Machine Learning for physicists

Arrest Prediction based on crime data from Los Angeles 2010-17

Ajeesh Garg

smajgarg@tu-dortmund.de

Contents

1 Motivation

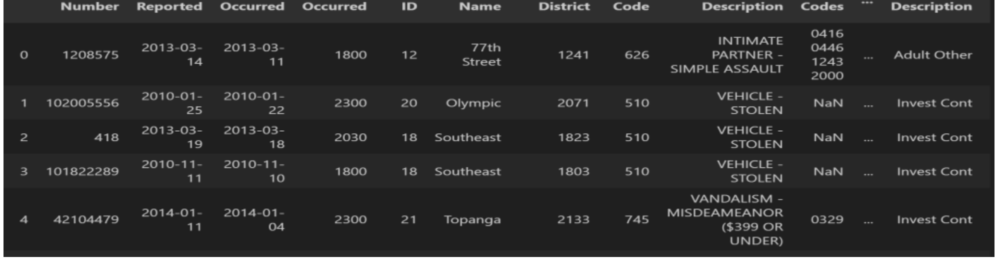
The motivation for my project arises from the alarming increase in crime rates across the United States. Recent Incidents of public shootings in schools [1] and the attack on Donald Trump [2] highlights the seriousness of this trend. One such notable surge in crime can be seen in California where one of the main reasons behind this increase is Proposition 47.

California's Proposition 47, introduced in 2014 [3] is a law which reclassified thefts under $950 from felonies to misdemeanours. The intention behind this legislation was to reduce prison populations and lower incarceration costs. While it did save the state money, it also led to a significant rise in shoplifting incidents [3] under $950. By making little thefts less serious, Proposition 47 encouraged individuals to steal minor goods knowing that consequences would be minimal.

Addressing such a problem of increasing crimes could be quite complex and requires innovative solutions. One possible solution of this problem could be to improve the probability of getting arrested. To explore this possibility further, I am focusing on a dataset from Los Angeles, one of the largest cities of California.

2 Description of Data

The data comes from Los Angeles Police Department (LAPD). This dataset spans from 2010 to 2017 and includes approximately 1.6 million instances and 26 columns, providing substantial dataset for analysis. Each instance in the dataset corresponds to each crime event and its corresponding characteristics can be seen as features of that instance. Some of the features of the dataset can be seen in Figure 1.



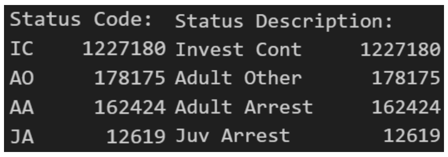
	DR Number	Date Reported	Date Occurred	Time Occurred	Area ID	Area Name	Reporting District	Crime Code	Crime Code Description	MO Codes	...	Status Description
0	1208575	2013-03-14	2013-03-11	1800	12	77th Street	1241	626	INTIMATE PARTNER - SIMPLE ASSAULT	0416 0446 1243 2000	...	Adult Other
1	102005556	2010-01-25	2010-01-22	2300	20	Olympic	2071	510	VEHICLE - STOLEN	NaN	...	Invest Cont
2	418	2013-03-19	2013-03-18	2030	18	Southeast	1823	510	VEHICLE - STOLEN	NaN	...	Invest Cont
3	101822289	2010-11-11	2010-11-10	1800	18	Southeast	1803	510	VEHICLE - STOLEN	NaN	...	Invest Cont
4	42104479	2014-01-11	2014-01-04	2300	21	Topanga	2133	745	VANDALISM - MISDEAMEANOR ($399 OR UNDER)	0329	...	Invest Cont

Figure 1 A glimpse of raw data

A glimpse of the dataset shows features like report number (DR Number), the date on which the crime occurred & when it was reported, Area ID, Area name, MO code linked to the type of crime committed, crime code related to the seriousness of the offense and much more. Their detailed description can be seen on the Los Angeles Police Department website [4].

Goal: The goal of this project is to classify crime events into two categories: arrest and non-arrest. By understanding the factors that lead to arrests, we can identify patterns and develop strategies to improve law enforcement efficiency and identify what circumstances lead to arrest and recognize the factors that contribute to system's weakness.

Target: The target variable for the dataset can be found under status code feature of the dataset [Figure 2] and their description has been given in adjacent status description column of the dataset. Some other categories were also present in negligible numbers. For our analysis, we have combined the categories – 'adult arrest' and 'juvenile arrest' as arrested and 'investigation cont' as not arrested. Based on this, a new binary feature namely 'arrest' has been created where '0' represents not arrested and '1' represents arrested.



```
Status Code:  Status Description:
IC    1227180 Invest Cont     1227180
AO     178175 Adult Other      178175
AA     162424 Adult Arrest     162424
JA      12619 Juv Arrest        12619
```

Figure 2 Category weightage of target vector

The glimpse of the target variable shows that the dataset have far more 'not arrest' instances over 'arrest', this could lead to a potential bias. So as to deal with this potential bias problem, method like class weights has been used which will be discussed later.

Why Machine Learning?

Machine Learning is an optimal solution for this problem due to 2 main reasons. Due to the large size of the dataset. Traditional techniques won't be sufficient to handle such a large dataset. Further, as the dataset contains lot of features and not just victim’s age and decent, Machine Learning can some hidden complex patterns in data that might be difficult or impossible for humans to detect.

3 Feature engineering and preprocessing

As stated already, the dataset used for the analysis have a lot of features which could potentially lead to Curse of Dimensionality problem. Furthermore, lot of features are also useless in various ways.

Missing values and variable Dimensionality: Some features have lot of Nan values. For example, features like Weapon code description and weapon code have around 66% of its instances as Nan and therefore, has been discarded. Feature like MO codes has varying dimensionality which makes it unfit for direct use. Therefore, it has also been removed.

Date time features: The data also has features like Date occurred and Date on which incident was reported. So, these have been subtracted to find out how long an incident took to be reported to the police. It sounds obvious to think if an incident went unreported for very long, the criminal would probably have fled by then. Therefore, a new feature “Days between” has been made and took into use.

Categorical Features: A lot of features although might look numerical but they only correspond to a finite group of unordered numbers.ie. they are categorical. Majority of features shown [Figure1,2] are indeed categorical.

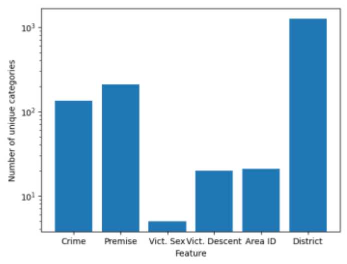


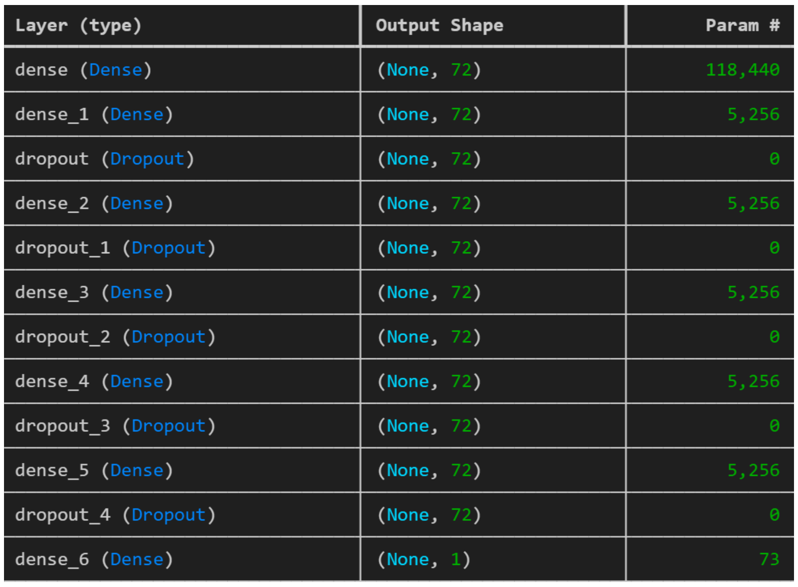
Figure 3 some potentially relavant categorical features

Since, Machine learning algorithms generally do not work well with categorical variables. Hence, lot of features like Area ID, Victim’ age, sex and descent has been one hot encoded. One hot encoding works by converting a categorical variable into a 1D binary array with size equal to the number of categories in that categorical variable. Since, the features are unordered, hot encoding is the most relevant technique. Additionally, after all the preprocessing, the data is left with around 1.1 million instances (1157749 precisely) and 13 useful features.

4 Choice and Motivation of Main Method

One limitation of one hot encoding is the feature space becomes very high dimensional. Traditional ML algorithms can't scale over thousand dimensions due to Curse of Dimensionality. since there are lot of features involved and the dataset is quite large with over a million instances after all the preprocessing, employing a dense neural network could be a nice approach to explore hidden patterns behind criminals not getting arrested.

The main method used for the task is a sequential feed forward neural network consisting of six dense layers including one input layer and one output layer. The input layer is set to 72 neurons and an input dimension is set to 1644, corresponding to the feature space of the input data. This high input space arises due to large number of hot encoded features. Further, to parameterise non linearities of the data,



Layer (type)	Output Shape	Param #
dense (Dense)	(None, 72)	118,440
dense_1 (Dense)	(None, 72)	5,256
dropout (Dropout)	(None, 72)	0
dense_2 (Dense)	(None, 72)	5,256
dropout_1 (Dropout)	(None, 72)	0
dense_3 (Dense)	(None, 72)	5,256
dropout_2 (Dropout)	(None, 72)	0
dense_4 (Dense)	(None, 72)	5,256
dropout_3 (Dropout)	(None, 72)	0
dense_5 (Dense)	(None, 72)	5,256
dropout_4 (Dropout)	(None, 72)	0
dense_6 (Dense)	(None, 1)	73

Figure 4 Network Architecture before Hyperparameter Optimisation

Activation function elu is employed in the network architecture. Elu is chosen as it works pretty similar to ReLu but it helps in avoiding vanishing gradient problem. Additionally, Alternate dropout layers are also used to avoid overfitting by setting 40% of input layers to zero. Thus, offering a better generalization to new - unseen data. The output layer consists of sigmoid function which is a binary classification activation function and outputs probabilities between 0 or 1. Also, Uniform number of layers is chosen to employ uniform learning throughout the network.

After that, the model is compiled using the Adam optimizer with a binary cross-entropy loss, and metrics including accuracy, precision, and AUC are used for model evaluation. The optimizer acts like a gradient descent algorithm by adjusting the model's weights based on the computed gradients of the loss function with respect to the weights, thus minimizing the loss and improving the model's performance and is chosen to be default. For the loss function, binary class Entropy is used because our target vector is binary (arrest or not arrest). This function measures the divergence between the predicted and the true probability, thus improving the model at every epoch. As a performance measure, the model is evaluated via three metrics: accuracy, precision, and AUC.

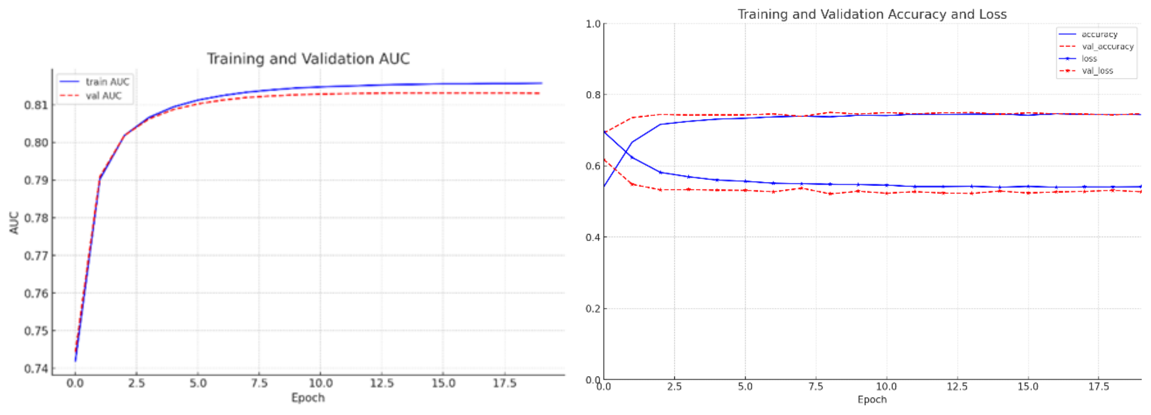


Figure 5 AUC-ROC Curve and loss-accuracy learning curve

Precision tells us out of all the instances, our model predicted as positive, how many were actually positive. Recall tells us out of all the actual positive instances, how many our model correctly identified as positive. AUC is more robust to sampling bias because it evaluates the model’s performance across all thresholds. This characteristic makes it a more reliable metric than accuracy.

The graph displays the AUC (Area Under the Curve) values for both training and validation data over 20 epochs. Further, the graph training AUC is slightly better than validation indicating model is performing well on both training and validation dataset showing no significant overfitting. Additionally, the learning curve on right also both the training accuracy and the validation accuracy steadily increase during training, while the training loss and the validation loss decrease. This is good. The validation curves are relatively far at first but later gets closer to each other showing model is generalizing very well. Can it be improved more based on hyperparameter tuning? Let’s perform the tuning and check.

5 Hyperparameter optimization

As every modern ML algorithm requires hyperparameters whose values are required to be set manually before the beginning of the training. Hyperparameter optimization is a way of fine tuning these hyperparameters to receive the most optimal results, as selecting best hyperparameters is always unique to the problem. However, it can be quite memory intensive task, which makes it a trade of between the computation power consumed and the performance improved.

Hyperparameter Training

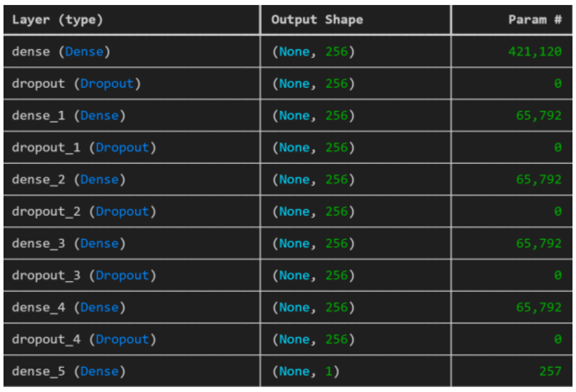
For our task, Keras Tuner library is chosen for building the hypermodel. Regarding the hypermodel’s parameter space: number of dense layers has been chosen to vary between 3-8, number of neurons has been chosen to vary upto a maximum of 256 neurons at each node and the dropout rate has been varied from 10% - 40% in multiples of 10. At the same time, super aggressive hyperparameter optimisation is avoided because the dataset used has over 1.1 million instances. A minimal improvement was seen with the smaller hyperspace indicates very limited room for further enhancement. Therefore, the hyperparameter space was not expanded.

Additionally, class weights has also been assigned, so as the model pays more attention to the minority class. The compute_class_weight function works by computing weights such that the model is penalized more for misclassifying the minority class (arrest class), thus helping in improving the model and minimise sampling bias.

For hyperparameter optimization, AUC score is used as the performance measure to select the best hyperparameter space because AUC is more robust to sampling bias because it evaluates the model’s performance across all thresholds. This characteristic makes it a more reliable metric.

Results of Optimization

The network architecture obtained after hyperparameter optimisation can be seen in the following figure. The network has 6 dense layers with 256 units at each (hidden layer and the best dropout was found to be 10% for every layer.



Layer (type)	Output Shape	Param #
dense (Dense)	(None, 256)	421,120
dropout (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 256)	65,792
dropout_1 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 256)	65,792
dropout_2 (Dropout)	(None, 256)	0
dense_3 (Dense)	(None, 256)	65,792
dropout_3 (Dropout)	(None, 256)	0
dense_4 (Dense)	(None, 256)	65,792
dropout_4 (Dropout)	(None, 256)	0
dense_5 (Dense)	(None, 1)	257

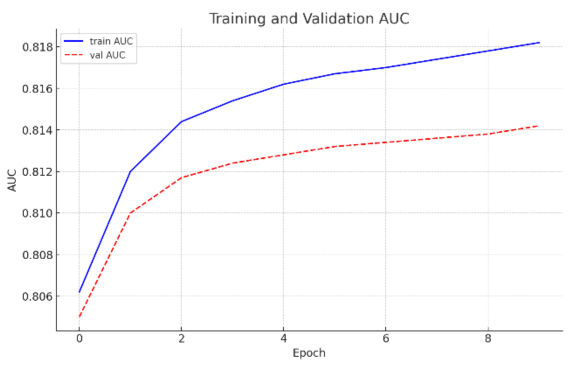


Figure 6 Network architecture and it’s corresponding ROC- AUC curve

The observed AUC curve is very similar to the previous one, whereas AUC score still lies around 0.81, showing with no significant improvement.

For further evaluation, two histograms representing the distribution of model predictions for two classes: 'no arrest' and 'arrest' has been plotted as well [Figure7]. Despite the class imbalance, the model has still been to discriminate between both labels to some extent. But at the same time, the overlap describes model had certain instances where it struggled to distinguish between the labels. Comparing both graphs, it can be seen that even the non-optimized model looks better on the histogram distribution but according to ROC Curve, optimized model is slightly better. This indicates there is not much room for improvement left.

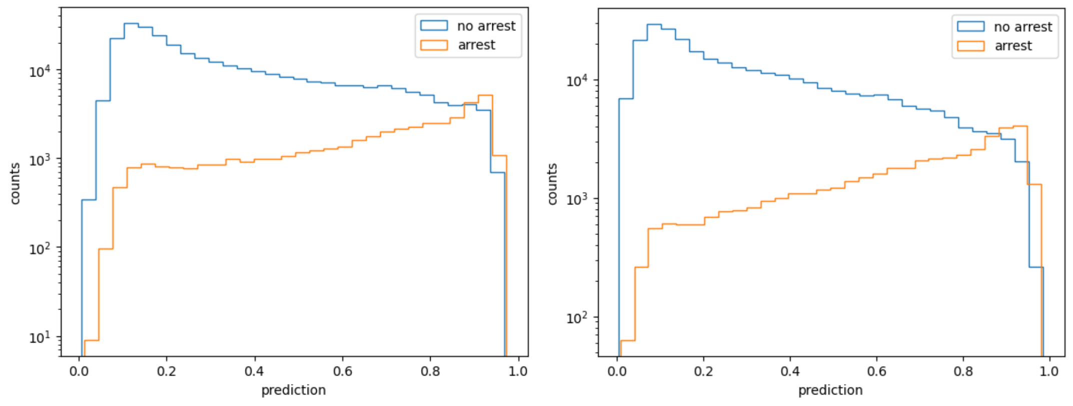


Figure 7 Histogram distribution of model's predictions to true values before (left) and after optimization (right)

Potential reasons for no more improvement could be because there are not much complex patters between the feature and the target vector. So, an alternative simpler machine learning algorithm could be tested on this dataset.

6 Alternate Method: Random Forests

As can be see, the task of classifying criminals as arrested or not arrested is a clearly defined classification task. The dataset has numerous features many of which may exhibit some complex relationships to the target. Given these characteristics, simpler algorithms such as Logistic Regression are unsuitable for the task. Additionally, the presence of many categorical features makes k-Nearest Neighbors (k-NN) an impractical choice as well. The next possible candidate for task is Random Forests Classification. Since, the algorithm uses ensemble of different random decision trees, it could help control sampling bias. This makes random forests an optimal algorithm for this task.

Class Imbalance control

Random Forest Classifier has a hyperparameter class weights and by setting it to balanced, it automatically adjusts the class weights. Class weights are adjusted such that lesser a target class instances, the more weight that class gets. This way, random forests classifier can perform well on our dataset which has far more ‘not arrest’ instances.

Dimensionality

After hot encoding features, we were left with a very high feature space. Since, our dataset has a lot of features, random forests is fitted on various subset of features but just by fitting it on 3 features – Crime code, premise code and Area ID. An AUC score of 0.81 is obtained. This shows that the task of classification is not very complicated and hence, a simpler algorithm with just 3 features is able to match the performance of a highly complex neural networks.

Overfitting checks

To verify that our model generalizes well and is not overfitting, we used 5-fold StratifiedKFold cross-validation where 4 subsets of data are used for training while the 5th unseen data is used for testing. Stratified sampling helped preserved class ratio while testing and training on subsets. After training, a mean ROC-AUC score of 0.81 was obtained, with a standard deviation of 0.0008 giving no indication of overfitting.

Performance Measure

For evaluating the performance of the Classifier, True positive rate vs False positive rate has been plotted for various thresholds. The key to reading the plot is that, the more the curve is near to the upper left corner, the stronger is the classifier. Further, the ideal AUC score for a classifier is 1 and a purely random classifier will have a score of 0.5. The score obtained by us was 0.8 signifying decent discriminative power with just 3 input features.

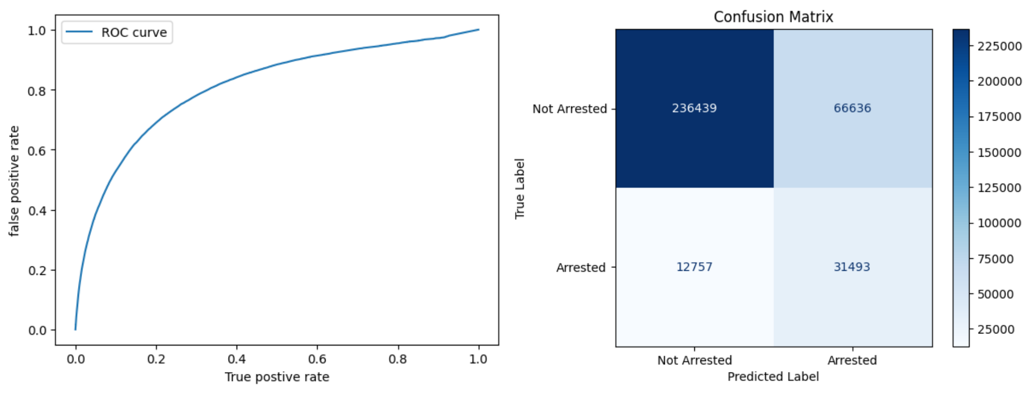


Figure 8 ROC curve and Confusion Matrix

A confusion matrix is a comparatively easier to calculate matrix which shows the number of True positives, False positives, True negatives and False negatives. However, confusion matrix is not a good performance measure for imbalanced dataset because for example, we can get very high accuracy even in the case when our classifier is just predicting the majority class the whole time. For this reason, measures like ROC curve are preferred for imbalanced datasets as it considers both precision and recall as performance measures.

7 Results & Conclusion

Comparison of NN with Alternate Method

As already mentioned, AUC is the best perimeter to measure the performance of the classifier employed on an imbalanced data. For comparison, ROC the curves from both Neural Network and Random forests is plotted on the same plot and DNN performed slightly better than Random Forests. This indicates that the model isn’t very complex as just 3 input features nearly matched the performance of the neural networks. In order to investigate further, HistGradient algorithm was also tested and was observed to perform slightly better than the performance of DNN.

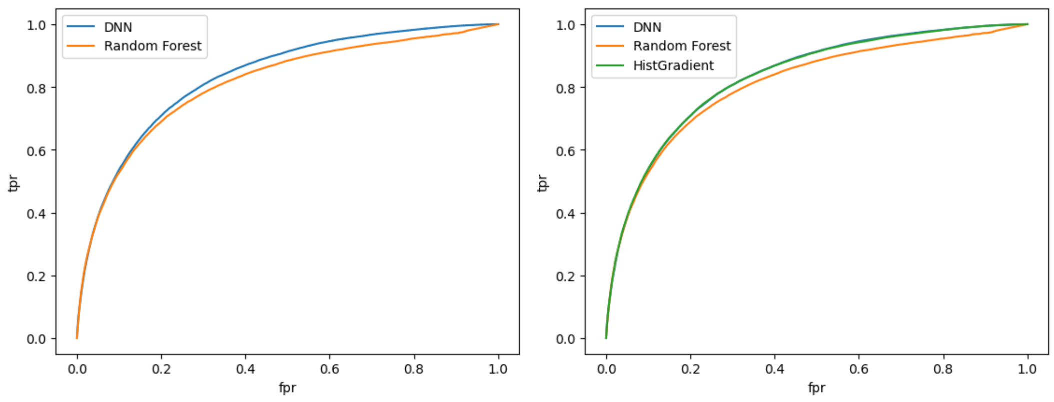


Figure 9 ROC Curves

The neural network is only found to perform slightly better than Random Forests even though Random Forests was trained only with 3 important features. This could be either because features are not relevant or there are dimensionality issues. To figure out the true importance of some potentially relevant features like victim's age, further analysis was carried out.

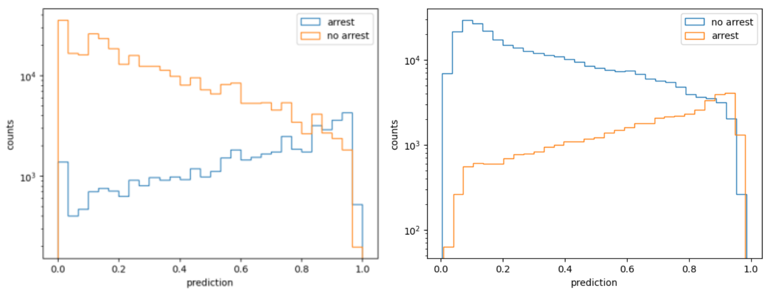


Figure 10 Random Forests (left) vs Neural networks (right)

Conclusion

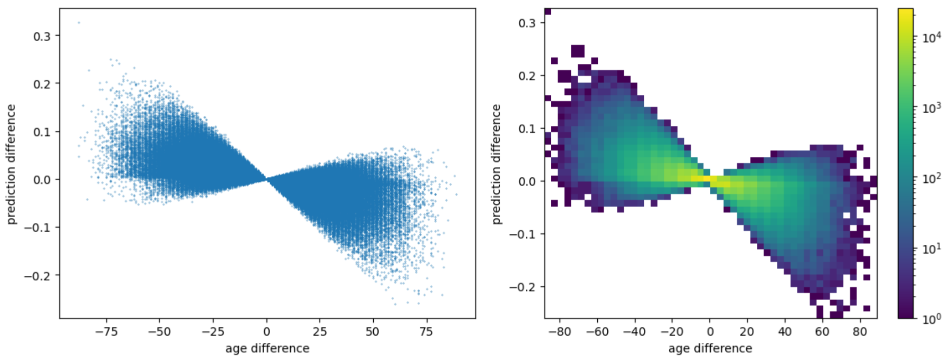


Figure 11 Variation of predictions with variation in age

The v shaped distribution shows that the model’s prediction error grows with larger deviations in age. The higher concentration of points at the centre shows that smaller age differences will result in smaller deviation between predictions while as the age difference increases or decreases, the prediction difference increases as well. For further evaluation of feature importance, some more suspected features used in neural network were selected. Then, a new test set was made where the column with a specific feature was shuffled. Then, how much the prediction changes after shuffling that feature was measured using the mean squared error (MSE).

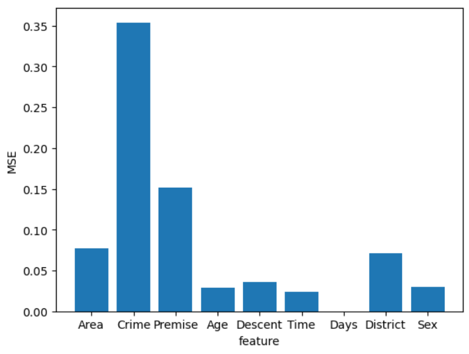
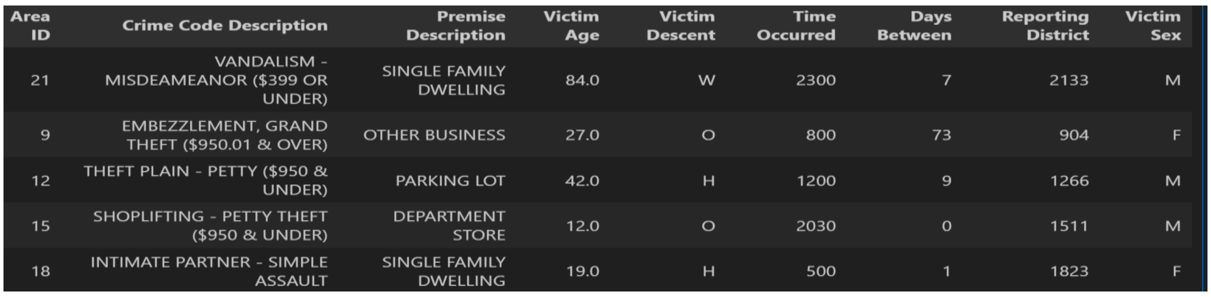


Figure 12 Histogram of some useful features as per neural networks

Since, the prediction changed the most with change in Crime instances, type of crime is very decisive feature behind deciding that the culprit will be arrested or not. Similarly, the location of crime – area and premise is also quite relevant. In conclusion, features with the highest mean squared error are the decisive ones.

Further, previous analysis using random forests showed that Crime code, premise code and Area ID are also quite decisive features by just using 3 features, random forests was able to get an AUC of 0.81.



Area ID	Crime Code Description	Premise Description	Victim Age	Victim Descent	Time Occurred	Days Between	Reporting District	Victim Sex
21	VANDALISM - MISDEAMEANOR ($399 OR UNDER)	SINGLE FAMILY DWELLING	84.0	W	2300	7	2133	M
9	EMBEZZLEMENT, GRAND THEFT ($950.01 & OVER)	OTHER BUSINESS	27.0	O	800	73	904	F
12	THEFT PLAIN - PETTY ($950 & UNDER)	PARKING LOT	42.0	H	1200	9	1266	M
15	SHOPLIFTING - PETTY THEFT ($950 & UNDER)	DEPARTMENT STORE	12.0	O	2030	0	1511	M
18	INTIMATE PARTNER - SIMPLE ASSAULT	SINGLE FAMILY DWELLING	19.0	H	500	1	1823	F

Figure 13 Final glimpse of dataset with features found to have discriminating power

Bibliography

1. https://www.gunviolencearchive.org/reports/mass-shooting
2. https://www.cnn.com/politics/live-news/election-biden-trump-07-13-24/index.html
3. https://www.nytimes.com/2024/07/23/us/california-crime-punishment-jail.html
4. https://data.lacity.org/Public-Safety/Crime-Data-from-2010-to-2019/63jg-8b9z/about_data
5. Aurélien Géron. (2023). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow (3rd ed.). O'Reilly Media