Chicago Crime ~~Recommendation~~ Prediction System

(From an Alternate Perspective)

Jarrod Johnson

University of Houston

Data Science II (COSC4337)

April 6, 2021

Abstract

After completing EDA of the data for the company, various models were assessed to make predictions related to the company’s future course of actions. While some of the more sophisticated models (MLP) indicated the potential to achieve higher accuracy, hardware limitations led me to focus on the decision tree model due to its overall higher accuracy and relative speed in completing parameter tuning and explorative iterations.

Note to company: Upgraded hardware could greatly improve performance.

Follow-Up

From the last report I continued working toward reducing dimensions and verifying features’ validity regarding the company’s goals. In examining Primary Type, I initially grouped them into two categories of ordinal values: felony=(1), and misdemeanor=(2). This led to an unexpected and interesting set of statistics. Because Arrest was a Boolean converted to [0,1], I was able to take the mean of all felony and misdemeanor events in Chicago from 2001-recent and see what their percentage of arrest was:

```
Primary Type
1    0.297819
2    0.260754
Name: Arrest, dtype: float64
```

This indicates felonies lead to arrest slightly more than misdemeanors, however, both values seem low regarding what the public may desire. This is a positive indicator for the company.

Later, I decided to expand Primary Type to provide the models more information to work with in the hopes of improving prediction accuracy. This gave me the opportunity to greatly expand the above statistic! This information should be very useful for the company in the general planning of activities to avoid and/or pursue (with a general recommendation to avoid activities related to sex and/or children). Meanwhile, the prediction system could give further credit to a choice.

```
Primary Type
ARSON                                0.105640
ASSAULT                              0.234022
BATTERY                              0.226699
BURGLARY                             0.055886
CONCEALED CARRY LICENSE VIOLATION    0.966667
CRIM SEXUAL ASSAULT                  0.136124
CRIMINAL DAMAGE                      0.062685
CRIMINAL SEXUAL ASSAULT              0.064276
CRIMINAL TRESPASS                    0.724931
DECEPTIVE PRACTICE                   0.141551
GAMBLING                             0.992500
HOMICIDE                             0.430506
HUMAN TRAFFICKING                    0.093750
INTERFERENCE WITH PUBLIC OFFICER     0.925858
INTIMIDATION                         0.160803
KIDNAPPING                           0.105942
LIQUOR LAW VIOLATION                 0.990114
MOTOR VEHICLE THEFT                  0.084038
NARCOTICS                            0.993846
NON - CRIMINAL                       0.157895
NON-CRIMINAL                         0.058480
NON-CRIMINAL (SUBJECT SPECIFIED)     1.000000
OBSCENITY                            0.808099
OFFENSE INVOLVING CHILDREN           0.186956
OTHER NARCOTIC VIOLATION             0.682171
OTHER OFFENSE                        0.200762
PROSTITUTION                         0.995681
PUBLIC INDECENCY                     0.994220
PUBLIC PEACE VIOLATION               0.678043
RITUALISM                            0.090909
ROBBERY                              0.094974
SEX OFFENSE                          0.291705
STALKING                             0.153145
THEFT                                0.114640
WEAPONS VIOLATION                    0.765973
Name: Arrest, dtype: float64
```

Models Used:

Decision Tree Classifier:

Logistic Regression:

Ridge Classifier:

Gaussian NB:

Bagging Classifier:

Random Forest Classifier:

MLP Classifier:

Initially, when the Primary Type was binary for felonies and misdemeanors, all these classifiers were used with default or minimal parameters to establish an initial baseline accuracy. The cross-validation function was applied to these classifiers to reveal potential overfitting (a wide-ranging set of values indicates overfitting), and for a well-balanced bias/variance trade off.

Performance Evaluation (Cross-Validation):

```
DT Scores:  [0.72459681 0.72405319 0.72453724 0.72421702 0.7245422
               0.72397127 0.72413759 0.72408229 0.72421882 0.72423371]

LGR Scores: [0.71972407 0.71972407 0.71972407 0.71972407 0.71972407
               0.71972407 0.71972159 0.71972337 0.71972337 0.71972337]

RDG Scores: [0.71972407 0.71972407 0.71972407 0.71972407 0.71972407
               0.71972407 0.71972159 0.71972337 0.71972337 0.71972337]

NB Scores:  [0.71972407 0.71972407 0.71972407 0.71972407 0.71972407
               0.71972407 0.71972159 0.71972337 0.71972337 0.71972337]

BG Scores:  [0.73214218 0.73177857 0.7319007 ]

RF Scores:  [0.72400738 0.72389355]
```

After Primary Type was expanded, there was considerable improvement with the bagging, random forest, and decision tree classifiers.

Performance Evaluation (Cross-Validation):

```
BG Scores:  [0.87733562 0.87701829 0.87674797]

RF Scores:  [0.8350105  0.83403781 0.83399834]

DT Scores:  [0.87671682 0.87724306 0.8768881  0.87656788 0.8764388
               0.87617816 0.87687817 0.87627963 0.87604133 0.87705162]
```

The bagging classifier had the least available parameters for tuning, so it was not explored further. The same choice was made for the logistic regression and naïve bayes classifiers since they provided no improvement after Primary Type expansion. That left decision tree and random forest (which is an expanded form of decision tree) available for further tuning. With the machine's performance between the two remaining classifiers dramatically favoring decision tree, I began investigating those parameters first.

Hyperparameter Tuning:

For decision tree, I started with the max_depth parameter with 5 being the initial preferred setting. Then, only later in the end of my investigating did I remove that limit all together allowing the classifier to allow nodes to be expanded until all leaves are pure or until all leaves contain less than min_samples_split samples. This led to a minor accuracy increase.

Also, I altered the criterion parameter between 'entropy' and 'gini', settling with 'gini'. Then, although not really being interested in changing the min_samples_split parameter, I tested it with 5 and 10, while also altering min_samples_leaf to 3 and 5. None of these produced an increase in accuracy.

Next, I altered the min_weight_fraction_leaf parameter, however, that only led to a decrease in accuracy, so it was left default. I preferred having the samples weighted equally, so that was a relief.

Lastly, I altered the max_features parameter. I didn't expect this to help since I had so few features to begin with and didn't want to reduce them further, but I wanted to make sure. Both alterations led to a minor decrease in accuracy.

Hyperparameter Tuning:

At this point I attempted a few alterations of parameters within the random forest classifier, but felt the time required to see results may not be worth the investment since processing the decision tree through cross validation might give me "similar" results as a random forest. I did later explore altering the n_estimators (the number of trees in the forest) and random_state parameters resulting in enormous time cost with no accuracy improvements.

Hyperparameter Tuning:

Since my data set consisted of 5.5 million rows, I was excited to work with the MLP (neural network) classifier. However, learning the parameters alone was worthy of a lesson. That, coupled with the growing time requirement for tuning iterations, left me falling short of my highly sought-after goal of exceeding 90% accuracy. The MLP consistently came in a few percentage points below the decision tree. I do suspect with further familiarization of parameter tuning in the MLP, and a faster machine, better results could be achieved.

Hyperparameter Tuning:

In the end, I decided to put all my efforts into further tuning the decision tree since it was able to produce outcomes relatively quick compared to the MLP and random forest. Because of that, I decided to perform loops of alterations on a few more parameters (ccp_alpha, min_impurity_decrease, min_samples_leaf). Slight gains of accuracy through min_impurity_decrease led me to keep an alteration there.

Performance Evaluation (metrics) & Model Comparison:

MLP (Neural Network) Metrics:

Confusion Matrix:

```
[[1215355   26889]
 [ 186558  297710]]
```

Classification Report:

```
             precision    recall  f1-score   support

           0       0.87      0.98      0.92   1242244
           1       0.92      0.61      0.74    484268

    accuracy                           0.88   1726512
   macro avg       0.89      0.80      0.83   1726512
weighted avg       0.88      0.88      0.87   1726512
```

F1 Score:

```
0.8679015963798768
```

```
Wall time: 2h 47min 2s
```

Decision Tree Metrics:

Cross-Validation

```
DT Scores:  [0.87731009 0.87795796 0.87749874 0.87697498 0.8769154
              0.87670689 0.87751612 0.87676119 0.87667431 0.87744134]
```

Confusion Matrix:

```
[[1237883    4361]
 [ 283548  200720]]
```

Classification Report:

```
             precision    recall  f1-score   support

           0       0.81      1.00      0.90   1242244
           1       0.98      0.41      0.58    484268

    accuracy                           0.83   1726512
   macro avg       0.90      0.71      0.74   1726512
weighted avg       0.86      0.83      0.81   1726512
```

Accuracy Score

```
0.8769947732769885
```

F1 Score

```
0.8078968481163529
```

```
Wall time: 3min 6s
```

Potential To Do’s:

Replace/support some visualizations with more sophisticated visualizations (plots/graphs): (overall model performance, parameter performance, metrics)

Detail confusion matrix.

Understand and explain difference in accuracy results within decision tree metrics.