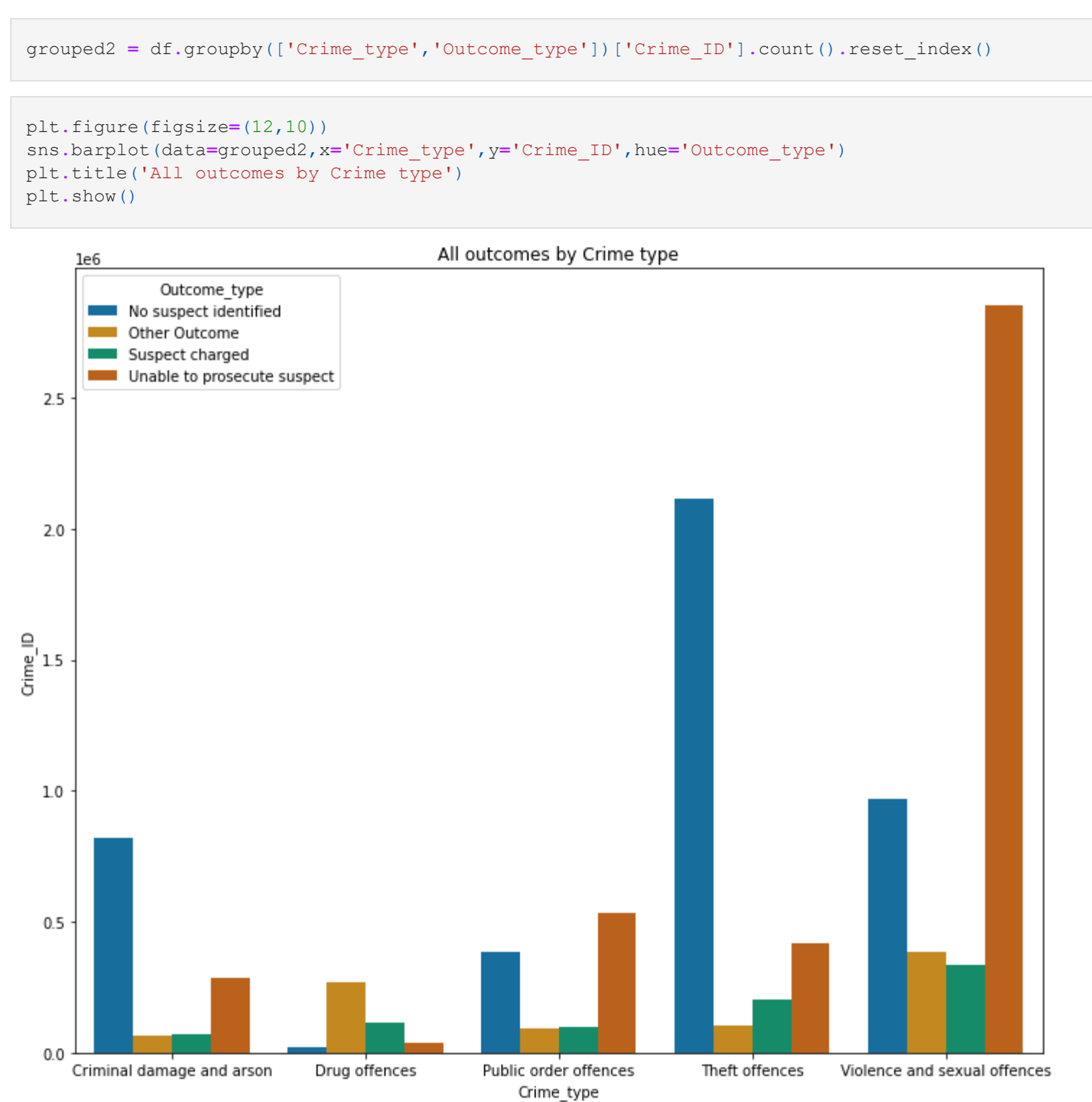


In [10]:

```python
grouped2 = df.groupby(['Crime_type','Outcome_type'])['Crime_ID'].count().reset_index()
```

In [11]:

```python
plt.figure(figsize=(12,10))
sns.barplot(data=grouped2,x='Crime_type',y='Crime_ID',hue='Outcome_type')
plt.ylabel('Count (Millions)')
plt.title('All outcomes by Crime type')
plt.show()
```



In [12]:

```python
grouped3 = df.groupby(['Year','Crime_type'])['Crime_ID'].count().reset_index()
```

In [13]:

```python
plt.figure(figsize=(12,10))
ax = sns.barplot(data=grouped3,x='Year',y='Crime_ID',hue='Crime_type')
for container in ax.containers:
    ax.bar_label(container)
plt.ylabel('Count (Millions)')
plt.legend()
#plt.legend(loc='center left',bbox_to_anchor=(1.0, 0.5))
plt.title('All Crime types by Year')
plt.legend(prop={'size': 10})
plt.savefig('Crime types by Year.png')
plt.show()
```

Probabilities of Outcomes

This involves creating cross tab tables then pivoting them to get them in the correct format for plotting.

In [14]:

```python
tab = pd.crosstab(df['Crime_type'],df['Outcome_type'],normalize=0)*100
```

In [15]:

```python
tab
```

Out[15]:

Outcome_type / Crime_type	No suspect identified	Other Outcome	Suspect charged	Unable to prosecute suspect
Criminal damage and arson	66.053033	5.315144	5.581935	23.049888
Drug offences	4.906264	60.537212	26.115081	8.441444
Public order offences	34.879873	8.269957	8.952953	47.897218
Theft offences	74.516240	3.628741	7.191615	14.663404
Violence and sexual offences	21.295684	8.504935	7.432865	62.766515

In [16]:

```python
grouped4 = df.groupby(['Year','Crime_type','Outcome_type'])['Crime_ID'].count().reset_index()
grouped4
```

Out[16]:

	Year	Crime_type	Outcome_type	Crime_ID
0	2019	Criminal damage and arson	No suspect identified	291371
1	2019	Criminal damage and arson	Other Outcome	24003
2	2019	Criminal damage and arson	Suspect charged	25998
3	2019	Criminal damage and arson	Unable to prosecute suspect	100564
4	2019	Drug offences	No suspect identified	6800
5	2019	Drug offences	Other Outcome	84881
6	2019	Drug offences	Suspect charged	44757
7	2019	Drug offences	Unable to prosecute suspect	13402
8	2019	Public order offences	No suspect identified	110839
9	2019	Public order offences	Other Outcome	29480
10	2019	Public order offences	Suspect charged	35363
11	2019	Public order offences	Unable to prosecute suspect	165243
12	2019	Theft offences	No suspect identified	780432
13	2019	Theft offences	Other Outcome	50071
14	2019	Theft offences	Suspect charged	94160
15	2019	Theft offences	Unable to prosecute suspect	170310
16	2019	Violence and sexual offences	No suspect identified	295211
17	2019	Violence and sexual offences	Other Outcome	132664
18	2019	Violence and sexual offences	Suspect charged	129288
19	2019	Violence and sexual offences	Unable to prosecute suspect	959445
20	2020	Criminal damage and arson	No suspect identified	280327
21	2020	Criminal damage and arson	Other Outcome	22361
22	2020	Criminal damage and arson	Suspect charged	25788
23	2020	Criminal damage and arson	Unable to prosecute suspect	98597
24	2020	Drug offences	No suspect identified	8924
25	2020	Drug offences	Other Outcome	102251
26	2020	Drug offences	Suspect charged	43813
27	2020	Drug offences	Unable to prosecute suspect	14192
28	2020	Public order offences	No suspect identified	139316
29	2020	Public order offences	Other Outcome	32250
30	2020	Public order offences	Suspect charged	37657
31	2020	Public order offences	Unable to prosecute suspect	182454
32	2020	Theft offences	No suspect identified	734742
33	2020	Theft offences	Other Outcome	30465
34	2020	Theft offences	Suspect charged	66027
35	2020	Theft offences	Unable to prosecute suspect	134004
36	2020	Violence and sexual offences	No suspect identified	372561
37	2020	Violence and sexual offences	Other Outcome	132177
38	2020	Violence and sexual offences	Suspect charged	127306
39	2020	Violence and sexual offences	Unable to prosecute suspect	992880
40	2021	Criminal damage and arson	No suspect identified	249040
41	2021	Criminal damage and arson	Other Outcome	19679
42	2021	Criminal damage and arson	Suspect charged	17572
43	2021	Criminal damage and arson	Unable to prosecute suspect	87244
44	2021	Drug offences	No suspect identified	5961
45	2021	Drug offences	Other Outcome	80434
46	2021	Drug offences	Suspect charged	26855
47	2021	Drug offences	Unable to prosecute suspect	9716
48	2021	Public order offences	No suspect identified	137254
49	2021	Public order offences	Other Outcome	30124
50	2021	Public order offences	Suspect charged	26420
51	2021	Public order offences	Unable to prosecute suspect	184295
52	2021	Theft offences	No suspect identified	599234
53	2021	Theft offences	Other Outcome	22430
54	2021	Theft offences	Suspect charged	43876
55	2021	Theft offences	Unable to prosecute suspect	111762
56	2021	Violence and sexual offences	No suspect identified	299728
57	2021	Violence and sexual offences	Other Outcome	121553
58	2021	Violence and sexual offences	Suspect charged	81094
59	2021	Violence and sexual offences	Unable to prosecute suspect	899267

In [17]:

```python
pivot = pd.pivot_table(grouped4,values='Crime_ID',columns=['Year','Crime_type'],index='Outcome_type')
```

In [18]:

```python
pivot
```

Out[18]:

Year	2019					2020					2021		
Crime_type / Outcome_type	Criminal damage and arson	Drug offences	Public order offences	Theft offences	Violence and sexual offences	Criminal damage and arson	Drug offences	Public order offences	Theft offences	Violence and sexual offences	Criminal damage and arson	Drug offences	Public order offences
No suspect identified	291371	6800	110839	780432	295211	280327	8924	139316	734742	372561	249040	5961	137254
Other Outcome	24003	84881	29480	50071	132664	22361	102251	32250	30465	132177	19679	80434	30124
Suspect charged	25998	44757	35363	94160	129288	25788	43813	37657	66027	127306	17572	26855	26420
Unable to prosecute suspect	100564	13402	165243	170310	959445	98597	14192	182454	134004	992880	87244	9716	184295

Normalised by column, e.g in 2019, for every criminal damage case 66% had no suspect identified, 5% had an other outcome etc.

In [19]:

```python
pivot2 = pivot/pivot.sum()
pivot2 = pivot2.applymap(lambda x: "{0:.0f}%".format(100*x))
pivot2
```

Out[19]:

Year	2019					2020					2021		
Crime_type / Outcome_type	Criminal damage and arson	Drug offences	Public order offences	Theft offences	Violence and sexual offences	Criminal damage and arson	Drug offences	Public order offences	Theft offences	Violence and sexual offences	Criminal damage and arson	Drug offences	Public order offences
No suspect identified	66%	5%	33%	71%	19%	66%	5%	36%	76%	23%	67%	5%	36%
Other Outcome	5%	57%	9%	5%	9%	5%	60%	8%	3%	8%	5%	65%	8%
Suspect charged	6%	30%	10%	9%	9%	6%	26%	10%	7%	8%	5%	22%	7%
Unable to prosecute suspect	23%	9%	48%	16%	63%	23%	8%	47%	14%	61%	23%	8%	49%

In [20]:

```python
import dataframe_image as dfi
```

In [21]:

```python
#dfi.export(pivot2,"mytable.png")
```

In [22]:

```python
pivot3 = pivot/pivot.sum()*100
pivot3
```

Out[22]:

Year	2019					2020						
Crime_type / Outcome_type	Criminal damage and arson	Drug offences	Public order offences	Theft offences	Violence and sexual offences	Criminal damage and arson	Drug offences	Public order offences	Theft offences	Violence and sexual offences	Criminal damage and arson	Dr offen
No suspect identified	65.930587	4.538174	32.511256	71.274086	19.465214	65.639130	5.274855	35.569104	76.120294	22.927903	66.671129	4.847
Other Outcome	5.431329	56.647758	8.647063	4.572807	8.747415	5.235873	60.439177	8.233825	3.156216	8.134350	5.268315	65.411
Suspect charged	5.882752	29.869861	10.372663	8.599299	8.524813	6.038312	25.897269	9.614300	6.840489	7.834582	4.704245	21.839
Unable to prosecute suspect	22.755331	8.944207	48.469018	15.553808	63.262557	23.086685	8.388698	46.582771	13.883001	61.103165	23.356312	7.901

Testing the differences between using pd.pivot_table and pd.crosstab. In this case, both are sufficient.

In [23]:

```python
pivot4 = pd.pivot_table(grouped4,values='Crime_ID',columns=['Year','Outcome_type'],index='Crime_type')
pivot4
```

Out[23]:

Year	2019				2020				2021			
Outcome_type / Crime_type	No suspect identified	Other Outcome	Suspect charged	Unable to prosecute suspect	No suspect identified	Other Outcome	Suspect charged	Unable to prosecute suspect	No suspect identified	Other Outcome	Suspect charged	Unable to prosecute suspect
Criminal damage and arson	291371	24003	25998	100564	280327	22361	25788	98597	249040	19679	17572	87244
Drug offences	6800	84881	44757	13402	8924	102251	43813	14192	5961	80434	26855	9716
Public order offences	110839	29480	35363	165243	139316	32250	37657	182454	137254	30124	26420	184295
Theft offences	780432	50071	94160	170310	734742	30465	66027	134004	599234	22430	43876	111762
Violence and sexual offences	295211	132664	129288	959445	372561	132177	127306	992880	299728	121553	81094	899267

In [24]:

```python
pivot2019 = pivot4.loc[slice(None),(2019)]
#pivot2019 = pivot2019/pivot2019.sum()
pivot2019 = pivot2019.div(pivot2019.sum(axis=1), axis=0)
pivot2019
```

Out[24]:

Outcome_type / Crime_type	No suspect identified	Other Outcome	Suspect charged	Unable to prosecute suspect
Criminal damage and arson	0.659306	0.054313	0.058828	0.227553
Drug offences	0.045382	0.566478	0.298699	0.089442
Public order offences	0.325113	0.086471	0.103727	0.484690
Theft offences	0.712741	0.045728	0.085993	0.155538
Violence and sexual offences	0.194652	0.087474	0.085248	0.632626

In [25]:

```python
tab2 = pd.crosstab(columns=[df['Year'],df['Crime_type']],index=df['Outcome_type'])
tab2
```

Out[25]:

Year	2019					2020							
Crime_type / Outcome_type	Criminal damage and arson	Drug offences	Public order offences	Theft offences	Violence and sexual offences	Criminal damage and arson	Drug offences	Public order offences	Theft offences	Violence and sexual offences	Criminal damage and arson	Drug offences	Public order offences
No suspect identified	291371	6800	110839	780432	295211	280327	8924	139316	734742	372561	249040	5961	137254
Other Outcome	24003	84881	29480	50071	132664	22361	102251	32250	30465	132177	19679	80434	30124
Suspect charged	25998	44757	35363	94160	129288	25788	43813	37657	66027	127306	17572	26855	26420
Unable to prosecute suspect	100564	13402	165243	170310	959445	98597	14192	182454	134004	992880	87244	9716	184295

Creating a function that does the correct slicing and converting to probabilities.

In [26]:

```python
def pivoting_stacks(df,year):
    res = df.loc[slice(None),(year)]
    res = res.div(res.sum(axis=1), axis=0)
    return res
```

In [27]:

```python
pivot2019 = pivoting_stacks(pivot4,2019)
pivot2020 = pivoting_stacks(pivot4,2020)
pivot2021 = pivoting_stacks(pivot4,2021)
```

In [57]:

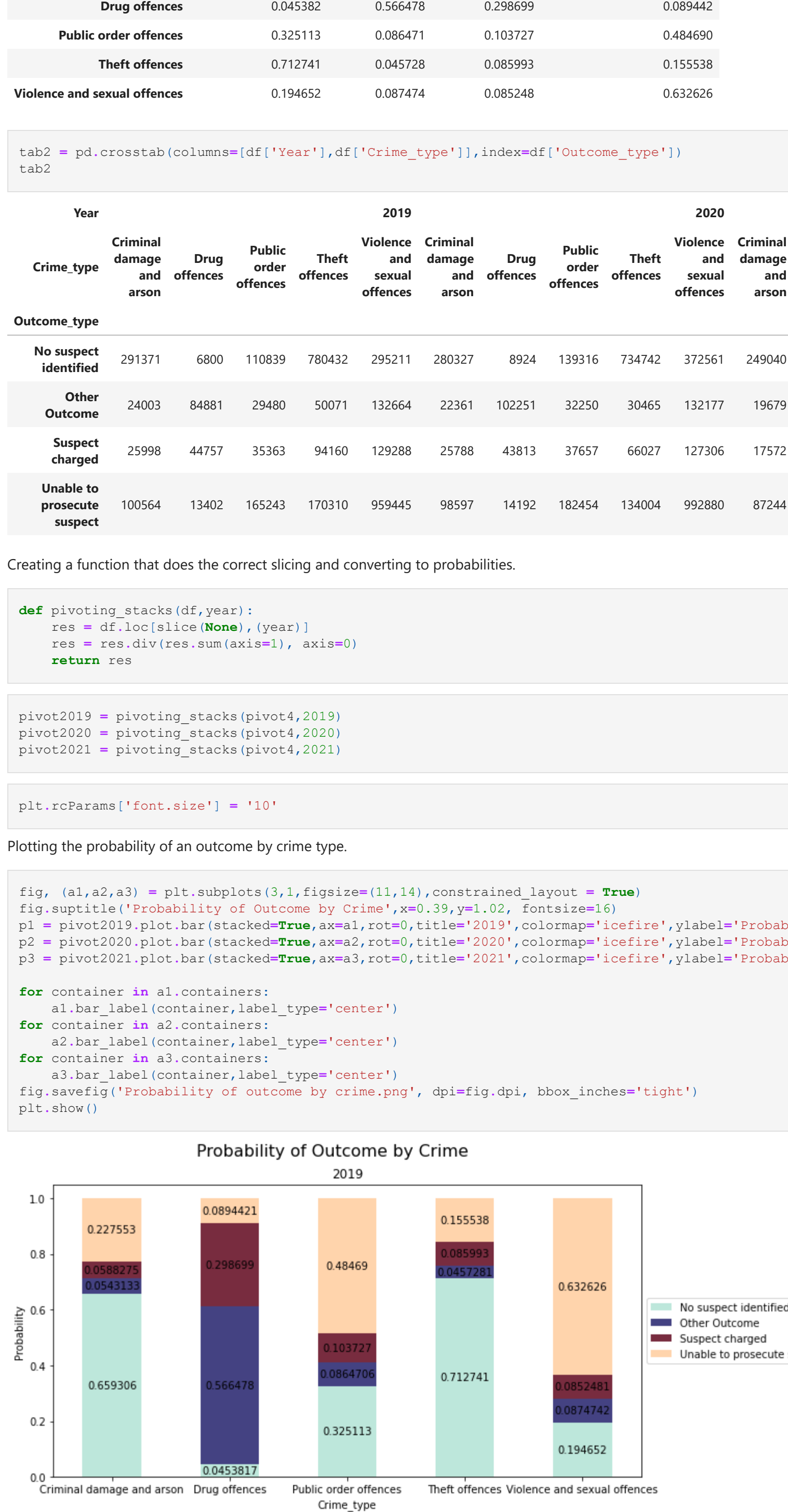
```python
plt.rcParams['font.size'] = '10'
```

Plotting the probability of an outcome by crime type.

In [62]:

```python
fig, (a1,a2,a3) = plt.subplots(3,1,figsize=(11,14),constrained_layout = True)
fig.suptitle('Probability of Outcome by Crime',x=0.39,y=1.02, fontsize=16)
p1 = pivot2019.plot.bar(stacked=True,ax=a1,rot=0,title='2019',colormap='icefire',ylabel='Probability').legend(l
p2 = pivot2020.plot.bar(stacked=True,ax=a2,rot=0,title='2020',colormap='icefire',ylabel='Probability').legend(l
p3 = pivot2021.plot.bar(stacked=True,ax=a3,rot=0,title='2021',colormap='icefire',ylabel='Probability').legend(l

for container in a1.containers:
    a1.bar_label(container,label_type='center')
for container in a2.containers:
    a2.bar_label(container,label_type='center')
for container in a3.containers:
    a3.bar_label(container,label_type='center')
fig.savefig('Probability of outcome by crime.png', dpi=fig.dpi, bbox_inches='tight')
plt.show()
```



Stratifying by Region.

In [29]:

```python
tab3 = pd.crosstab(columns=[df['Year'],df['Outcome_type']],index=df['Region'])
tab3
```

Out[29]:

Year	2019				2020				2021			
Outcome_type / Region	No suspect identified	Other Outcome	Suspect charged	Unable to prosecute suspect	No suspect identified	Other Outcome	Suspect charged	Unable to prosecute suspect	No suspect identified	Other Outcome	Suspect charged	Unable to prosecute suspect
East Midlands	338914	68193	79448	386654	286892	79261	70447	388328	289821	70388	49721	376532
North	141791	31301	27657	87823	117535	24916	25873	86284	109002	19342	17726	81614
South East	539389	118064	101709	372020	762806	122999	97239	369298	588702	106110	63474	334526
South West	182176	49256	57773	260915	138312	46239	53655	259116	122095	36482	34887	240208
West Midlands	282383	54285	62979	301552	230325	46089	53377	319101	181597	41898	30009	259404

In [30]:

```python
pivot2019_2 = pivoting_stacks(tab3,2019)
pivot2020_2 = pivoting_stacks(tab3,2020)
pivot2021_2 = pivoting_stacks(tab3,2021)
```

Plotting the regional probabilities.

In [64]:

```python
fig, (a1,a2,a3) = plt.subplots(3,1,figsize=(11,14),constrained_layout = True)
fig.suptitle('Probability of Outcome by Region',x=0.39,y=1.01,fontsize=16)
pivot2019_2.plot.bar(stacked=True,ax=a1,title='2019',rot=0,colormap='icefire',ylabel='Probability').legend(loc=
pivot2020_2.plot.bar(stacked=True,ax=a2,title='2020',rot=0,colormap='icefire',ylabel='Probability').legend(loc=
pivot2021_2.plot.bar(stacked=True,ax=a3,rot=0,title='2021',colormap='icefire',ylabel='Probability').legend(loc=
plt.ylabel('Probability')
for container in a1.containers:
    a1.bar_label(container,label_type='center')
for container in a2.containers:
    a2.bar_label(container,label_type='center')
for container in a3.containers:
    a3.bar_label(container,label_type='center')
#plt.savefig('Prob_of_outcome_by_region.png',dpi=fig.dpi, bbox_inches='tight')
plt.show()
```

Normalising by populations

Using extra data such as regional populations and police populations to normalise crime rate counts.

In [32]:

```python
grouped4 = df.groupby(['Month','Region','Outcome_type'])['Crime_ID'].count().reset_index()
```

In [33]:

```python
grouped4['Month'] = pd.to_datetime(grouped4['Month'])
```

In [34]:

```python
grouped4
```

Out[34]:

	Month	Region	Outcome_type	Crime_ID
0	2019-01-01	East Midlands	No suspect identified	28641
1	2019-01-01	East Midlands	Other Outcome	5746
2	2019-01-01	East Midlands	Suspect charged	6477
3	2019-01-01	East Midlands	Unable to prosecute suspect	29608
4	2019-01-01	North	No suspect identified	11534
...	...	...	...	...
715	2021-12-01	South West	Unable to prosecute suspect	8247
716	2021-12-01	West Midlands	No suspect identified	7097
717	2021-12-01	West Midlands	Other Outcome	1392
718	2021-12-01	West Midlands	Suspect charged	648
719	2021-12-01	West Midlands	Unable to prosecute suspect	6961

720 rows × 4 columns

In [35]:

```python
grouped5 = df.groupby(['Month','Region','Crime_type'])['Crime_ID'].count().reset_index()
grouped5['Month'] = pd.to_datetime(grouped5['Month'])
grouped5.head()
```

Out[35]:

	Month	Region	Crime_type	Crime_ID
0	2019-01-01	East Midlands	Criminal damage and arson	9274
1	2019-01-01	East Midlands	Drug offences	2146
2	2019-01-01	East Midlands	Public order offences	6168
3	2019-01-01	East Midlands	Theft offences	23778
4	2019-01-01	East Midlands	Violence and sexual offences	29106

In [36]:

```python
grouped5['Count_normed'] = grouped5.apply(lambda row: (row['Crime_ID']/10475600)*10000 if row['Region']=='East 
)*10000 if row['Region']=='South East' else ((row['Crime_ID']/9937900)*10000 if row['Region']=='South West' els
```

In [37]:

```python
grouped5['Count_normed_pol'] = grouped5.apply(lambda row: (row['Crime_ID']/20522)*100 if row['Region']=='East 
                                               ((row['Crime_ID']/58161)*100 if row['Region']=='South East' else
                                                ((row['Crime_ID']/18605)*100 if row['Region']=='South West' else
                                                 ((row['Crime_ID']/18156)*100 if row['Region']=='West Midlands'
                                                 ((row['Crime_ID']/8062)*100 if row['Region']=='North' else 'tes
```

In [38]:

```python
grouped5
```

Out[38]:

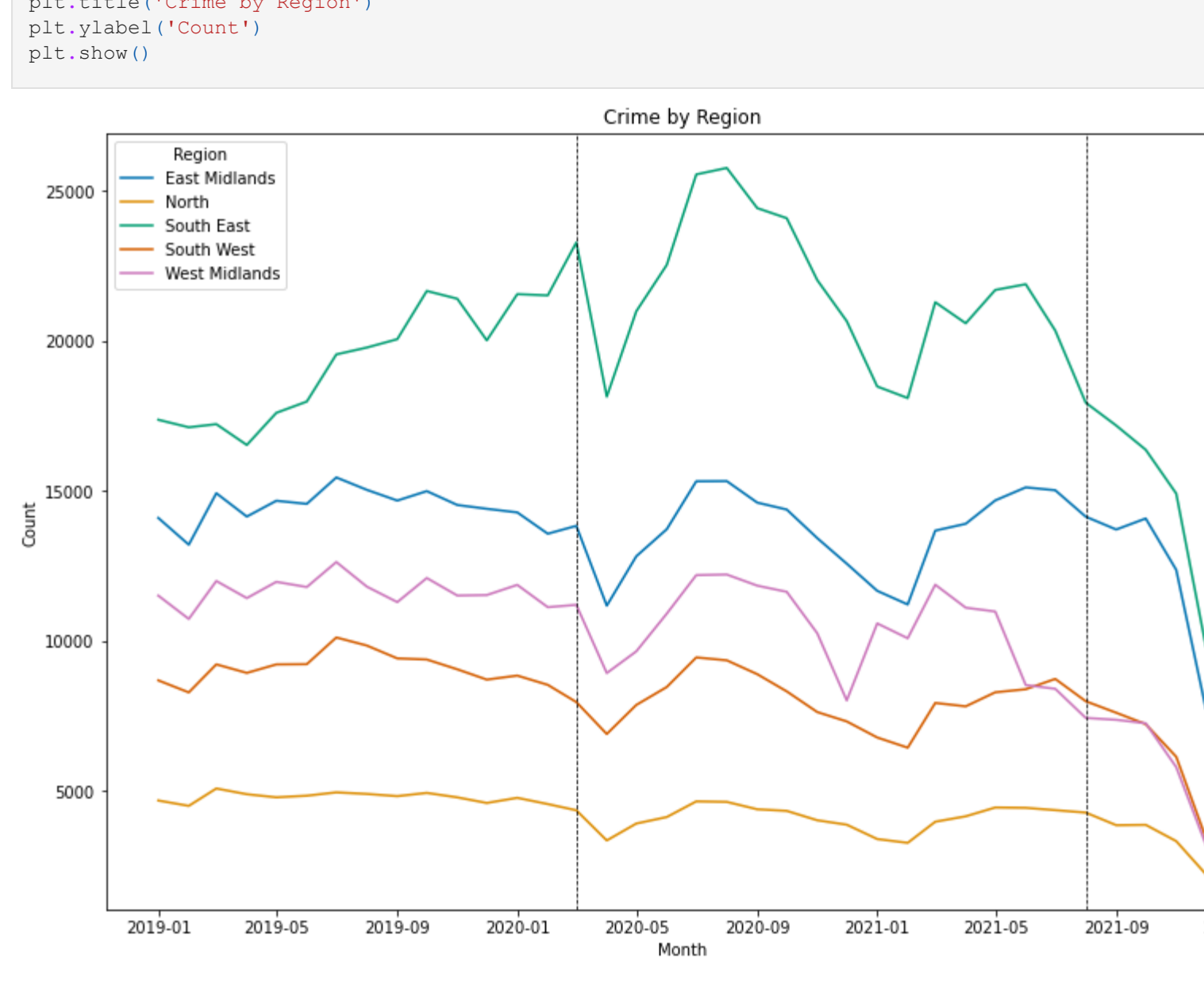
	Month	Region	Crime_type	Crime_ID	Count_normed	Count_normed_pol
0	2019-01-01	East Midlands	Criminal damage and arson	9274	8.852954	45.190527
1	2019-01-01	East Midlands	Drug offences	2146	2.048570	10.457070
2	2019-01-01	East Midlands	Public order offences	6168	5.887968	30.055550
3	2019-01-01	East Midlands	Theft offences	23778	22.698461	115.865900
4	2019-01-01	East Midlands	Violence and sexual offences	29106	27.784566	141.828282
...	...	...	...	...	...	...
895	2021-12-01	West Midlands	Criminal damage and arson	2102	2.193238	11.577440
896	2021-12-01	West Midlands	Drug offences	432	0.450751	2.379379
897	2021-12-01	West Midlands	Public order offences	2140	2.232888	11.786737
898	2021-12-01	West Midlands	Theft offences	4154	4.334306	22.879489
899	2021-12-01	West Midlands	Violence and sexual offences	7270	7.585558	40.041859

900 rows × 6 columns

Raw regional crime counts.

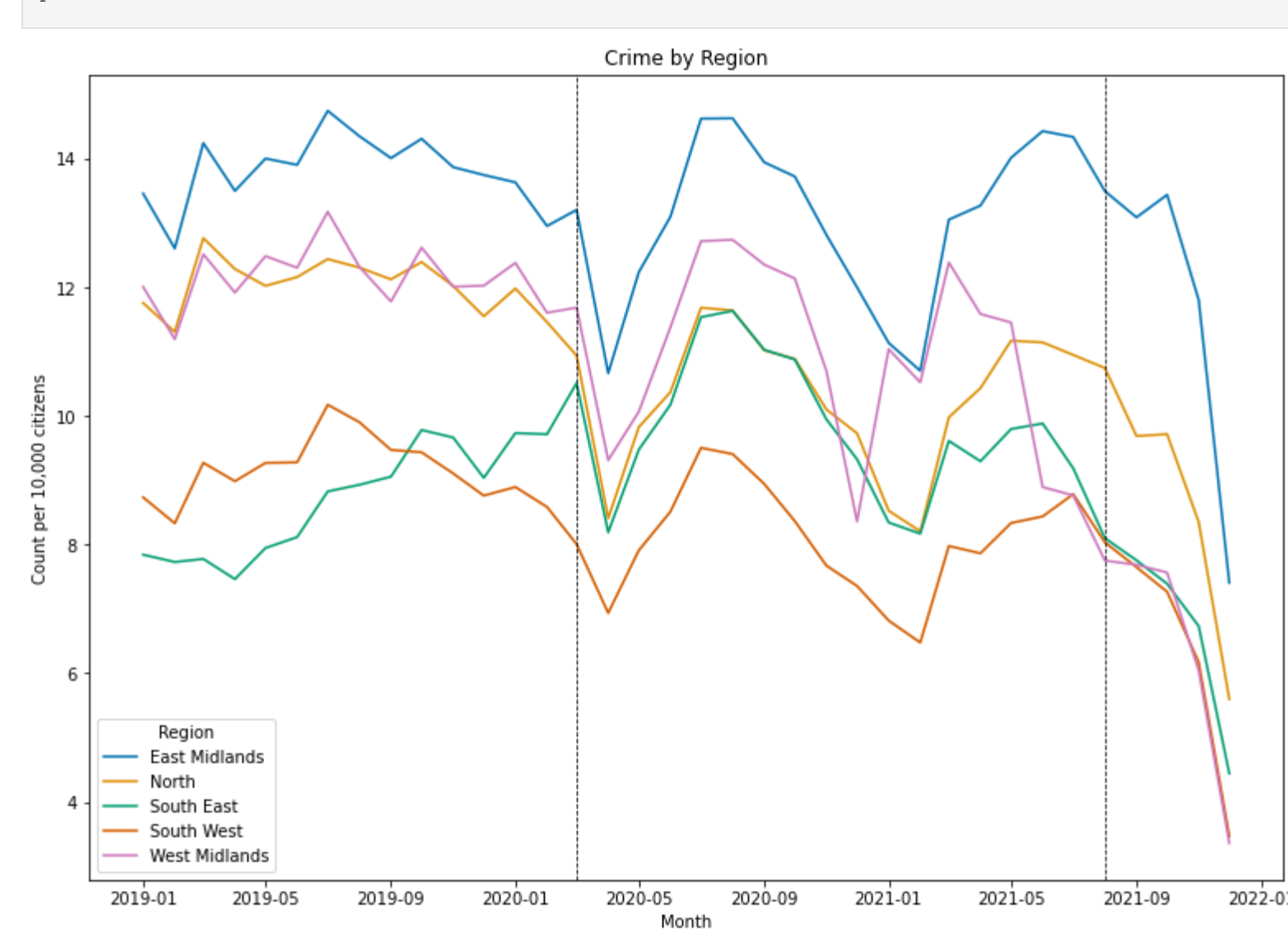
In [39]:

```python
plt.figure(figsize=(13,9))
sns.lineplot(data=grouped5,x='Month',y='Crime_ID',hue='Region',ci=None)
plt.axvline(dt.datetime(2020, 3, 1),linestyle='--',color='black',linewidth=0.75)
plt.axvline(dt.datetime(2021, 8, 1),linestyle='--',color='black',linewidth=0.75)
plt.title('Crime by Region')
plt.ylabel('Count')
plt.show()
```



Counts normalised by 10,000 citizens.

In [40]:
```
plt.figure(figsize=(13,9))
sns.lineplot(data=grouped5,x='Month',y='Count_normed',hue='Region',ci=None)
plt.axvline(dt.datetime(2020, 3, 11),linestyle='--',color='black',linewidth=0.75)
plt.axvline(dt.datetime(2021, 8, 11),linestyle='--',color='black',linewidth=0.75)
plt.title('Crime by Region')
plt.ylabel('Count per 10,000 citizens')
plt.savefig('Crime by Region per 10,000 citizens.png')
plt.show()
```

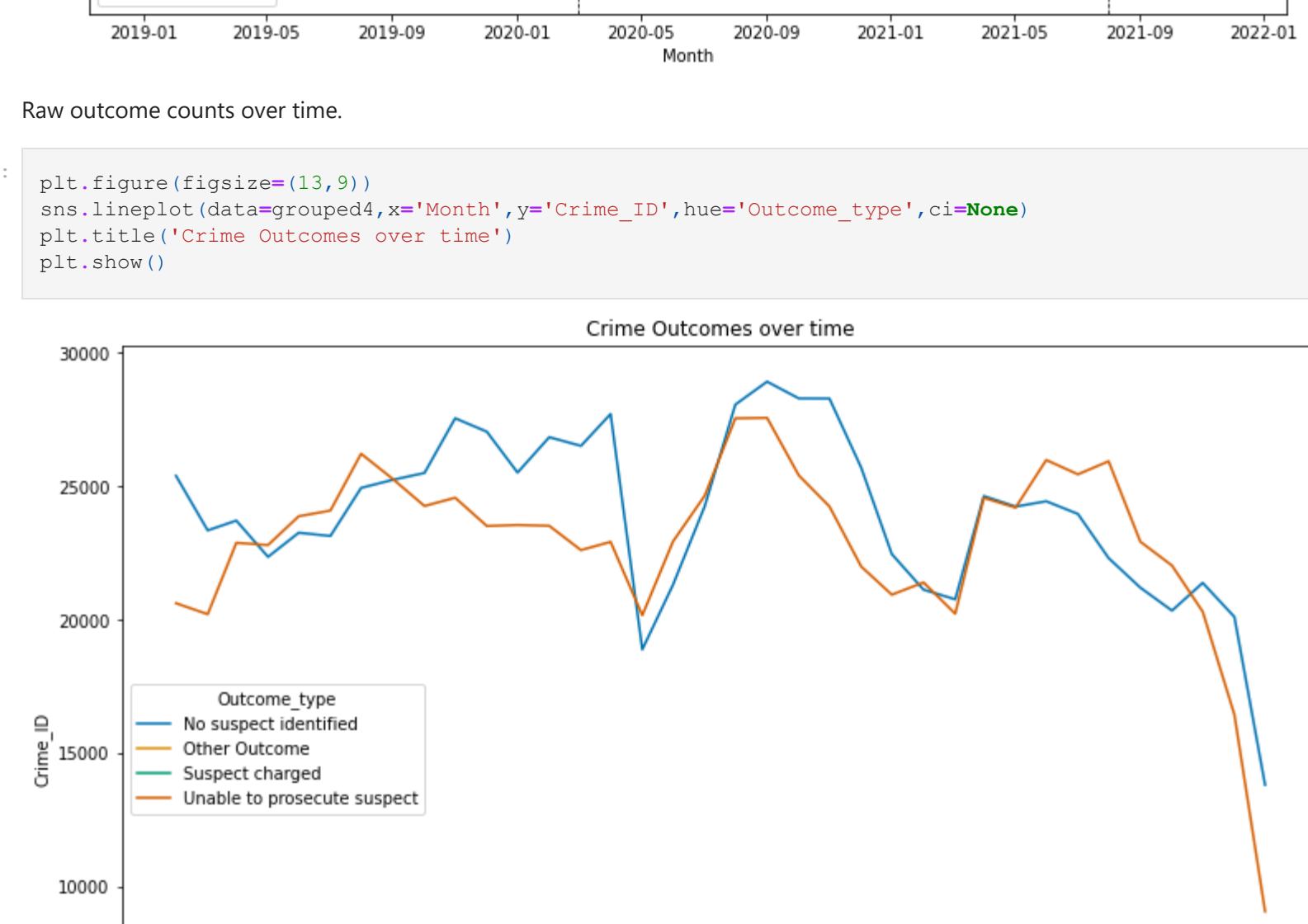


Counts normalised by 100 police officers.

In [41]:
```
plt.figure(figsize=(13,9))
sns.lineplot(data=grouped5,x='Month',y='Count_normed_pol',hue='Region',ci=None)
plt.axvline(dt.datetime(2020, 3, 11),linestyle='--',color='black',linewidth=0.75)
plt.axvline(dt.datetime(2021, 8, 11),linestyle='--',color='black',linewidth=0.75)
plt.title('Crime by Region')
plt.ylabel('Count per 100 police officers')
plt.show()
```

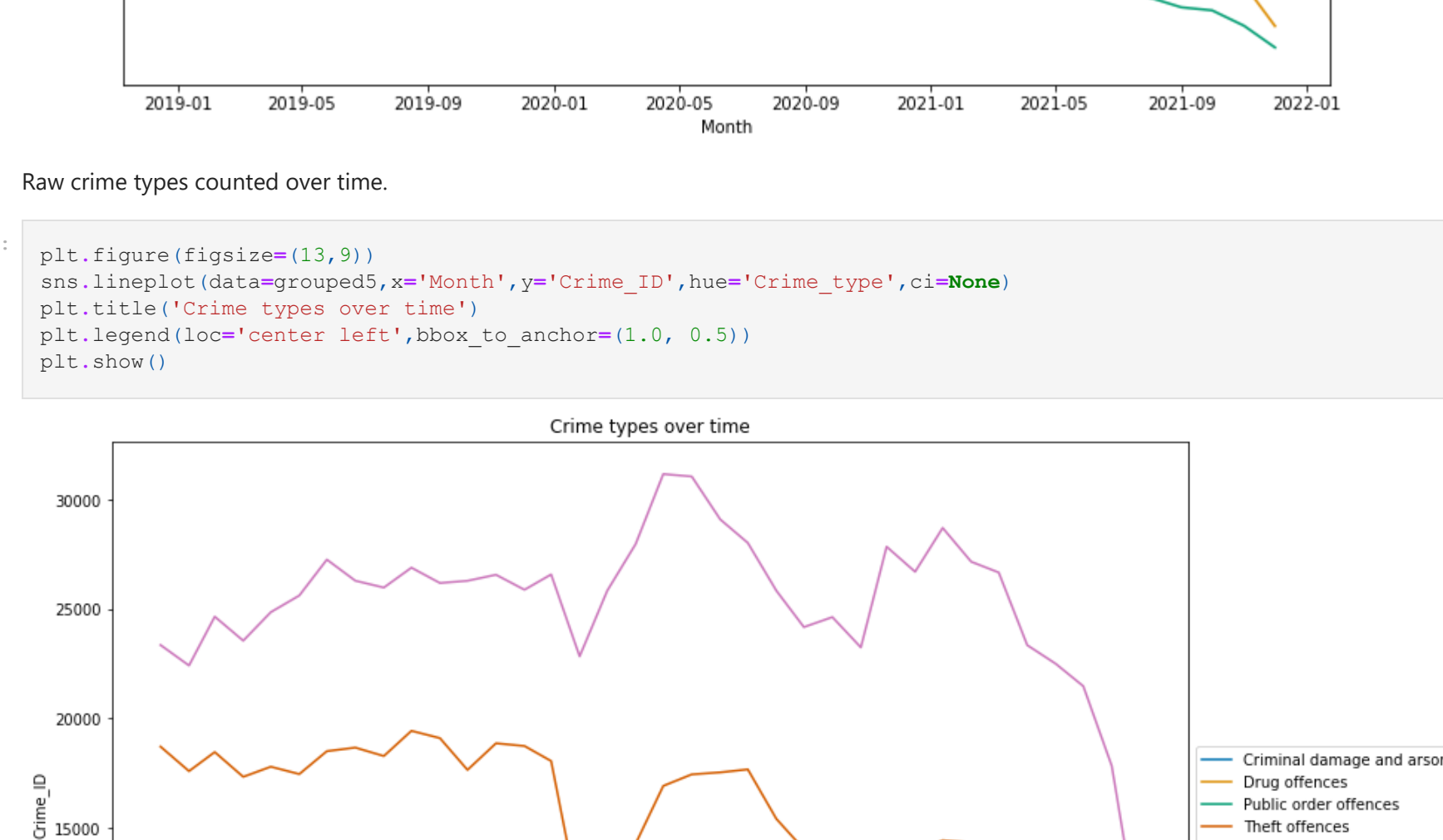
Raw outcome counts over time.

In [42]:
```
plt.figure(figsize=(13,9))
sns.lineplot(data=grouped5,x='Month',y='Crime_ID',hue='Outcome_type',ci=None)
plt.title('Crime Outcomes over time')
plt.show()
```



Raw crime types counted over time.

In [43]:
```
plt.figure(figsize=(13,9))
sns.lineplot(data=grouped5,x='Month',y='Crime_ID',hue='Crime_type',ci=None)
plt.title('Crime types over time')
plt.legend(loc='center left',bbox_to_anchor=(1.0, 0.5))
plt.show()
```



In [44]:
```
grouped6 = df.groupby(['Level_of_Lockdown','Region'])['Crime_ID'].count()
```

In [45]:
```
grouped6
```

Out[45]:
```
Level_of_Lockdown  Region
High               East Midlands     824928
                   North             254608
                   South East       1352342
                   South West        497322
                   West Midlands     648892
Medium             East Midlands     786462
                   North             227684
                   South East       1092812
                   South West        433672
                   West Midlands     512908
No Lockdown        East Midlands     873209
                   North             288572
                   South East       1131182
                   South West        550120
                   West Midlands     701199
Name: Crime_ID, dtype: int64
```

Probabilities of each crime type by region over the entire time period.

In [46]:
```
tab4 = pd.crosstab(columns=[df['Crime_type']],index=df['Region'],normalize='columns')
tab4
```

Out[46]:

Crime_type	Criminal damage and arson	Drug offences	Public order offences	Theft offences	Violence and sexual offences
Region					
East Midlands	0.252309	0.183610	0.269411	0.233156	0.248532
North	0.104903	0.049959	0.076133	0.076080	0.069995
South East	0.315409	0.485330	0.318576	0.399137	0.326538
South West	0.148867	0.128484	0.153546	0.125833	0.156665
West Midlands	0.178512	0.152618	0.182334	0.165795	0.198270