Do prison reform programs reduce recidivism?

Introduction

Business Context. Recidivism is a measure of a former prisoner's likelihood to be re-arrested, re-convicted, or returned to prison with or without a new sentence during a three-year period following the prisoner's release. It has been used to study the performance and effectiveness of privately and publicly managed prisons. Recidivism rates among prisoners in the United States are high, with more than 60% of released prisoners re-offending within three years of release.

Consequently, prison reform programs have invested hundreds of millions of dollars per year on programs designed to reduce re-offense. However, implementing these programs is costly and evaluating them can be complex. Currently, policymakers are interested in evaluating a training program designed to reduce recidivism among paroled inmates. They have come to you for assistance in determining whether it is worthwhile.

Business Problem. The policymakers want you to answer the following: **"Will there be a reduction in recidivism if the proposed program were to be widely adopted?"**

Analytical Context. The new program was piloted for a subset of parolees (recently released prisoners) from two prisons. New parolees at these two prisons were given the option to volunteer to participate in the recidivism reduction program. If we determine that the program is worthwhile, it would subsequently be scaled to all parolees from twenty-five eligible prisons in the federal penal system.

The case is structured as follows: you will (1) analyze the setup of the program and identify potential observational study design problems; (2) learn about matching and use it to correct for some of these problems; and finally (3) conduct exploratory data analysis to determine which features seem to account for most of the corrections/differences between the policymakers' study and our matched study.

Initial analysis of the program

The program we are evaluating in this case study was focused on vocational training and coping skills and was applied to approximately $n = 7000$ parolees from 2013 - 2015 in the southeastern United States. Participants were followed for 30 months after completing the program and identified as a "responder" if they did not commit a serious offense during that period, and a "non-responder" if they did. The data collected in the study are of the form $\{\mathbf{X}_i, Y_i\}_{i=1}^n$ comprising n independent $(\mathbf{X}, Y)$ pairs, one per participating parolee, where $\mathbf{X}$ is a vector of parolee characteristics and $Y \in \{0, 1\}$ is the binary outcome (1 meaning re-offended, 0 meaning did not re-offend). Parolee characteristics are as follows:

Variable	Description
ID	Unique prisoner identifier
Max severity	(1-5) Severity of most serious charge in current conviction
Ave severity	(1-5) Average severity across all charges in currennt conviction
Sentence time	Sentence length (months) assigned
Sentence served	Time served (months)
Education	(1-4) coded so that: 1 - less than High School, 2 - High School, 3 - Some College, and 4 - College Degree
Repeat offender	Binary indicator of previous conviction
Substance use	Binary indicator of substance abuse
Support	Binary indicator that parolee is being provided with transitional support (e.g., halfway house)

In [1]:

```
from plotly.offline import init_notebook_mode, plot
import numpy as np
import numpy.random as nr
from numpy.linalg import norm
import plotly.express as px
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
study_data_pd = pd.read_csv('rec_study.csv')
study_data_pd.head()
```

Out[1]:

	max_severity	ave_severity	sentence_time	sentence_served	repeat_offender	substance_u
0	1	1.000000	15.281172	4.032279	0	
1	1	1.000000	19.816019	1.625926	0	
2	2	1.333333	59.293718	38.978041	0	
3	2	1.333333	72.026361	25.082745	1	
4	1	1.000000	10.618477	9.412762	0	

Exercise 1:

What is one potential issue you see with the program design that might dilute the conclusiveness of your results?

Answer. Because enrollment into the program was voluntary, the studied sample in the experiment was not randomized among all eligible prisoners. Thus, care must be taken in generalizing any observed effects in the participating parolees.

Exercise 2:

Conduct a basic exploratory data analysis to see if any of the covariates appear to be related to the probability of re-offense. What is the average rate of re-offense of the participants in the study?

Answer. All variables are either numeric, or numeric categories with clear directionality in their labels (e.g. for `Education`, the higher the label number, the more educated said participant is). This means that a correlation matrix is a good initial tool which will quickly give us basic insight into all the variables at once:

In [3]:

```
# Correlations
study_data_pd.corr()
```

Out[3]:

	max_severity	ave_severity	sentence_time	sentence_served	repeat_offende
max_severity	1.000000	0.817755	0.793862	0.515839	0.13719
ave_severity	0.817755	1.000000	0.475187	0.308140	0.09402
sentence_time	0.793862	0.475187	1.000000	0.715039	0.12782
sentence_served	0.515839	0.308140	0.715039	1.000000	0.05733
repeat_offender	0.137198	0.094026	0.127827	0.057331	1.00000
substance_use	0.082750	0.076825	0.057291	0.038360	0.02772
support	-0.001907	-0.004226	0.006061	0.023929	-0.00560
education	-0.007973	-0.001395	-0.006468	0.000696	0.00747
reoffend	-0.242499	-0.092318	-0.379868	-0.092872	-0.02306

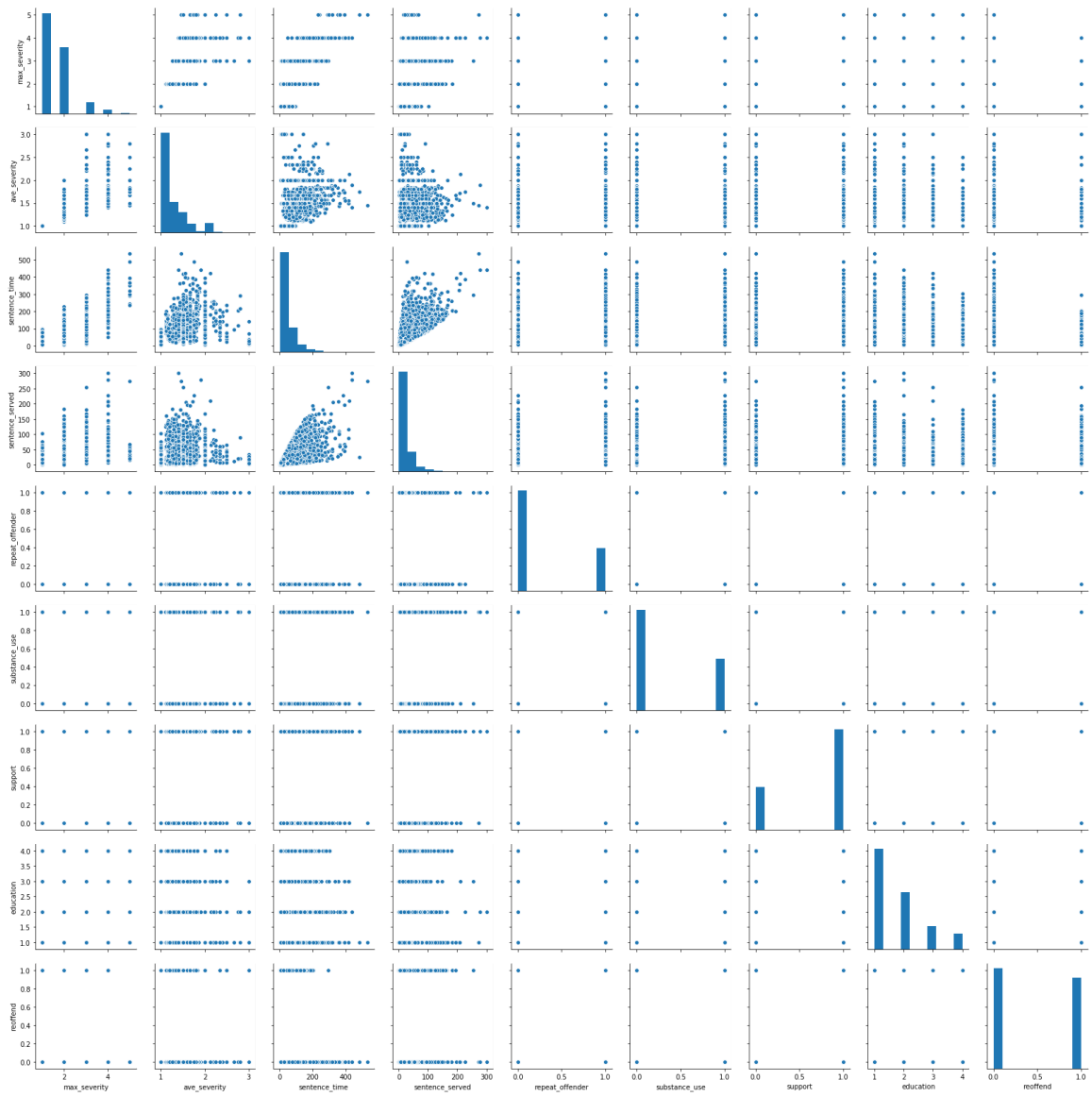
Since correlations are best suited for analyzing linear relationships, let's create a matrix of 2D scatterplots for each pair of variables also, so that we can visually pick up on any potential non-linear relationships:

In [4]:

```
# Scatter plot matrix
import seaborn as sns
sns.pairplot(study_data_pd)
```

Out[4]:

```
<seaborn.axisgrid.PairGrid at 0x13518828>
```



From this, we can see that the likelihood of re-offense increases with the following variables:

1. Maximum severity of the crime (i.e., the worst crime they were convicted of)
2. Substance use
3. Whether or not they were a repeat offender before

However, we alo see that it decreases as the gap between sentence time and time served. Possible explanations include that increased severity and prior offense indicate a pattern of criminal behavior and therefore an increased propensity for re-offense. A (positive) gap between sentence time an time served indicates that the subject was released early which is often an indicator of 'good behavior' which in tern may be an indicator of a desire for change, self-control, or other positive predictors for response.

In [5]:

```
study_data_pd['reoffend'].mean()
```

Out[5]:

```
0.4751583391977481
```

The avearge rate of re-offense is 47.5%, which is significantly lower than the national average of 60%. A reduction of nearly 12.5%, *if* it held throughout the entire system, would be a significant reduction translating into tens of millions of dollars saved on incarceration costs along with immeasurable other societal benefits.

Exercise 3:

A reduction of 12.5% *would* be remarkable *if* such results held across all prisoners released across the entire prison system. Recall that policy makers are interested in the value of a program if it were instituted across the entire system. Can you identify any sources of bias that might prevent the study from generalizing to the entire prison system?

Answer. A major issue is that the type of prisoner who would opt into such a program may have a strong desire to avoid future incarceration and therefore be less likely to re-offend. Put another way, the prisoners in the study may not be representative of the population of released prisoners in terms of their response because the factors influencing their decision to join the study may also influence their likelihood of response.

Another potential source of bias is the fact that prisoners were only enrolled from two of the twenty-five eligible prisons. If these two prisons somehow differ from the other twenty-three in terms of the likelihood that prisoners will respond to the program (that is not captured in the measured covariates) then this could also bias our results.

Let's assume now that the two sampled prisons are *not* systematically different from the others.

Exercise 4:

Given that the set of prisoners who opt into the program could be biased, suggest three possible approaches to estimate the response rate in the entire population of released prisoners. Discuss the costs and benefits of these approaches.

Answer. Three potential approaches are:

1. Run a follow-up study where subjects are randomly enrolled into the program and forced to participate as part of the terms of their release. This would certainly provide a representative sample but could be extremely costly and would delay implementation of full-scale implementation of the program.
2. Interview prisoners to identify factors influencing their decision to enroll in the study. Use qualitative or quantitative analyses to discern whether those who are more likely to respond(and not re-offend) were more likely to enroll. This approach might be reasonable if it is very difficult or impossible to learn anything about the entire population of prisoners any other way. However, this would also mean that it is very difficult or impossible to validate whether or not the potentially influential factors that were abundant amongst the prisoners in the study were also relatively absent in the general population of prisoners. This is an example of potential **confirmation bias**.
3. Use the covariates of the prisoners that did not enroll in the study (measured at release) to identify and adjust for differences between those enrolled in the study with those that did not. In our analysis, we will use **matching** which is an example of this type of approach.

Causal analysis via matching

The core idea of matching is to group untreated individuals with treated individuals (here, the treatment is the recivdivism reduction program) based on similarities in measured covariates to estimate the effect of treatment across the entire population. Assume that in addition to the *in-study data*, $\mathcal{S} = \{\mathbf{X}_i, Y_i\}_{i=1}^n$ we have data from prisoners released during the study period that either opted out of the study or were not eligible because they were not at one of the two prisons included in the study. Let us call this set of features $\mathcal{E} = \{\mathbf{X'}_j\}_{j=1}^N$. Let us also call this the *out-of-study data*.

For each subject j in the out-of-study dataset $\mathcal{E}$, we associate a subject $i(j)$ in the in-study dataset $\mathcal{S}$ that is the closest match according to some measure of distance between subjects. In our study, this measure of distance will be based on the covariates $\mathbf{x}$ and $\mathbf{x}'$ using Euclidean distance:

$$d(\mathbf{x}, \mathbf{x}') = \sqrt{\sum_{k=1}^{p} (x_k - x'_k)^2}.$$

Thus for each $j \in \mathcal{E}$ define $i(j)$ to be the observation in $\mathcal{S}$ that is closest to j in the Euclidean distance mentioned above. The intuition for the matched data set is that subject $i(j)$ in the study is the "closest"/most similar to subject j in $\mathcal{E}$.

After matching, we will compute the mean recidivism rate for the matched sample. This form of matching is called **nearest-neighbor-one matching (KNN-1)** and is among the most popular and intuitive variants of matching.

The out-of-study dataset $\mathcal{E}$ is stored in `rec_popn.csv`. Let's first read in the `rec_study.csv` data into a dataframe called `in_study_data`, then read in `rec_popn.csv` into a dataframe called `out_of_study`:

In [8]:

```
in_study_data = pd.read_csv('rec_study.csv')
n, p = np.shape(in_study_data)
out_of_study_data =  pd.read_csv('rec_popn.csv')
m, q = out_of_study_data.shape
```

Let's normalize the data now:

In [9]:

```
#https://stackoverflow.com/questions/9171158/how-do-you-get-the-magnitude-of-a-vector-in-numpy
in_study_norm = norm(in_study_data.iloc[:,0:8], axis=0)
in_study_data.iloc[:,0:8]/= in_study_norm
out_of_study_data/=  in_study_norm
```

Question:

Can you explain what the above code is doing? Why is it important?

`tqdm` package

Sometimes, you will have to use for-loops, even if they are not efficient. You can use the `tqdm` package to get an estimate of how long it will take for stuff to run:

In [11]:

```
from tqdm import tqdm_notebook, tnrange
for j in tnrange(1000000, desc = "Iterating"):
    if j%50000 == 0:
     print("Finished " + str(j) + " iterations")
```

```
Finished 0 iterations
Finished 50000 iterations
Finished 100000 iterations
Finished 150000 iterations
Finished 200000 iterations
Finished 250000 iterations
Finished 300000 iterations
Finished 350000 iterations
Finished 400000 iterations
Finished 450000 iterations
Finished 500000 iterations
Finished 550000 iterations
Finished 600000 iterations
Finished 650000 iterations
Finished 700000 iterations
Finished 750000 iterations
Finished 800000 iterations
Finished 850000 iterations
Finished 900000 iterations
Finished 950000 iterations
```

Exercise 5:

Write code to create the matched data set for the first 10000 rows of the normalized out-of-study data using the normalized in-study data. (It may take a while to run, so you can use `tqdm` to track your progresss.) Once you have completed the matching, compute the estimated matched mean.

In [12]:

```
study_outcome = in_study_data['reoffend']
in_study_data = np.array(in_study_data)
out_of_study_data = np.array(out_of_study_data)
running_matched_mean = 0.0
n_run = 10000 # number of matches to use in average
for j in tnrange(n_run, desc="Number records processed"):
    dist = np.inf
    ij = None
    for i in range(n):
        i_dist =  np.sqrt(sum(map(lambda x: x*x,(out_of_study_data[j,:] - in_study_data
[i,0:8]))))
        if i_dist < dist:
            ij = i
            dist = i_dist
    running_matched_mean += study_outcome[ij]
    if j % 500 == 0 and j > 1:
        print("Finished processing: " + str(j) + " records")
        print(running_matched_mean/j)
print(running_matched_mean/n_run)
```

```
Finished processing: 500 records
0.538
Finished processing: 1000 records
0.564
Finished processing: 1500 records
0.558
Finished processing: 2000 records
0.552
Finished processing: 2500 records
0.55
Finished processing: 3000 records
0.5473333333333333
Finished processing: 3500 records
0.5494285714285714
Finished processing: 4000 records
0.546
Finished processing: 4500 records
0.5504444444444444
Finished processing: 5000 records
0.5486
Finished processing: 5500 records
0.5492727272727272
Finished processing: 6000 records
0.5505
Finished processing: 6500 records
0.5495384615384615
Finished processing: 7000 records
0.5511428571428572
Finished processing: 7500 records
0.554
Finished processing: 8000 records
0.552
Finished processing: 8500 records
0.552
Finished processing: 9000 records
0.5547777777777778
Finished processing: 9500 records
0.554

0.5549
```

We see from the results that the estimated benefit to the population outside the study was significantly worse than the in-study population; in fact, it was closer to being on par with the average among the untreated (60%). Thus, we might hesitate to recommend the policy be widely adopted across the entire population.

Unfortunately, this is a common phenomenon in these types of studies. Let's see if we can dig into this result and understand why the results differ. Along the way, we'll explore ways to improve our matching criterion and to speed up the matching algorithm.

Looking at differentials in certain features

To begin, let's consider inclusion in the study as a binary classification problem, wherein a subject is labeled as a "1" if they were included in the study and "0" if they were not. To explore differences in the study and non-study populations, we can consider building a model that predicts whether or not a subject was in the study using their baseline covariates. As a first step in building such a predictive model, though, let's do some basic exploratory data analysis and look at distributions of the covariates.

Exercise 6:

6.1

Create a new column on both the datasets `rec_study.csv` and `res_popn.csv` ; call it `sample_ind` . This column should take the value "1" for `rec_study.csv` data and "0" for `res_popn.csv` data. Merge them into a single dataset using outer merge.

Answer. One possible solution is given below:

In [24]:

```
in_study_data = pd.read_csv('rec_study.csv')
out_of_study_data =  pd.read_csv('rec_popn.csv')
in_study_data['sample_ind'] = 1
out_of_study_data['sample_ind']  = 0
```

In [25]:

```
df_merged = pd.merge(in_study_data, out_of_study_data, how = 'outer')
df_merged.head(-5)
```

Out[25]:

	max_severity	ave_severity	sentence_time	sentence_served	repeat_offender	substan
0	1	1.000000	15.281172	4.032279	0	
1	1	1.000000	19.816019	1.625926	0	
2	2	1.333333	59.293718	38.978041	0	
3	2	1.333333	72.026361	25.082745	1	
4	1	1.000000	10.618477	9.412762	0	
5	1	1.000000	8.526859	5.647969	1	
6	1	1.000000	19.977480	14.798382	0	
7	1	1.000000	5.295902	3.467049	1	
8	4	1.666667	298.701097	140.876225	0	
9	1	1.000000	17.946980	2.773393	0	
10	2	1.400000	124.125845	31.672182	1	
11	2	1.142857	79.799244	37.458051	0	
12	1	1.000000	10.902982	11.370792	0	
13	1	1.000000	23.726816	7.190237	0	
14	1	1.000000	18.809509	19.316012	0	
15	1	1.000000	38.572868	5.651393	0	
16	2	1.200000	80.612917	4.339537	1	
17	2	1.200000	93.574935	7.388788	1	
18	2	1.333333	100.981624	71.097431	0	
19	1	1.000000	19.907603	4.259912	0	
20	1	1.000000	14.205047	13.852795	0	
21	1	1.000000	12.546560	3.142385	0	
22	1	1.000000	17.406404	9.767415	0	
23	2	1.333333	34.968850	30.823045	0	
24	3	1.500000	83.500166	11.950706	0	
25	3	1.800000	146.726828	81.908194	1	
26	1	1.000000	17.348317	10.834608	0	
27	2	2.000000	25.934207	20.638645	1	
28	2	1.250000	88.154936	56.569514	0	
29	1	1.000000	14.309322	11.359071	1	
...	...	...	...	...	...	
62869	2	1.666667	28.324300	30.338066	0	
62870	4	2.000000	102.849893	33.093896	1	
62871	2	2.000000	11.743496	5.990822	0	
62872	2	1.200000	68.955877	49.039900	0	
62873	4	2.333333	186.048714	95.669880	0	
62874	2	1.333333	59.718572	59.491885	0	

	max_severity	ave_severity	sentence_time	sentence_served	repeat_offender	substan
62875	3	2.000000	80.269069	54.199397	1	
62876	2	1.666667	70.968099	34.283895	0	
62877	1	1.000000	6.743815	1.454250	0	
62878	1	1.000000	71.489122	18.575055	0	
62879	2	1.600000	107.949823	37.815765	1	
62880	4	3.666667	136.044937	70.693149	1	
62881	2	1.200000	92.476670	54.831578	0	
62882	1	1.000000	22.962494	16.915258	0	
62883	4	2.750000	118.446061	74.125936	0	
62884	2	2.000000	19.984081	21.762326	0	
62885	1	1.000000	15.404371	10.446179	0	
62886	2	1.666667	48.325122	48.640648	1	
62887	2	1.666667	60.907890	8.165176	0	
62888	2	1.333333	28.625346	21.555069	0	
62889	1	1.000000	8.301588	8.568174	0	
62890	5	1.800000	169.614193	38.625058	1	
62891	1	1.000000	41.335552	31.803976	1	
62892	1	1.000000	10.360475	1.799266	1	
62893	5	2.000000	111.927893	111.650732	0	
62894	2	1.500000	78.036900	14.733427	0	
62895	2	1.333333	38.725135	38.685805	0	
62896	4	2.500000	92.901046	8.347756	1	
62897	1	1.000000	15.977986	9.667846	1	
62898	4	2.000000	157.926831	143.709528	1	

62899 rows × 10 columns

6.2

Create two plots: a boxplot of max severity for the two datasets, and a density plot of max severity for each severity bucket (1 - 5) for the two datasets (use `sns.kdeplot()`). What can you conclude?

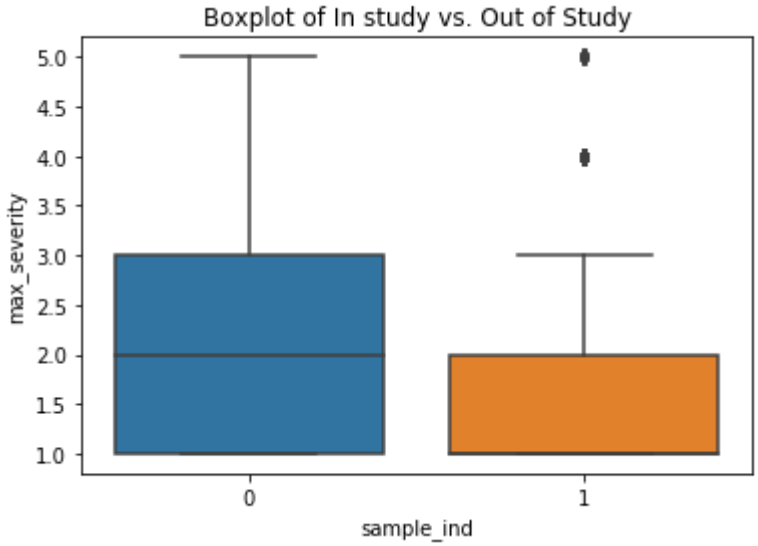
Answer. One possible solution is given below:

In [26]:

```
sns.boxplot(y = 'max_severity', x = 'sample_ind',data = df_merged)
plt.title("Boxplot of In study vs. Out of Study")
```

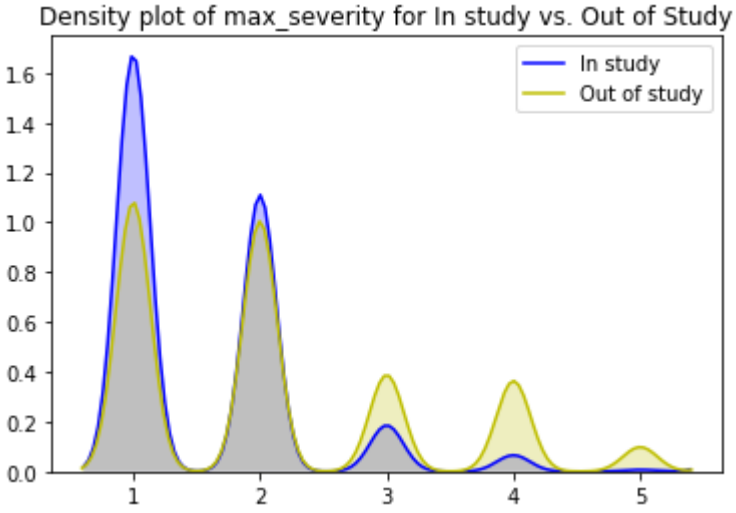
Out[26]:

```
Text(0.5, 1.0, 'Boxplot of In study vs. Out of Study')
```



In [27]:

```
sns.kdeplot(df_merged.loc[df_merged['sample_ind'] == 1,'max_severity'],shade = True, la
bel="In study",color="b")
sns.kdeplot(df_merged.loc[df_merged['sample_ind'] == 0,'max_severity'],shade = True, la
bel="Out of study",color="y")
plt.title("Density plot of max_severity for In study vs. Out of Study");
```



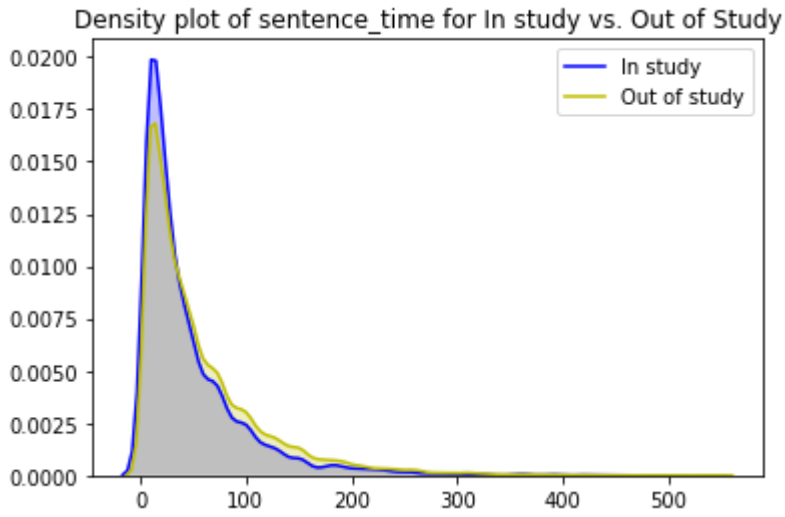
It can be seen that the distribution of max severity shifted to the right among non-study participants; i.e. criminals convicted of less severe crimes were more likely to enroll in the study.

6.3

Compare the distributions of the remaining variables for in-study vs. out-of-study. Also, look at the distribution in the sentence gap: sentence time minus sentence served.

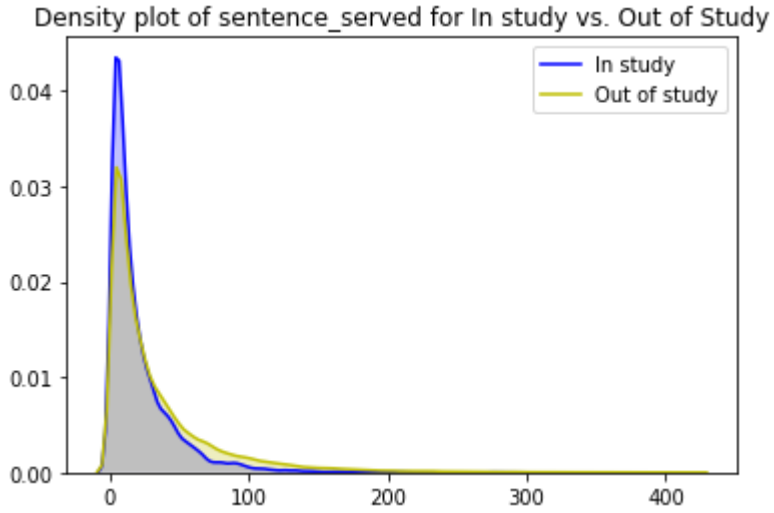
In [28]:

```
sns.kdeplot(df_merged.loc[df_merged['sample_ind'] == 1,'sentence_time'],shade = True, l
abel="In study",color="b")
sns.kdeplot(df_merged.loc[df_merged['sample_ind'] == 0,'sentence_time'],shade = True, l
abel="Out of study",color="y")
plt.title("Density plot of sentence_time for In study vs. Out of Study");
```



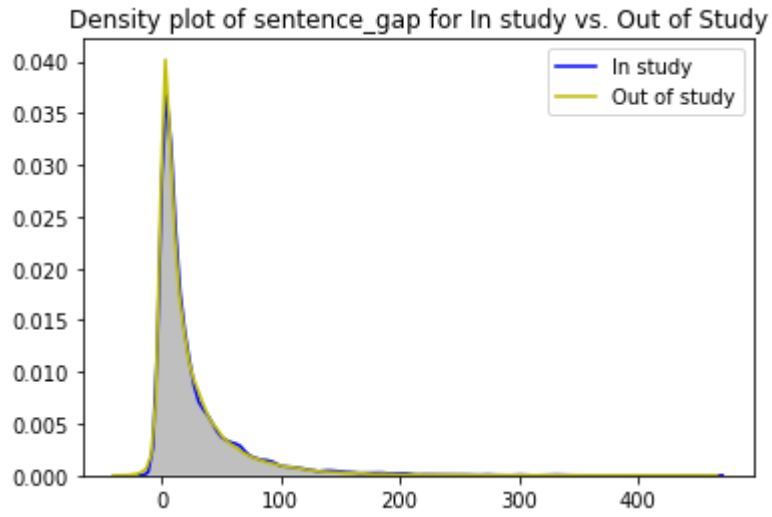
In [29]:

```
sns.kdeplot(df_merged.loc[df_merged['sample_ind'] == 1,'sentence_served'],shade = True,
label="In study",color="b")
sns.kdeplot(df_merged.loc[df_merged['sample_ind'] == 0,'sentence_served'],shade = True,
label="Out of study",color="y")
plt.title("Density plot of sentence_served for In study vs. Out of Study");
```



In [30]:

```
df_merged['sentence_gap'] = df_merged['sentence_time'] - df_merged['sentence_served']
sns.kdeplot(df_merged.loc[df_merged['sample_ind'] == 1,'sentence_gap'],shade = True, la
bel="In study",color="b")
sns.kdeplot(df_merged.loc[df_merged['sample_ind'] == 0,'sentence_gap'],shade = True, la
bel="Out of study",color="y")
plt.title("Density plot of sentence_gap for In study vs. Out of Study");
```



Similarly, if we look at the distribution of the sentencing gap, i.e., time sentenced minus time served, we see a difference in the distributions of the study and non-study populations. The figure below shows a **QQ plot** of the gap distributions. It can be seen that the right tail lies below the 45-degree line indicating that the right tail of the in-study population is larger than that of the out-of-study population. This suggests that those with a large gap (i.e. those released before their full sentence was served, often for "good behavior") were more likely to enter the study:

Conclusions

We examined matching as a means of analyzing the effectiveness of an observational recidivism study. We see that even though the study looked promising on the study population, this was primarily due to factors driving selection into the study. After adjusting for these factors, the program no longer appeared to be effective. We implemented the matching algorithm based on Euclidean distance between covariates.

Takeaways

In this case, we saw that one must take great care in analyzing observational data and pay particular attention to factors driving the construction of such studies. In particular, sampling and treatment assignment methodology are key, as lack of prudence in these steps can lead to significant differences between the in-study and out-of-study data points. Matching is one potential approach to analyzing such data. It is a conceptually simple approach but is also crude in that it does not weight covariates in terms of their importance for sample selection. In the future, it will be good to look at matching algorithms that weight covariates differently based on sensitivity to the objective variable.