Predict_Recidivism

February 3, 2023

```
[ ]: #Christine Orosco, Myra Wheeler, Kyle Hansen
     #Project to predict probability of recidivisim for a group of inmates
     # Dataset is the 3-yr Recidivism Rates for offenders released from Iowa prisons
```

```
[1]: # Import libraries
     import pandas as pd
     import numpy as np
     import seaborn as sns
     import matplotlib.pyplot as plt
```

```
[2]: # Import dataset
     df = pd.read_csv("3-Year_Recidivism_for_Offenders_Released_from_Prison_in_Iowa.
      ↪csv")
     df.head()
```

```
[2]:    Fiscal Year Released  Recidivism Reporting Year Main Supervising District  \
     0                  2010                       2013                        7JD
     1                  2010                       2013                        NaN
     2                  2010                       2013                        5JD
     3                  2010                       2013                        6JD
     4                  2010                       2013                        NaN

                      Release Type      Race - Ethnicity Age At Release    Sex  \
     0                      Parole  Black - Non-Hispanic          25-34  Male
     1  Discharged - End of Sentence  White - Non-Hispanic          25-34  Male
     2                      Parole  White - Non-Hispanic          35-44  Male
     3                      Parole  White - Non-Hispanic          25-34  Male
     4  Discharged - End of Sentence  Black - Non-Hispanic          35-44  Male

       Offense Classification Offense Type Offense Subtype Return to Prison  \
     0              C Felony      Violent         Robbery              Yes
     1              D Felony     Property           Theft              Yes
     2              B Felony         Drug     Trafficking              Yes
     3              B Felony        Other  Other Criminal               No
     4              D Felony      Violent         Assault              Yes

        Days to Return Recidivism Type New Offense Classification New Offense Type  \
0          433.0             New                 C Felony           Drug
1          453.0            Tech                      NaN            NaN
2          832.0            Tech                      NaN            NaN
3            NaN   No Recidivism                      NaN            NaN
4          116.0            Tech                      NaN            NaN

  New Offense Sub Type Target Population
0           Trafficking               Yes
1                   NaN                No
2                   NaN               Yes
3                   NaN               Yes
4                   NaN                No
```

[3]:

```python
# Get a list of column names
list(df.columns)
```

[3]:

```
['Fiscal Year Released',
 'Recidivism Reporting Year',
 'Main Supervising District',
 'Release Type',
 'Race - Ethnicity',
 'Age At Release ',
 'Sex',
 'Offense Classification',
 'Offense Type',
 'Offense Subtype',
 'Return to Prison',
 'Days to Return',
 'Recidivism Type',
 'New Offense Classification',
 'New Offense Type',
 'New Offense Sub Type',
 'Target Population']
```

[3]:

```python
# Pull out the info for the target variable to plot
recid_counts = pd.DataFrame(df['Return to Prison'].value_counts())
recid_counts.reset_index(inplace=True)
recid_counts
```

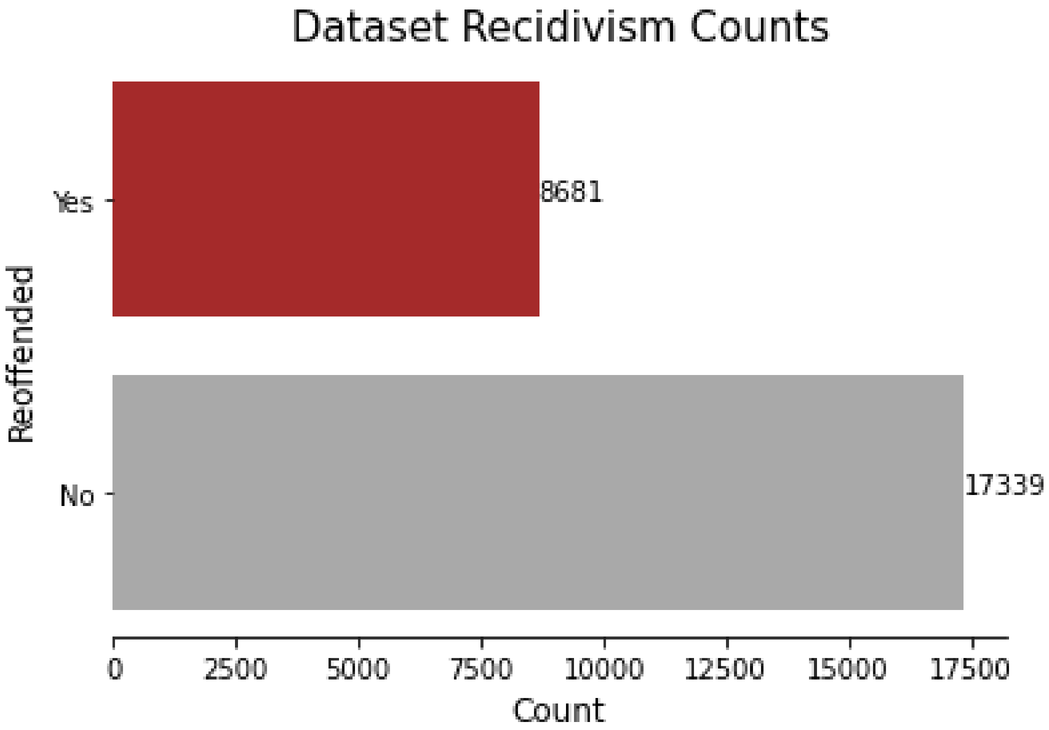
[3]:

```
  index  Return to Prison
0    No             17339
1   Yes              8681
```

[78]:

```python
# Plot counts of target variable
fig, ax = plt.subplots()

# Remove the right border
```

```
right = ax.spines["right"]
right.set_visible(False)

# Remove the top border
top = ax.spines["top"]
top.set_visible(False)

# Remove left border
left = ax.spines["left"]
left.set_visible(False)

# Plot
plt.barh(recid_counts.index, recid_counts['Return to Prison'],␣
 ↪color=['darkgray', 'brown'])
plt.ylabel("Reoffended", size=12)
plt.xlabel("Count", size=12)
plt.title("Dataset Recidivism Counts", size=15)
plt.yticks([0, 1], ["No", "Yes"])

for index, value in enumerate(recid_counts['Return to Prison']):
    plt.text(value, index, str(value))

plt.show()
```



[3]:

```
# Create new df dropping columns that are not going to be used
clean_df = df.drop(columns = ['Recidivism Reporting Year', 'Main Supervising␣
 ↪District', 'Race - Ethnicity', 'Days to Return',
                    'Recidivism Type', 'New Offense Classification', 'New␣
 ↪Offense Type', 'New Offense Sub Type',
                    'Target Population'])
clean_df.head()
```

[3]:

```
   Fiscal Year Released                  Release Type Age At Release    Sex  \
0                  2010                        Parole           25-34  Male   
1                  2010  Discharged - End of Sentence           25-34  Male   
2                  2010                        Parole           35-44  Male   
3                  2010                        Parole           25-34  Male   
4                  2010  Discharged - End of Sentence           35-44  Male   

  Offense Classification Offense Type Offense Subtype Return to Prison
0               C Felony      Violent         Robbery              Yes
1               D Felony     Property           Theft              Yes
2               B Felony         Drug     Trafficking              Yes
3               B Felony        Other  Other Criminal               No
4               D Felony      Violent         Assault              Yes
```

[4]:

```
# Change column names to be more usable
clean_df = clean_df.rename(columns={"Fiscal Year Released": "year", "Release␣
 ↪Type": "release", "Age At Release ": "age",
                                    "Sex": "sex", "Offense Classification":␣
 ↪"classification", "Offense Type": "type",
                                    "Offense Subtype": "subtype", "Return to␣
 ↪Prison": "reoffend"})
clean_df.head()
```

[4]:

```
   year                       release    age   sex classification      type  \
0  2010                        Parole  25-34  Male       C Felony   Violent   
1  2010  Discharged - End of Sentence  25-34  Male       D Felony  Property   
2  2010                        Parole  35-44  Male       B Felony      Drug   
3  2010                        Parole  25-34  Male       B Felony     Other   
4  2010  Discharged - End of Sentence  35-44  Male       D Felony   Violent   

          subtype reoffend
0         Robbery      Yes
1           Theft      Yes
2     Trafficking      Yes
3  Other Criminal       No
4         Assault      Yes
```

[6]:

```
# Look at unique values for columns
clean_df.release.unique()
```

```
[6]: array(['Parole', 'Discharged - End of Sentence', 'Special Sentence', nan,
            'Interstate Compact Parole', 'Parole Granted',
            'Discharged - Expiration of Sentence',
            'Paroled w/Immediate Discharge', 'Paroled to Detainer - Iowa',
            'Paroled to Detainer - U.S. Marshall',
            'Paroled to Detainer - Out of State',
            'Released to Special Sentence', 'Paroled to Detainer - INS'],
           dtype=object)
```

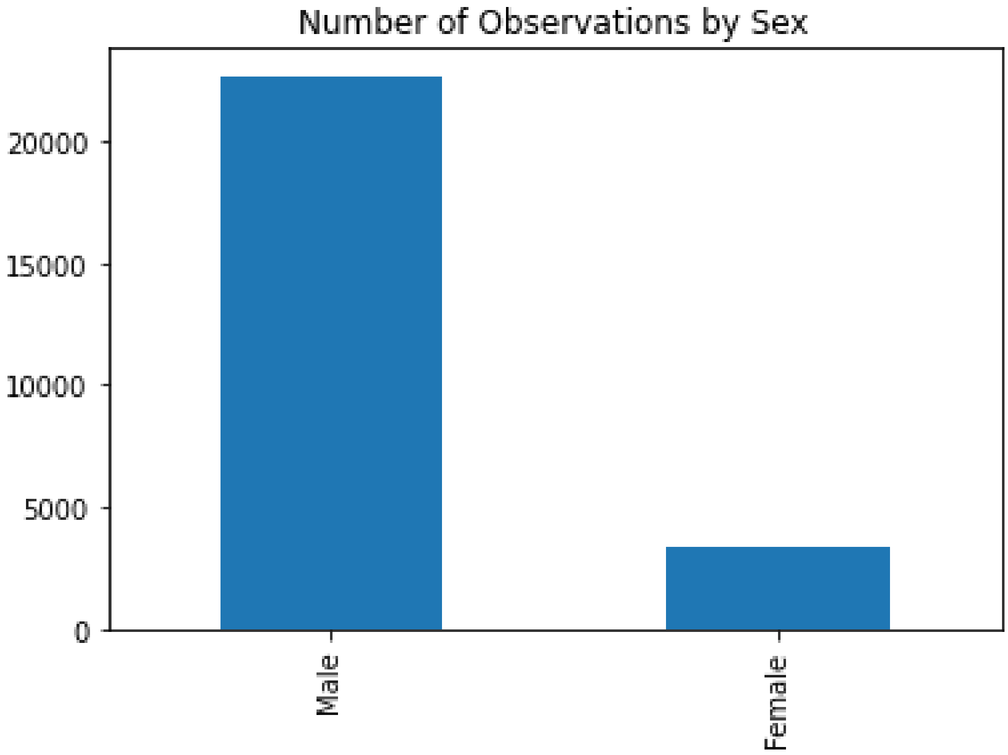
```
[7]: # Look at unique values for columns
     clean_df.classification.unique()
```

```
[7]: array(['C Felony', 'D Felony', 'B Felony', 'Felony - Enhanced',
            'Aggravated Misdemeanor', 'Other Felony (Old Code)',
            'Serious Misdemeanor', 'Sexual Predator Community Supervision',
            'Simple Misdemeanor', 'Felony - Enhancement to Original Penalty',
            'Special Sentence 2005', 'Felony - Mandatory Minimum',
            'Other Felony', 'A Felony', 'Other Misdemeanor'], dtype=object)
```

```
[5]: # The way records were kept changed over time. Combine values that are the same␣
      ↪but labeled differently.
     clean_df2 = clean_df.replace({'release': {'Discharged - End of Sentence':␣
      ↪'Discharged - Expiration of Sentence',
                                                'Parole': 'Parole Granted', 'Special␣
      ↪Sentence': 'Released to Special Sentence'}})

     clean_df2 = clean_df2.replace({'classification': {'Other Felony (Old Code)':␣
      ↪'Other Felony'}})

     # Replace nan with Unknown
     clean_df2['release'].fillna('Unknown', inplace=True)
```

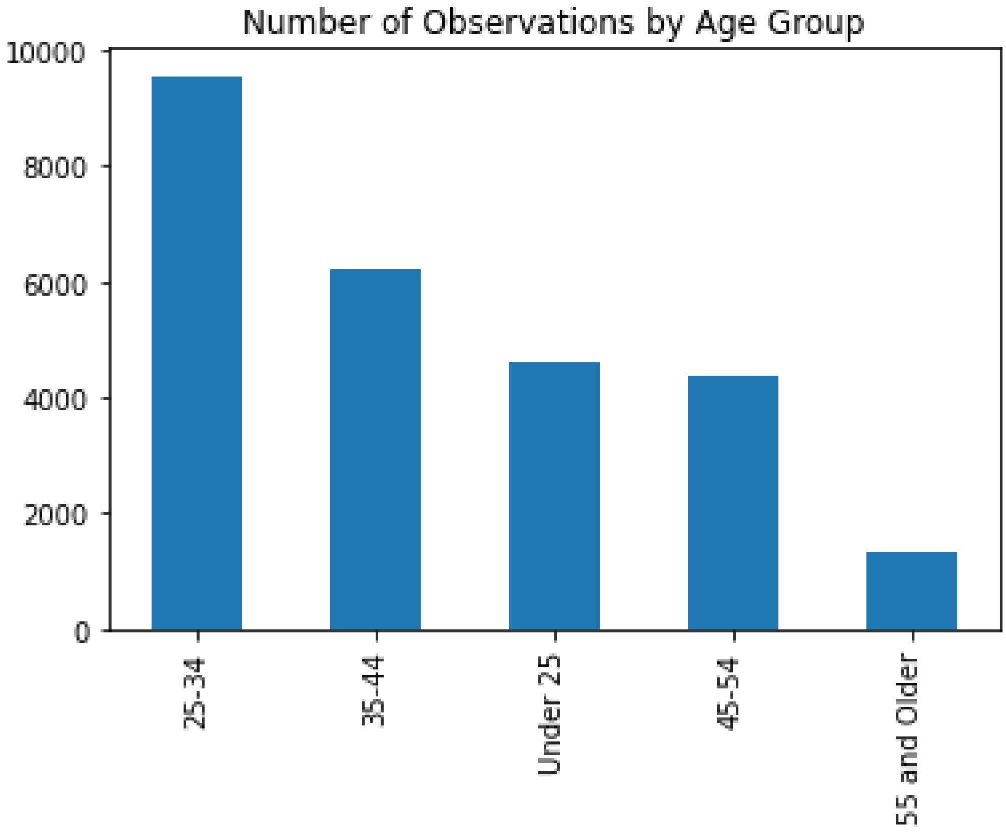
```
[6]: clean_df2.sex.value_counts().plot(kind="bar")
     plt.title("Number of Observations by Sex")
```

```
[6]: Text(0.5, 1.0, 'Number of Observations by Sex')
```



```
[10]: clean_df2.age.value_counts().plot(kind="bar")
      plt.title("Number of Observations by Age Group")
```

```
[10]: Text(0.5, 1.0, 'Number of Observations by Age Group')
```



[6]:
```
# Look to see how many nans are left and replace them with mode values
clean_df2.isna().sum()

# Replace nan in sex with mode
clean_df2['sex'].fillna('Male', inplace=True)

# Replace nan in age with mode
clean_df2['age'].fillna('25-34', inplace=True)
```

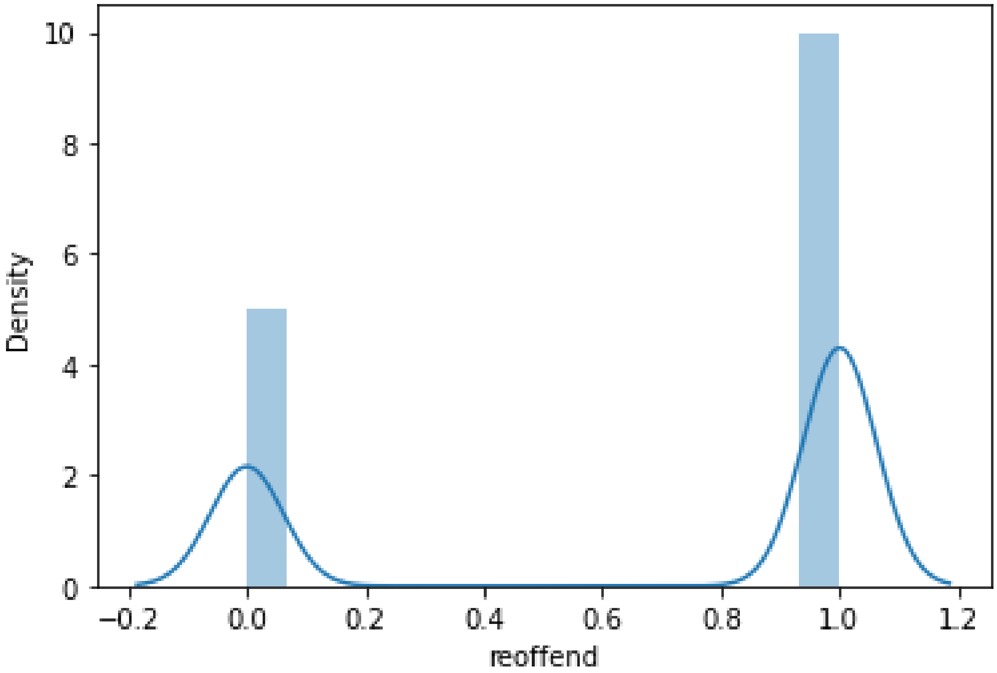
[7]:
```
# Create binary target vector indicating if class 0
clean_df2['reoffend'] = clean_df2['reoffend'].replace({"No": 0, "Yes": 1})
clean_df2.head()
```

[7]:
```
   year                               release    age   sex classification  \
0  2010                        Parole Granted  25-34  Male       C Felony
1  2010  Discharged - Expiration of Sentence  25-34  Male       D Felony
2  2010                        Parole Granted  35-44  Male       B Felony
3  2010                        Parole Granted  25-34  Male       B Felony
4  2010  Discharged - Expiration of Sentence  35-44  Male       D Felony
```

```
       type         subtype  reoffend
0   Violent         Robbery         1
1  Property           Theft         1
2      Drug     Trafficking         1
3     Other  Other Criminal         0
4   Violent         Assault         1
```

[12]:
```
# Look at density plot of target variable using seaborn
import seaborn as sns
sns.distplot(clean_df2.reoffend)
```

[12]:
```
<AxesSubplot:xlabel='reoffend', ylabel='Density'>
```



[8]:
```
# Write cleaned up dataframe to excel file
clean_df2.to_excel("recidivism_cleaned.xlsx")
```

[8]:
```
# Change categorical features to numerical values
#categorical data
categorical_cols = ['release', 'age', 'sex', 'classification', 'type',␣
 ↪'subtype']

# Use get dummies to one-hot encode the categorical columns
clean_df3 = pd.get_dummies(clean_df2, columns = categorical_cols)
clean_df3
```

[8]:

```
       year  reoffend  release_Discharged - Expiration of Sentence  \
0      2010         1                                            0
1      2010         1                                            1
2      2010         1                                            0
3      2010         0                                            0
4      2010         1                                            1
…     …       …                                          …
26015  2015         0                                            0
26016  2015         0                                            0
26017  2015         0                                            0
26018  2015         0                                            0
26019  2015         1                                            0

       release_Interstate Compact Parole  release_Parole Granted  \
0                                      0                       1
1                                      0                       0
2                                      0                       1
3                                      0                       1
4                                      0                       0
…                                  …                     …
26015                                  0                       0
26016                                  0                       0
26017                                  0                       1
26018                                  0                       0
26019                                  0                       0

       release_Paroled to Detainer - INS  release_Paroled to Detainer - Iowa  \
0                                      0                                   0
1                                      0                                   0
2                                      0                                   0
3                                      0                                   0
4                                      0                                   0
…                                  …                                 …
26015                                  1                                   0
26016                                  0                                   0
26017                                  0                                   0
26018                                  0                                   0
26019                                  0                                   0

       release_Paroled to Detainer - Out of State  \
0                                               0
1                                               0
2                                               0
3                                               0
4                                               0
…                                           …
26015                                           0
```

```
26016                                           0
26017                                           0
26018                                           0
26019                                           1

       release_Paroled to Detainer - U.S. Marshall  \
0                                                0
1                                                0
2                                                0
3                                                0
4                                                0
…                                            …
26015                                            0
26016                                            0
26017                                            0
26018                                            0
26019                                            0

       release_Paroled w/Immediate Discharge  …  subtype_Robbery  \
0                                          0  …                1
1                                          0  …                0
2                                          0  …                0
3                                          0  …                0
4                                          0  …                0
…                                      …  …              …
26015                                      0  …                0
26016                                      0  …                0
26017                                      0  …                0
26018                                      1  …                0
26019                                      0  …                0

       subtype_Sex  subtype_Sex Offender Registry/Residency  \
0                0                                        0
1                0                                        0
2                0                                        0
3                0                                        0
4                0                                        0
…            …                                      …
26015            0                                        0
26016            1                                        0
26017            0                                        0
26018            0                                        0
26019            0                                        0

       subtype_Special Sentence Revocation  subtype_Stolen Property  \
0                                        0                        0
1                                        0                        0
```

```
2                                        0                        0
3                                        0                        0
4                                        0                        0
…                                    …                      …
26015                                    0                        0
26016                                    0                        0
26017                                    0                        0
26018                                    0                        0
26019                                    0                        0

       subtype_Theft  subtype_Traffic  subtype_Trafficking  subtype_Vandalism  \
0                  0                0                    0                  0
1                  1                0                    0                  0
2                  0                0                    1                  0
3                  0                0                    0                  0
4                  0                0                    0                  0
…              …              …                  …                …
26015              0                0                    0                  0
26016              0                0                    0                  0
26017              0                1                    0                  0
26018              1                0                    0                  0
26019              0                0                    0                  0

       subtype_Weapons
0                    0
1                    0
2                    0
3                    0
4                    0
…                …
26015                0
26016                0
26017                0
26018                0
26019                0

[26020 rows x 64 columns]
```

[9]:
```
# Move the reoffend column to the last column
clean_df4 = clean_df3[[c for c in clean_df3 if c not in ['reoffend']]
       + ['reoffend']]
clean_df4
```

[9]:
```
       year  release_Discharged - Expiration of Sentence  \
0      2010                                            0
1      2010                                            1
2      2010                                            0
```

```
3      2010                                            0
4      2010                                            1
…     …                                          …
26015  2015                                            0
26016  2015                                            0
26017  2015                                            0
26018  2015                                            0
26019  2015                                            0

       release_Interstate Compact Parole  release_Parole Granted  \
0                                      0                       1
1                                      0                       0
2                                      0                       1
3                                      0                       1
4                                      0                       0
…                                  …                     …
26015                                  0                       0
26016                                  0                       0
26017                                  0                       1
26018                                  0                       0
26019                                  0                       0

       release_Paroled to Detainer - INS  release_Paroled to Detainer - Iowa  \
0                                      0                                   0
1                                      0                                   0
2                                      0                                   0
3                                      0                                   0
4                                      0                                   0
…                                  …                                 …
26015                                  1                                   0
26016                                  0                                   0
26017                                  0                                   0
26018                                  0                                   0
26019                                  0                                   0

       release_Paroled to Detainer - Out of State  \
0                                               0
1                                               0
2                                               0
3                                               0
4                                               0
…                                           …
26015                                           0
26016                                           0
26017                                           0
26018                                           0
26019                                           1
```

```
       release_Paroled to Detainer - U.S. Marshall  \
0                                                0
1                                                0
2                                                0
3                                                0
4                                                0
…                                           …
26015                                            0
26016                                            0
26017                                            0
26018                                            0
26019                                            0

       release_Paroled w/Immediate Discharge  \
0                                          0
1                                          0
2                                          0
3                                          0
4                                          0
…                                      …
26015                                      0
26016                                      0
26017                                      0
26018                                      1
26019                                      0

       release_Released to Special Sentence  …  subtype_Sex  \
0                                         0  …            0
1                                         0  …            0
2                                         0  …            0
3                                         0  …            0
4                                         0  …            0
…                                     …  …          …
26015                                     0  …            0
26016                                     1  …            1
26017                                     0  …            0
26018                                     0  …            0
26019                                     0  …            0

       subtype_Sex Offender Registry/Residency  \
0                                            0
1                                            0
2                                            0
3                                            0
4                                            0
…                                        …
```

```
26015                                        0
26016                                        0
26017                                        0
26018                                        0
26019                                        0

       subtype_Special Sentence Revocation  subtype_Stolen Property  \
0                                        0                        0
1                                        0                        0
2                                        0                        0
3                                        0                        0
4                                        0                        0
…                                    …                      …
26015                                    0                        0
26016                                    0                        0
26017                                    0                        0
26018                                    0                        0
26019                                    0                        0

       subtype_Theft  subtype_Traffic  subtype_Trafficking  subtype_Vandalism  \
0                  0                0                    0                  0
1                  1                0                    0                  0
2                  0                0                    1                  0
3                  0                0                    0                  0
4                  0                0                    0                  0
…              …              …                  …                …
26015              0                0                    0                  0
26016              0                0                    0                  0
26017              0                1                    0                  0
26018              1                0                    0                  0
26019              0                0                    0                  0

       subtype_Weapons  reoffend
0                    0         1
1                    0         1
2                    0         1
3                    0         0
4                    0         1
…                …       …
26015                0         0
26016                0         0
26017                0         0
26018                0         0
26019                0         1

[26020 rows x 64 columns]
```

[10]:

```
# Split data into features and target data
features = clean_df4.iloc[: , :-1]
target = clean_df4.iloc[:,-1:]
```

[11]:

```
# Use RandomForestClassifier for feature selection because the dataset contains␣
 ↪both numeric and categorical features
from sklearn.ensemble import RandomForestClassifier
from sklearn.feature_selection import SelectFromModel

# Specify random forest instance, indicating the number of trees
sel = SelectFromModel(RandomForestClassifier(n_estimators = 100))

# Use selectFromModel object to automatically select features
sel.fit(features, target.values.ravel())

# Make a list and count the selected features
selected_feat= features.columns[(sel.get_support())]

print("Best Number of Features: ", len(selected_feat))
print("List of Feature Names: ", selected_feat)
```

```
Best Number of Features:  16
List of Feature Names:  Index(['year', 'release_Discharged - Expiration of
Sentence',
       'release_Parole Granted', 'release_Released to Special Sentence',
       'release_Unknown', 'age_25-34', 'age_35-44', 'age_45-54',
       'age_55 and Older', 'age_Under 25', 'sex_Female', 'sex_Male',
       'classification_Aggravated Misdemeanor', 'classification_C Felony',
       'classification_D Felony', 'type_Violent'],
      dtype='object')
```

[12]:

```
# Create a features dataframe that has the identified best features
best_features = features[['year', 'release_Discharged - Expiration of Sentence',
       'release_Parole Granted', 'release_Released to Special Sentence',
       'release_Unknown', 'age_25-34', 'age_35-44', 'age_45-54',
       'age_55 and Older', 'age_Under 25', 'sex_Female', 'sex_Male',
       'classification_Aggravated Misdemeanor', 'classification_C Felony',
       'classification_D Felony', 'type_Violent']]
best_features
```

[12]:

```
       year  release_Discharged - Expiration of Sentence  \
0      2010                                             0
1      2010                                             1
2      2010                                             0
3      2010                                             0
4      2010                                             1
…      …                                             …
```

```
26015  2015                                            0
26016  2015                                            0
26017  2015                                            0
26018  2015                                            0
26019  2015                                            0

       release_Parole Granted  release_Released to Special Sentence  \
0                           1                                     0
1                           0                                     0
2                           1                                     0
3                           1                                     0
4                           0                                     0
…                       …                                   …
26015                       0                                     0
26016                       0                                     1
26017                       1                                     0
26018                       0                                     0
26019                       0                                     0

       release_Unknown  age_25-34  age_35-44  age_45-54  age_55 and Older  \
0                    0          1          0          0                 0
1                    0          1          0          0                 0
2                    0          0          1          0                 0
3                    0          1          0          0                 0
4                    0          0          1          0                 0
…                …        …        …        …               …
26015                0          0          0          0                 0
26016                0          0          1          0                 0
26017                0          1          0          0                 0
26018                0          1          0          0                 0
26019                0          0          1          0                 0

       age_Under 25  sex_Female  sex_Male  \
0                 0           0         1
1                 0           0         1
2                 0           0         1
3                 0           0         1
4                 0           0         1
…             …         …       …
26015             1           0         1
26016             0           0         1
26017             0           1         0
26018             0           0         1
26019             0           0         1

       classification_Aggravated Misdemeanor  classification_C Felony  \
0                                          0                        1
```

```
1                                        0                        0
2                                        0                        0
3                                        0                        0
4                                        0                        0
…                                    …                      …
26015                                    0                        1
26016                                    0                        1
26017                                    1                        0
26018                                    0                        0
26019                                    0                        0

       classification_D Felony  type_Violent
0                            0             1
1                            1             0
2                            0             0
3                            0             0
4                            1             1
…                        …            …
26015                        0             1
26016                        0             1
26017                        0             0
26018                        1             0
26019                        1             1

[26020 rows x 16 columns]
```

[13]:

```python
# reset index
best_features.reset_index(inplace=True, drop=True)
```

[14]:

```python
# Train Test Split
from sklearn.model_selection import train_test_split

X_train, X_val, y_train, y_val = train_test_split(best_features, target,␣
 ↪test_size =0.3, random_state=11)
```

[23]:

```python
# number of samples in each set
print("No. of samples in training set: ", X_train.shape[0])
print("No. of samples in validation set:", X_val.shape[0])
```

```
No. of samples in training set:  18214
No. of samples in validation set: 7806
```

[24]:

```python
# Check if training target vector had imbalanced classes
y_train['reoffend'].value_counts()
```

[24]:

```
0    12186
1     6028
```

```
Name: reoffend, dtype: int64
```

Upsampling Training Data

[15]:

```
# To fix imbalanced classes, upsample training data.
from sklearn.utils import resample

#combine them back for resampling
train_data = pd.concat([X_train, y_train], axis=1)

# separate minority and majority classes
negative = train_data[train_data.reoffend==0]
positive = train_data[train_data.reoffend==1]

# upsample minority
pos_upsampled = resample(positive,
 replace=True, # sample with replacement
 n_samples=len(negative), # match number in majority class
 random_state=27) # reproducible results

# combine majority and upsampled minority
upsampled = pd.concat([negative, pos_upsampled])

# check new class counts
upsampled.reoffend.value_counts()
```

[15]:

```
0    12186
1    12186
Name: reoffend, dtype: int64
```

[16]:

```
# Reassign upsampled training features and training target
X_train = upsampled.drop("reoffend",axis=1)
y_train = upsampled["reoffend"]
```

[17]:

```
# number of samples in each set
print("No. of samples in training set: ", X_train.shape[0])
print("No. of samples in validation set:", X_val.shape[0])
```

```
No. of samples in training set:  24372
No. of samples in validation set: 7806
```

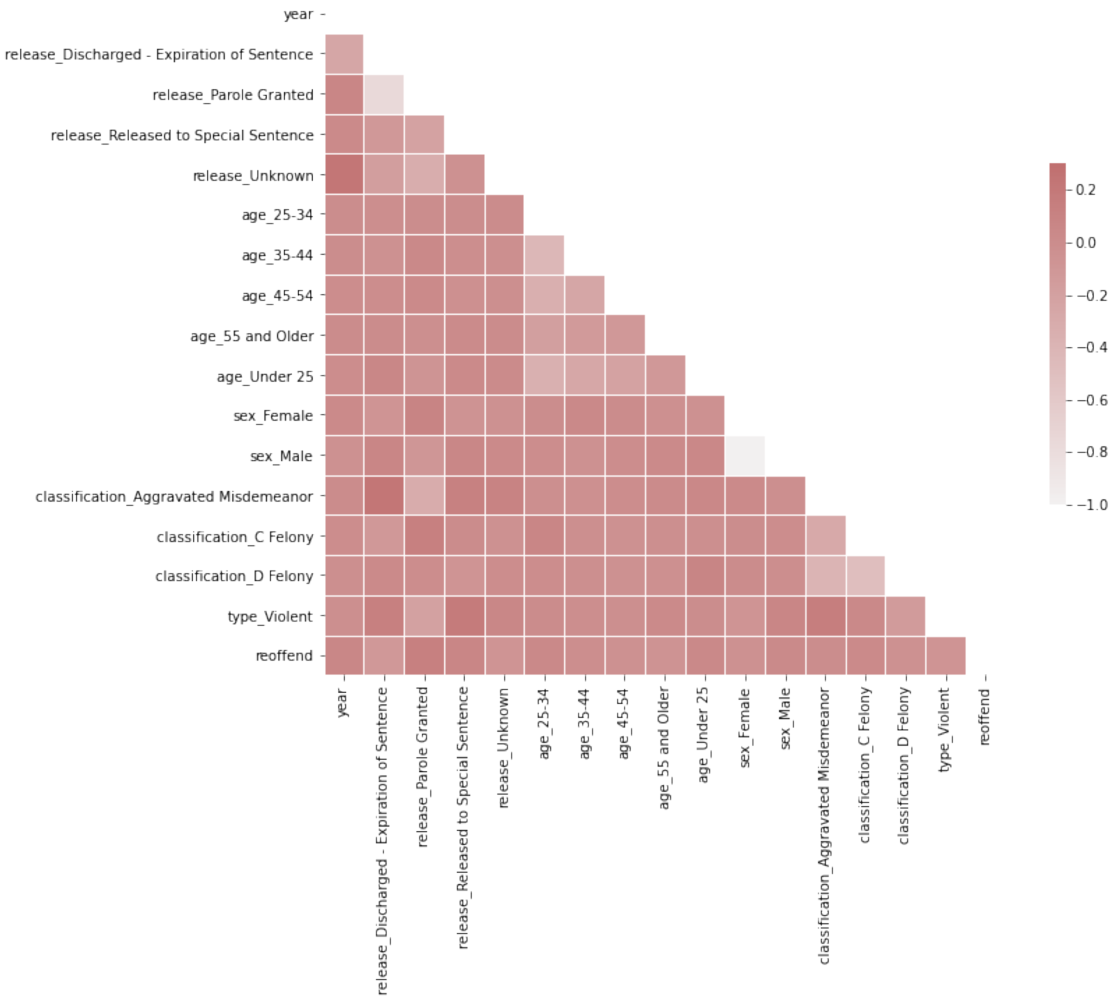
Correlation Matrix of Best Features

[49]:

```
# Compute the correlation matrix
corr = train_data.corr()

# Generate a mask for the upper triangle
mask = np.triu(np.ones_like(corr, dtype=bool))
```

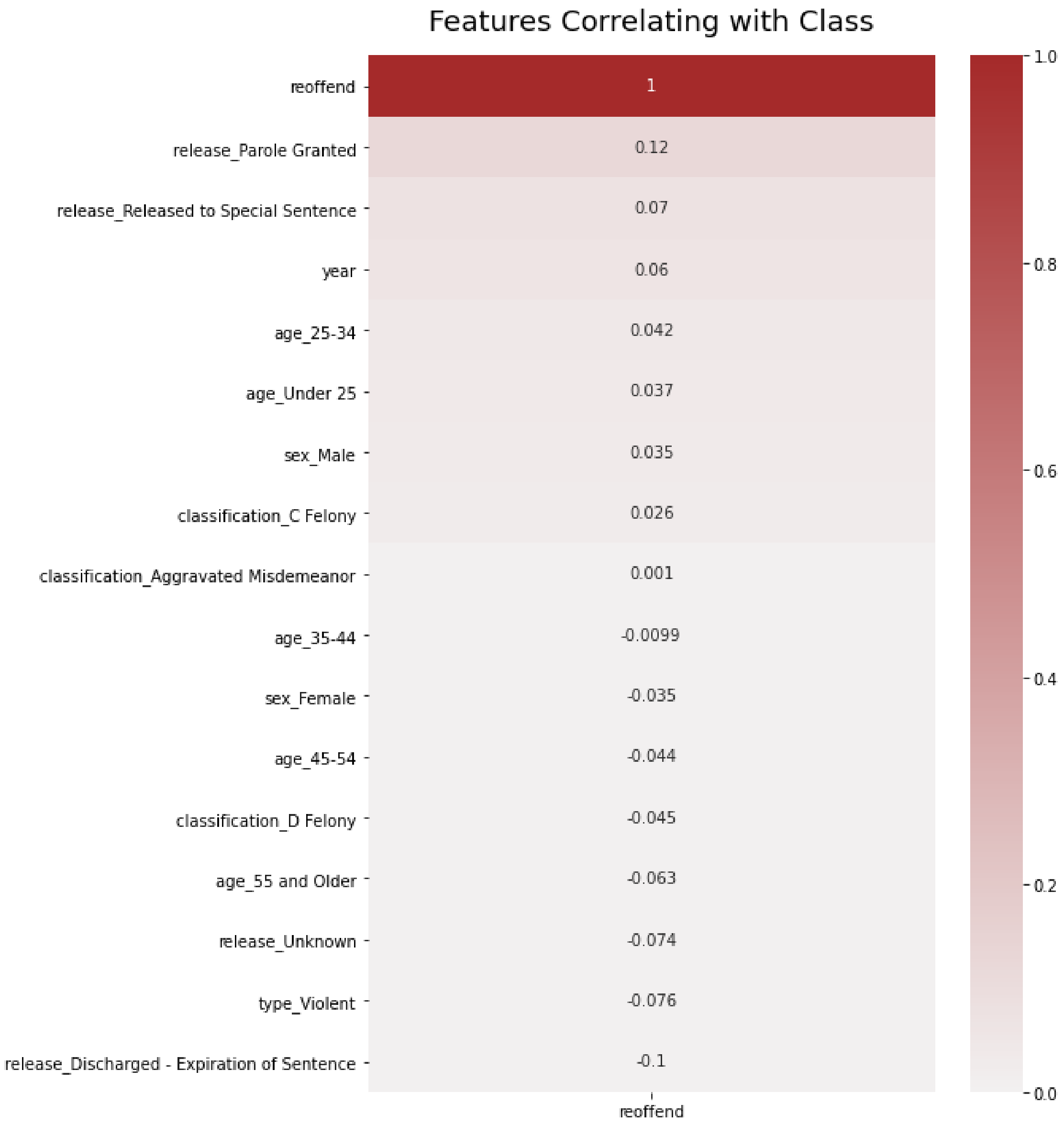
```
# Set up the matplotlib figure
f, ax = plt.subplots(figsize=(11, 9))

cmap = sns.light_palette("Brown", reverse=False, as_cmap=True)

# Draw the heatmap with the mask and correct aspect ratio
sns.heatmap(corr, mask=mask, cmap=cmap, vmax=.3, center=0,
            square=True, linewidths=.5, cbar_kws={"shrink": .5})
```

```
[49]: <AxesSubplot:>
```



```
[50]: # What I'm interested is any correlation the attributes might have with the␣
       ↪Class
      # Compute the correlation matrix
      corr = train_data.corr()
```

```
plt.figure(figsize=(8, 12))
heatmap = sns.heatmap(train_data.corr()[['reoffend']].
 ↪sort_values(by='reoffend', ascending=False), vmin=0, vmax=1, annot=True,␣
 ↪cmap=cmap)
heatmap.set_title('Features Correlating with Class', fontdict={'fontsize':18},␣
 ↪pad=16);
```



```
[ ]:
```

Baseline Classification Model

[19]:
```
# Create a baseline classification model to use for comparison
from sklearn.dummy import DummyClassifier

# Create classifier
dummy = DummyClassifier(strategy='uniform', random_state=0)

# Train model
dummy.fit(X_train, y_train.values.ravel())

# Get accuracy score
dummy.score(X_val, y_val.values.ravel())
```

[19]:
```
0.49743786830643094
```

Random Forest Model

[23]:
```
from sklearn.ensemble import RandomForestClassifier

# Create random forest classifier object
randomforest = RandomForestClassifier(random_state=0, n_jobs=-1,␣
 ↪class_weight="balanced")

#Train model
rf_model = randomforest.fit(X_train, y_train.values.ravel())
```

[24]:
```
# view model accuracy score
rf_model.score(X_val, y_val.values.ravel())
```

[24]:
```
0.5807071483474251
```

Hyperparameter Tuning

[34]:
```
# Use RandomizedSearchCV to tune hyperparameters because it is computationally␣
 ↪cheaper than an exhaustive search.
from sklearn.model_selection import RandomizedSearchCV

# Create range for number of trees in random forest
n_estimators = [200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000]

# Create range for number of features to consider at every split
max_features = ['auto', 'sqrt']

# Create range for maximum number of levels in tree
max_depth = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, None]

# Create range for minimum number of samples required to split a node
min_samples_split = [2, 5, 10]
```

```
# Create range for minimum number of samples required at each leaf node
min_samples_leaf = [1, 2, 4]

# Define method of selecting samples for training each tree
bootstrap = [True, False]

# Create hyperparameter options
hyperparameters = dict(n_estimators=n_estimators, max_features=max_features,␣
 ↪max_depth=max_depth,
                       min_samples_split=min_samples_split,␣
 ↪min_samples_leaf=min_samples_leaf, bootstrap=bootstrap)

# Create randomized search
randomizedsearch = RandomizedSearchCV(rf_model, hyperparameters,␣
 ↪random_state=1, n_iter=100, cv=5, verbose=0, n_jobs=-1)

# Fit randomized search
best_rf_model = randomizedsearch.fit(X_train, y_train.values.ravel())
```

[35]:

```
# View best hyperparameters
best_rf_model.best_params_
```

[35]:

```
{'n_estimators': 400,
 'min_samples_split': 5,
 'min_samples_leaf': 4,
 'max_features': 'sqrt',
 'max_depth': 10,
 'bootstrap': True}
```

[36]:

```
# Get accuracy score for best_model and compare to intial model
best_rf_model.score(X_val, y_val.values.ravel())
```

[36]:

```
0.6026133743274404
```

Adaboost

[20]:

```
# Try to use boosting in attempt to improve the model
from sklearn.ensemble import AdaBoostClassifier
from sklearn.model_selection import cross_val_score

# Create adaboost tree classifier object
adaboost = AdaBoostClassifier(random_state=0)

# Train model on the important features data set
ab_model = adaboost.fit(X_train, y_train.values.ravel())

# Cross-validate Adaboost model
```

```
cvScore = cross_val_score(ab_model, X_val, y_val.values.ravel(),␣
 ↪scoring="accuracy")
print("CV Scores:", cvScore)
print("CV Scores Mean:", cvScore.mean())
```

```
CV Scores: [0.66133163 0.66495836 0.66367713 0.65406791 0.66303652]
CV Scores Mean: 0.661414307286852
```

[30]:

```
# Tune hyperparameters with Adaboost
from sklearn.model_selection import GridSearchCV

# Define hyperparameter candidates
param_grid = {'n_estimators': [100, 200, 400, 600, 800, 1000],
              'learning_rate': [0.001, 0.01, 0.1, 0.2, 0.5]}

# Instatiate GridSearchCV model
gs_ab = GridSearchCV(AdaBoostClassifier(), param_grid = param_grid)

# Train model
gs_ab_model = gs_ab.fit(X_train, y_train.values.ravel())

# Cross-validate model
cvScore2 = cross_val_score(gs_ab_model, X_val, y_val.values.ravel(),␣
 ↪scoring="accuracy")
print("CV Scores:", cvScore2)
print("CV Scores Mean:", cvScore2.mean())
```

```
CV Scores: [0.66517286 0.66495836 0.66303652 0.65150545 0.66303652]
CV Scores Mean: 0.6615419381351295
```

[32]:

```
# View best hyperparameters
gs_ab_model.best_params_
```

[32]:

```
{'learning_rate': 0.1, 'n_estimators': 600}
```

Skip to Here to Run Final Model & Results

[18]:

```
### After upsampling & changing target factoring run Adaboost with best params
from sklearn.ensemble import AdaBoostClassifier
from sklearn.model_selection import cross_val_score

# Create classifier object
myBoost = AdaBoostClassifier(random_state=0, n_estimators=600, learning_rate=0.
 ↪1)

# Train model
final_model = myBoost.fit(X_train, y_train.values.ravel())
```

```python

# Cross-validate model
cvScore2 = cross_val_score(final_model, X_val, y_val.values.ravel(),␣
 ↪scoring="accuracy")
print("CV Scores:", cvScore2)
print("CV Scores Mean:", cvScore2.mean())
```

```
CV Scores: [0.66389245 0.66559898 0.66367713 0.65663037 0.6623959 ]
CV Scores Mean: 0.6624389631716102
```

[19]:

```python
# Compute predictions
from sklearn.metrics import roc_auc_score, roc_curve, auc, plot_roc_curve,\
    precision_score, recall_score, f1_score, classification_report,␣
 ↪precision_recall_curve, plot_precision_recall_curve, \
    average_precision_score

# compute predict probabilties on test set
# only need the probabilities for the positive class only
y_pred_probs = final_model.predict_proba(X_val)[:, 1]

# compute predictions on test set
y_preds = final_model.predict(X_val)
```

[20]:

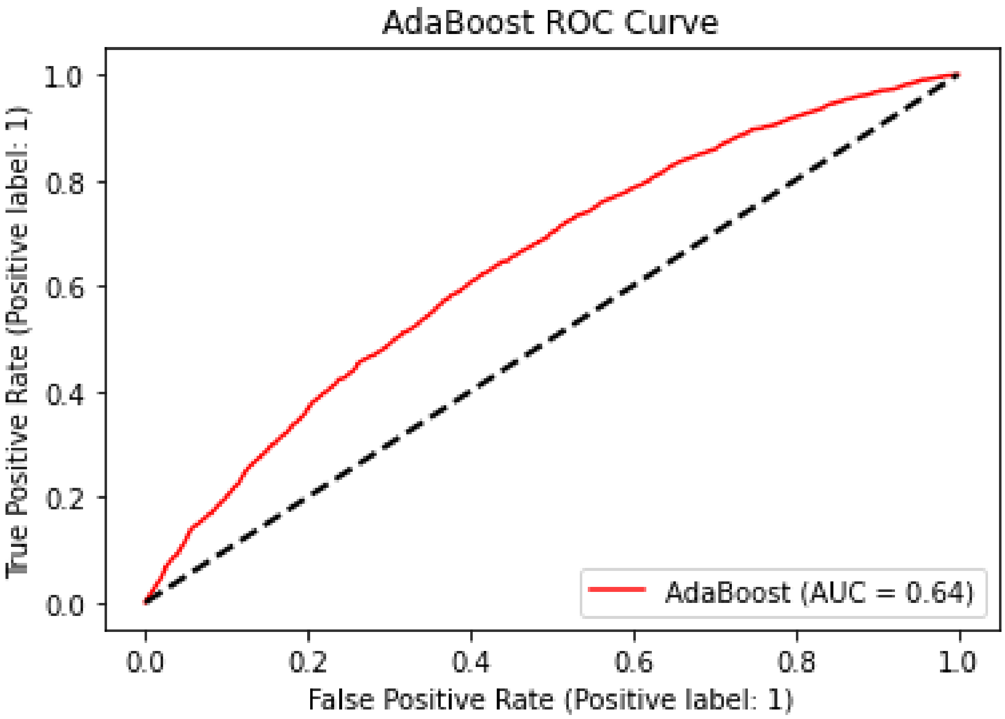
```python
# Compute AUC
auc_score = roc_auc_score(y_val, y_pred_probs)
round(auc_score,4)
```

[20]:

```
0.6427
```

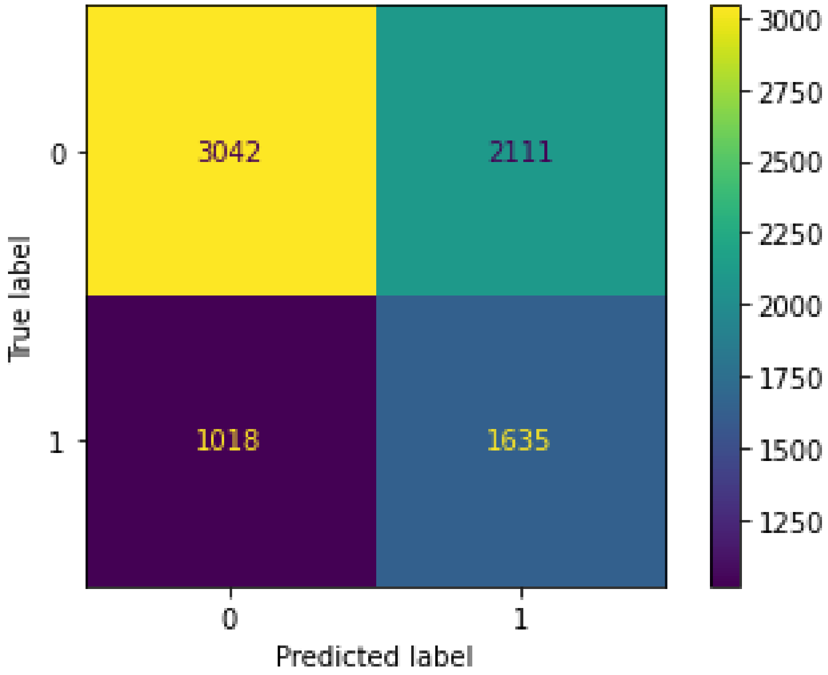
[22]:

```python
# Plot ROC Curve
lw = 2
plot_roc_curve(final_model, X_val, y_val, name = 'AdaBoost',
               color="red", pos_label=None)
plt.plot([0, 1], [0, 1], 'k--', lw=lw)
plt.title('AdaBoost ROC Curve')
plt.show()
```



[23]:

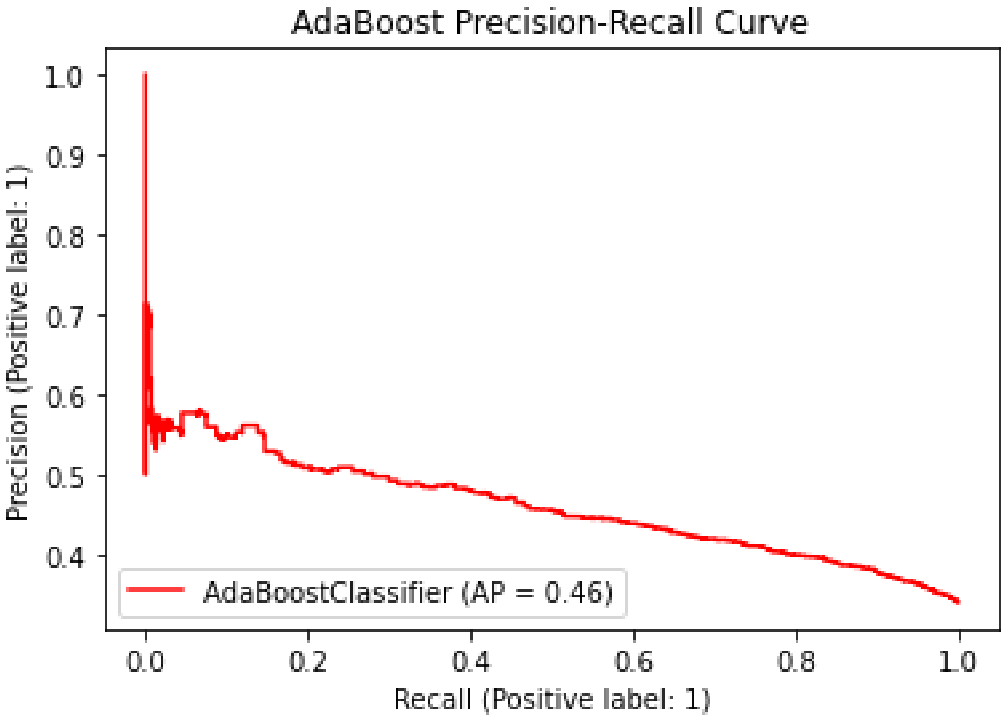
```
# Plot predictions on the test set
from sklearn.metrics import confusion_matrix
from sklearn.metrics import ConfusionMatrixDisplay

# Create confusion matrix
cm = confusion_matrix(y_val, y_preds)
cm_display = ConfusionMatrixDisplay(cm).plot()
```



[24]:
```
# Build Classification report
print(classification_report(y_val, y_preds))
```

```
              precision    recall  f1-score   support

           0       0.75      0.59      0.66      5153
           1       0.44      0.62      0.51      2653

    accuracy                           0.60      7806
   macro avg       0.59      0.60      0.59      7806
weighted avg       0.64      0.60      0.61      7806
```

[26]:
```
# Compute Average Precision Score
y_score = final_model.decision_function(X_val)

average_precision = average_precision_score(y_val, y_score)

# Plot PR curve
disp = plot_precision_recall_curve(final_model, X_val, y_val, name=None,␣
 ↪pos_label=1, color="red")
disp.ax_.set_title('AdaBoost Precision-Recall Curve')
plt.show()
```

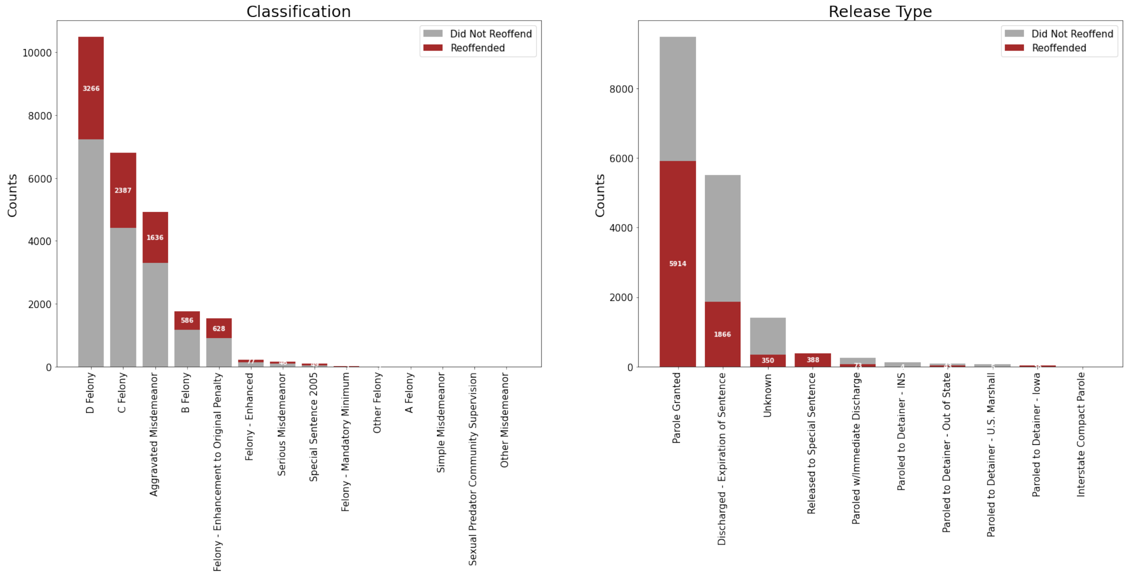


Additional Visualizations for Presentation Below

[49]:
```
plt.rcParams['figure.figsize'] = (30, 10)

# make subplots
fig, axes = plt.subplots(nrows = 1, ncols = 2)

# make the data for Classification to feed into the visulizer
Class_No = clean_df2.replace({'reoffend': {1: 'reoffend', 0:␣
 ↪'Yes'}})[clean_df2['reoffend']==1]['classification'].value_counts()
Class_Yes = clean_df2.replace({'reoffend': {1: 'reoffend', 0:␣
 ↪'No'}})[clean_df2['reoffend']==0]['classification'].value_counts()
Class_Yes = Class_Yes.reindex(index = Class_No.index)
# make the bar plot
p5 = axes[0].bar(Class_No.index, Class_No.values, color='darkgray')
p6 = axes[0].bar(Class_Yes.index, Class_Yes.values, bottom=Class_No.values,␣
 ↪color='brown')
axes[0].set_title('Classification', fontsize=25)
axes[0].set_ylabel('Counts', fontsize=20)
axes[0].tick_params(axis='y', labelsize=15)
axes[0].tick_params(axis='x', labelsize=15, rotation=90)
axes[0].legend((p5[0], p6[0]), ('Did Not Reoffend', 'Reoffended'), fontsize =␣
 ↪15, loc='upper right')
```

```
axes[0].bar_label(p6, label_type='center', color='white', weight="bold")

# make the data for Release Type to feed into the visulizer
Release_No = clean_df2.replace({'reoffend': {1: 'reoffend', 0:␣
 ↪'Yes'}})[clean_df2['reoffend']==1]['release'].value_counts()
Release_Yes = clean_df2.replace({'reoffend': {1: 'reoffend', 0:␣
 ↪'No'}})[clean_df2['reoffend']==0]['release'].value_counts()
Release_Yes = Release_Yes.reindex(index = Release_No.index)
# make the bar plot
p9 = axes[1].bar(Release_No.index, Release_No.values, color='darkgray')
p10 = axes[1].bar(Release_Yes.index, Release_Yes.values, color='brown')
axes[1].set_title('Release Type', fontsize=25)
axes[1].set_ylabel('Counts', fontsize=20)
axes[1].tick_params(axis='y', labelsize=15)
axes[1].tick_params(axis='x', labelsize=15, rotation=90)
axes[1].legend((p9[0], p10[0]), ('Did Not Reoffend', 'Reoffended'), fontsize =␣
 ↪15, loc='upper right')
axes[1].bar_label(p10, label_type='center', color='white', weight="bold")
```

```
[49]: [Text(0, 0, '5914'),
       Text(0, 0, '1866'),
       Text(0, 0, '350'),
       Text(0, 0, '388'),
       Text(0, 0, '73'),
       Text(0, 0, '4'),
       Text(0, 0, '43'),
       Text(0, 0, '5'),
       Text(0, 0, '38'),
       Text(0, 0, 'nan')]
```

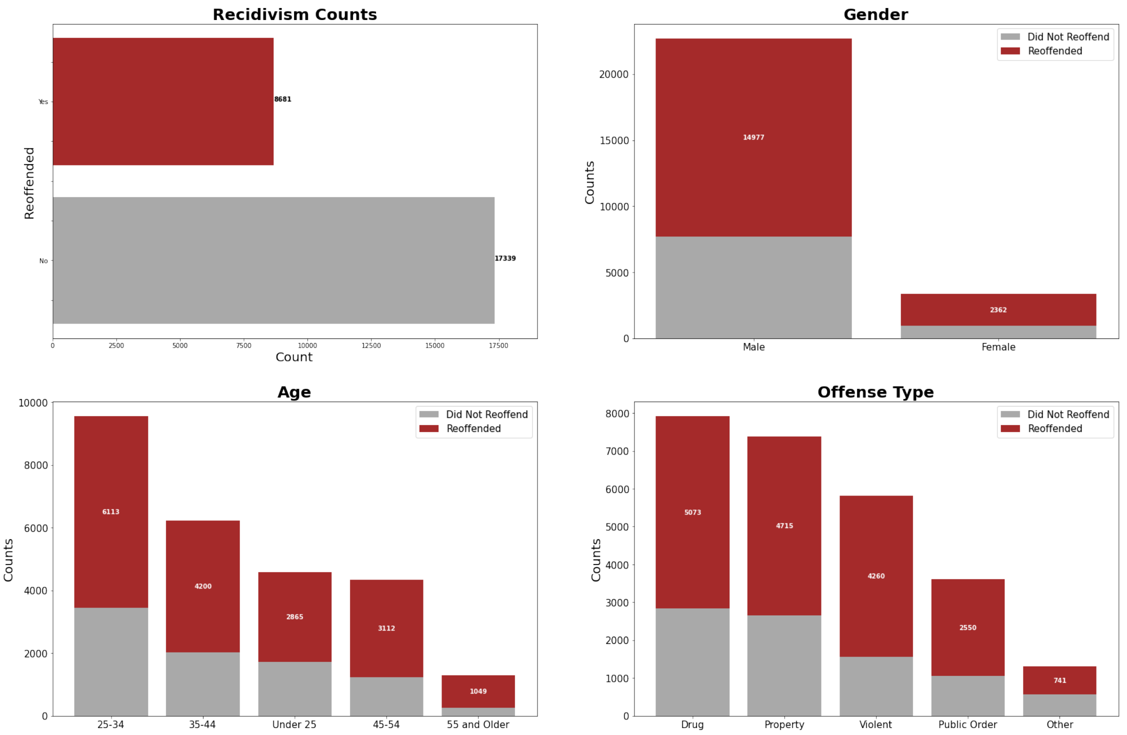


[51]:

```
# Split the stacked bar charts apart
#set up the figure size
plt.rcParams['figure.figsize'] = (30, 20)

# make subplots
fig, axes = plt.subplots(nrows = 2, ncols = 2)

# make the data for Reoffend to feed into the visulizer
axes[0, 0].barh(recid_counts.index, recid_counts['Return to Prison'],␣
 ↪color=['darkgray', 'brown'])
axes[0, 0].set_ylabel("Reoffended", size=20)
axes[0, 0].set_xlabel("Count", size=20)
axes[0, 0].set_title("Recidivism Counts", size=25, weight="bold")
axes[0, 0].set_yticklabels(("", "", "No","", "", "", "Yes"))
axes[0, 0].set_xlim(0, 19000)
for index, value in enumerate(recid_counts['Return to Prison']):
    axes[0, 0].text(value, index, str(value), weight="bold")

# make the data for Gender to feed into the visulizer
Sex_No = clean_df2.replace({'reoffend': {1: 'reoffend', 0:␣
 ↪'Yes'}})[clean_df2['reoffend']==1]['sex'].value_counts()
Sex_Yes = clean_df2.replace({'reoffend': {1: 'reoffend', 0:␣
 ↪'No'}})[clean_df2['reoffend']==0]['sex'].value_counts()
Sex_Yes = Sex_Yes.reindex(index = Sex_No.index)
# make the bar plot
p3 = axes[0, 1].bar(Sex_No.index, Sex_No.values, color='darkgray')
p4 = axes[0, 1].bar(Sex_Yes.index, Sex_Yes.values, bottom=Sex_No.values,␣
 ↪color='brown')
axes[0, 1].set_title('Gender', fontsize=25, weight="bold")
axes[0, 1].set_ylabel('Counts', fontsize=20)
axes[0, 1].tick_params(axis='both', labelsize=15)
axes[0, 1].legend((p3[0], p4[0]), ('Did Not Reoffend', 'Reoffended'), fontsize␣
 ↪= 15)
axes[0, 1].bar_label(p4, label_type='center', color='white', weight="bold")

# make the data for Age to feed into the visulizer
Age_No = clean_df2.replace({'reoffend': {1: 'reoffend', 0:␣
 ↪'Yes'}})[clean_df2['reoffend']==1]['age'].value_counts()
Age_Yes = clean_df2.replace({'reoffend': {1: 'reoffend', 0:␣
 ↪'No'}})[clean_df2['reoffend']==0]['age'].value_counts()
Age_Yes = Age_Yes.reindex(index = Age_No.index)
# make the bar plot
p1 = axes[1, 0].bar(Age_No.index, Age_No.values, color='darkgray')
p2 = axes[1, 0].bar(Age_Yes.index, Age_Yes.values, bottom=Age_No.values,␣
 ↪color='brown')
```

```
axes[1, 0].set_title('Age', fontsize=25, weight="bold")
axes[1, 0].set_ylabel('Counts', fontsize=20)
axes[1, 0].tick_params(axis='both', labelsize=15)
axes[1, 0].legend((p1[0], p2[0]), ('Did Not Reoffend', 'Reoffended'), fontsize␣
 ↪= 15)
axes[1, 0].bar_label(p2, label_type='center', color='white', weight="bold")

# make the data for Offense Type to feed into the visulizer
Offense_No = clean_df2.replace({'reoffend': {1: 'reoffend', 0:␣
 ↪'Yes'}})[clean_df2['reoffend']==1]['type'].value_counts()
Offense_Yes = clean_df2.replace({'reoffend': {1: 'reoffend', 0:␣
 ↪'No'}})[clean_df2['reoffend']==0]['type'].value_counts()
Offense_Yes = Offense_Yes.reindex(index = Offense_No.index)
# make the bar plot
p7 = axes[1, 1].bar(Offense_No.index, Offense_No.values, color='darkgray')
p8 = axes[1, 1].bar(Offense_Yes.index, Offense_Yes.values, bottom=Offense_No.
 ↪values, color='brown')
axes[1, 1].set_title('Offense Type', fontsize=25, weight="bold")
axes[1, 1].set_ylabel('Counts', fontsize=20)
axes[1, 1].tick_params(axis='both', labelsize=15)
axes[1, 1].legend((p7[0], p8[0]), ('Did Not Reoffend', 'Reoffended'), fontsize␣
 ↪= 15)
axes[1, 1].bar_label(p8, label_type='center', color='white', weight="bold")

plt.show()
```

```
<ipython-input-51-ed512fe37307>:13: UserWarning: FixedFormatter should only be
used together with FixedLocator
  axes[0, 0].set_yticklabels(("", "", "No","", "", "", "Yes"))
```



[56]:

```
# Create table with Pearson Chi-square and Cramers V between best features and␣
 ↪target variable. (From Christine).
import researchpy as rp

# Create new DF to store results
new_test = pd.DataFrame(columns=['feature','Pearson Chi-square',\
                                  'p-value', 'Cramers V',\
                                  'Cramer_val'])
# Define columns to use
cols = ['release', 'year', 'age', 'sex', 'classification', 'type', 'subtype']

# Iterate through columns and calculate statistics
for i in cols:
    crosstab, test_results, expected = rp.crosstab(
        clean_df2[i], clean_df2['reoffend'],
        test= "chi-square",
        expected_freqs= True,
        prop= "cell")

    # Create new var to append to new test df
    if test_results.iloc[2,1] > .25:
           new_var = 'Very Strong'
    elif test_results.iloc[2,1] > .15:
        new_var = 'Strong'
    elif test_results.iloc[2,1] > .10:
           new_var = 'Moderate'
    elif test_results.iloc[2,1] > .05:
        new_var = 'Weak'
    else:
        new_var = 'No or very weak'

    # Append row  to new_test DF
    new_row = {'feature':i, \
               'Pearson Chi-square': test_results.iloc[0,1], \
                'p-value': test_results.iloc[1,1],\
                'Cramers V': test_results.iloc[2,1],\
                'Cramer_val': new_var}

    new_test = new_test.append(new_row, ignore_index=True)

# https://www.pythonfordatascience.org/chi-square-test-of-independence-python/
# Phi and Cramer's V     Interpretation
# >0.25     Very strong
# >0.15     Strong
# >0.10     Moderate
# >0.05     Weak
# >0     No or very weak

# View results
new_test
```

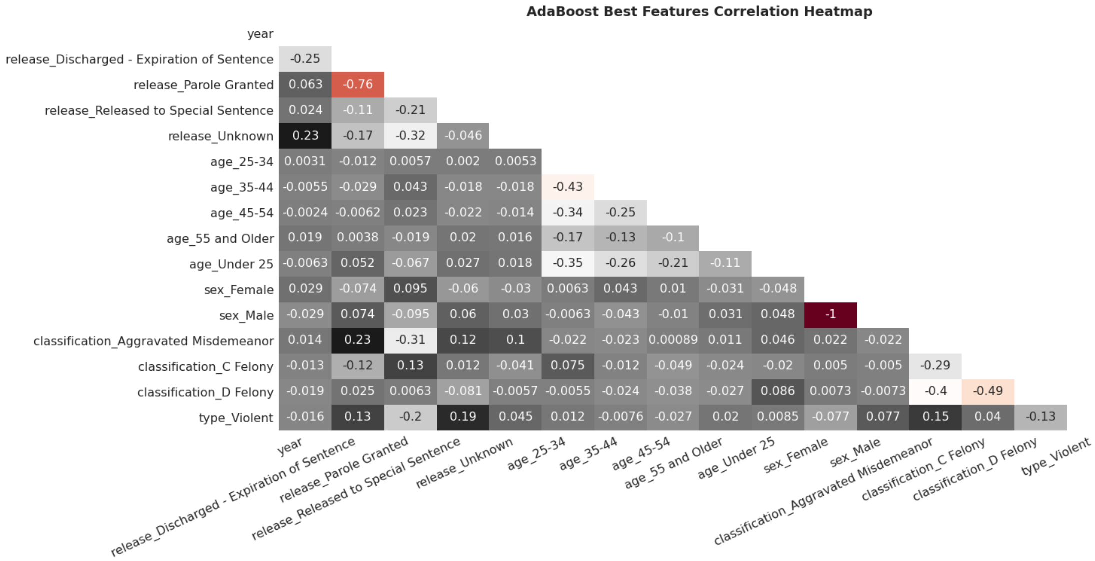
[56]:

```
          feature  Pearson Chi-square  p-value  Cramers V       Cramer_val
0         release            771.7566      0.0     0.1722           Strong
1            year             93.4152      0.0     0.0599             Weak
2             age            230.2147      0.0     0.0941             Weak
3             sex             28.9993      0.0     0.0334  No or very weak
4  classification             90.9506      0.0     0.0591             Weak
5            type            247.1988      0.0     0.0975             Weak
6         subtype            355.6772      0.0     0.1169         Moderate
```

[27]:

```
# Compute correlation coefficents using the pearson method and create a mask for
# input to seaborn to plot one-half of the heatmap
corrmat = best_features.corr(method='pearson')

ab_mask = np.triu(np.ones(corrmat.shape)).astype(np.bool)
title = "AdaBoost Best Features Correlation Heatmap"

plt.rcParams['font.size'] = (16)
with sns.axes_style("white"):
    f, ax = plt.subplots(figsize=(20, 10))
    ax = sns.heatmap(corrmat,
            mask=ab_mask,
            cmap="RdGy",
            annot=True,
            cbar=None,
             )

ax.set_xticklabels(
  ax.get_xticklabels(),
  rotation=25,
  horizontalalignment='right')
ax.set_title(title, fontsize=18, weight="bold" )

plt.show()
```

```
<ipython-input-27-2f90113f0912>:5: DeprecationWarning: `np.bool` is a deprecated
alias for the builtin `bool`. To silence this warning, use `bool` by itself.
Doing this will not modify any behavior and is safe. If you specifically wanted
the numpy scalar type, use `np.bool_` here.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  ab_mask = np.triu(np.ones(corrmat.shape)).astype(np.bool)
```


AdaBoost Best Features Correlation Heatmap
year
release_Discharged - Expiration of Sentence -0.25
release_Parole Granted 0.063 -0.76
release_Released to Special Sentence 0.024 -0.11 -0.21
release_Unknown 0.23 -0.17 -0.32 -0.046
age_25-34 0.0031 -0.012 0.0057 0.002 0.0053
age_35-44 -0.0055 -0.029 0.043 -0.018 -0.018 -0.43
age_45-54 -0.0024 -0.0062 0.023 -0.022 -0.014 -0.34 -0.25
age_55 and Older 0.019 0.0038 -0.019 0.02 0.016 -0.17 -0.13 -0.1
age_Under 25 -0.0063 0.052 -0.067 0.027 0.018 -0.35 -0.26 -0.21 -0.11
sex_Female 0.029 -0.074 0.095 -0.06 -0.03 0.0063 0.043 0.01 -0.031 -0.048
sex_Male -0.029 0.074 -0.095 0.06 0.03 -0.0063 -0.043 -0.01 0.031 0.048 -1
classification_Aggravated Misdemeanor 0.014 0.23 -0.31 0.12 0.1 -0.022 -0.023 0.00089 0.011 0.046 0.022 -0.022
classification_C Felony -0.013 -0.12 0.13 0.012 -0.041 0.075 -0.012 -0.049 -0.024 -0.02 0.005 -0.005 -0.29
classification_D Felony -0.019 0.025 0.0063 -0.081 -0.0057 -0.0055 -0.024 -0.038 -0.027 0.086 0.0073 -0.0073 -0.4 -0.49
type_Violent -0.016 0.13 -0.2 0.19 0.045 0.012 -0.0076 -0.027 0.02 0.0085 -0.077 0.077 0.15 0.04 -0.13
year
release_Discharged - Expiration of Sentence
release_Parole Granted
release_Released to Special Sentence
release_Unknown
age_25-34
age_35-44
age_45-54
age_55 and Older
age_Under 25
sex_Female
sex_Male
classification_Aggravated Misdemeanor
classification_C Felony
classification_D Felony
type_Violent